Jaguar 9.0

User Manual

Schrödinger Press

October 2015

Contents

Document Conventions xiii

Chapter 1: Introduction 1

1.1 About This Manual 1

1.2 Running Schrödinger Software 2

1.3 Starting Jobs from the Maestro Interface 4

1.4 Citing Jaguar in Publications 5

Chapter 2: Running Jaguar From Maestro 7

2.1 Sample Calculation 7

2.2 The Jaguar Panel 10

2.3 The Edit Job Dialog Box 11

2.4 Molecular Structure Input 13

2.4.1 Cartesian Format for Geometry Input 13

2.4.2 Variables in Cartesian Input 13

2.4.3 Constraining Cartesian Coordinates 14

2.4.4 Z-Matrix Format for Geometry Input 15

2.4.5 Variables and Dummy Atoms in Z-Matrix Input 16

2.4.6 Constraining Z-Matrix Bond Lengths or Angles 17

2.4.7 Counterpoise Calculations 18

2.4.8 Specifying Coordinates for Hessian Refinement 19

2.5 Reading Files 20

2.6 Setting Charge and Multiplicity 21

2.7 Cleaning up Molecular Geometries 22

2.7.1 Quick Geometry Optimization 22

2.7.2 Symmetrization 23

2.8 Writing Files 24

2.9 Running Jobs 25

2.10 Running Jaguar Batch Jobs 26

2.11 Output 29

Chapter 3: Options 31

3.1 Input Structures 31

3.2 Basis Sets 33

3.3 Density Functional Theory (DFT) Settings 38

3.3.1 LDA (Local Density Functionals) 41

3.3.2 Gradient-Corrected Functionals 41

3.3.3 Hybrid Functionals 42

3.3.4 Long-Range-Corrected Functionals 44

3.3.5 B3LYP-MM A Posteriori-Corrected Functional 46

3.3.6 B3LYP-LOC A Posteriori-Corrected Functional 47

3.3.7 Constrained DFT Calculations 49

3.4 Hartree-Fock and TDHF Settings 50

3.5 Local MP2 Settings 50

3.6 Hamiltonian Settings 53

3.7 SCF Settings 53

3.7.1 Accuracy Level 54

3.7.2 Initial Guess 55

3.7.3 Convergence Criteria 56

3.7.4 Convergence Methods 56

3.7.5 Orbital Treatment 57

3.8 Solvation Settings 58

3.8.1 Poisson-Boltzmann Solvation Model 58

3.8.2 Solvation Models 8 and 6 (SM8 and SM6) 61

3.9 Properties 62

3.9.1 Charges from Electrostatic Potential Fitting 62

3.9.2 Mulliken Population Analysis 64

3.9.3 Multipole Moments 65

3.9.4 Natural Bond Orbital (NBO) Analysis 65

3.9.5 Polarizability and Hyperpolarizability 66

3.9.6 NMR Shielding Constants 67

3.9.7 Atomic Fukui Indices 68

3.9.8 Stockholder Charges 68

3.9.9 Molecular Properties from SM6 and SM8 Calculations 68

3.10 Frequencies and Related Properties 69

3.10.1 Frequencies 70

3.10.2 Atomic Masses 70

3.10.3 Scaling of Frequencies 70

3.10.4 Animation of Frequencies 72

3.10.5 Infrared and Raman Intensities 73

3.10.6 Thermochemical Properties 74

3.10.7 Vibrational Circular Dichroism 74

3.11 Surfaces 75

3.11.1 Generating Surfaces 77

3.11.2 Displaying Surfaces in Maestro 78

3.12 Plotting Spectra 80

3.13 Options for Extra Output 82

3.13.1 File Output Options 85

3.13.2 Output Options for Orbitals 86

Chapter 4: Optimizations and Scans **91**

4.1 Geometry Optimization: The Basics 91

4.1.1 SCF and Geometry Convergence 92

4.1.2 The Initial Hessian 94

4.1.3 Coordinate Systems 95

4.2 Constraining Coordinates 96

4.2.1 Freezing Specific Coordinates 96

4.2.2 Applying Harmonic Constraints 97

4.2.3 Applying Constraints by Using Variables 98

4.2.4 Applying Dynamic Constraints 99

4.3 Transition-State Optimizations 100

4.3.1 Transition-State Search Method 100

4.3.2 Specifying Structures for the Reaction 102

4.3.3 Specifying the Symmetry, State, and Basis Set 103

4.3.4 Searching Along a Particular Hessian Eigenvector 103
4.3.5 Refinement of the Initial Hessian 104

4.4 Geometry Scans 106
4.4.1 Setting up Scans in Maestro 106
4.4.2 Setting up Input Files for Scans 108
4.4.3 Restarting Scans 109
4.4.4 Scan Results 109

4.5 Intrinsic Reaction Coordinate Calculations 110

Chapter 5: Jaguar Workflows **115**

5.1 Binding Energies of Hydrogen-Bonded Complexes 115

5.2 Fukui Functions 117

5.3 Vibrational Circular Dichroism (VCD) Spectra 119

5.4 Heat of Formation 121

5.5 Reaction Energies 122

Chapter 6: Output **125**

6.1 Summarizing Jaguar Results 125
6.1.1 Reporting Final Results From One or More Jobs 130
6.1.2 Reporting Intermediate Results 131
6.1.3 Reporting Results for Each Atom 131

6.2 Output From a Standard HF Calculation 132

6.3 Output File Content for Various Calculation Types 136
6.3.1 DFT 136
6.3.1.1 B3LYP-LOC 137
6.3.2 ZORA Relativistic Calculations 137
6.3.3 LMP2 138
6.3.4 Geometry or Transition-State Optimization 138
6.3.5 Solvation 141
6.3.5.1 Output from a PBF Calculation 141
6.3.5.2 Output from an SM8 Calculation 145

6.3.6 Geometry Optimization in Solution 148
6.3.7 Properties 149
6.3.7.1 Multipole Moments and Charge Fitting 149
6.3.7.2 Polarizabilities and Hyperpolarizabilities 150
6.3.7.3 Electron Density 152
6.3.7.4 Mulliken Populations 152
6.3.7.5 Atomic Fukui Indices 153
6.3.7.6 NBO Calculations 156
6.3.7.7 ESP Analysis 156
6.3.8 Frequency, IR Intensity, and Thermochemistry Output 157
6.3.9 CIS Calculations 159
6.3.10 Basis Set 160
6.3.11 Jaguar Reactions 163
6.3.12 Methods 163
6.4 The Log File 164
Chapter 7: Using Jaguar **167**
7.1 Choosing an Initial Guess 167
7.1.1 Overview 167
7.1.2 Initial Guess for Molecules Containing Transition Metals 168
7.2 SCF Convergence 170
7.3 Geometry Optimization 171
7.4 Transition-State Searches 173
7.5 Restarting Jobs and Using Previous Results 174
7.6 Conformational Searches 176
7.7 Timings for Typical Jobs 177
7.8 Noncovalent Interactions 178
7.9 Generating Input Files for GAUSSIAN 180
7.9.1 Other Jaguar Options for the `.gau` File 180
7.9.2 Writing Orbitals for GAUSSIAN 181

Chapter 8: Theory 183

8.1 The Pseudospectral Method 183

8.2 Pseudospectral Local MP2 Techniques 185

8.3 Density Functional Theory 188

8.3.1 Long-Range-Corrected Density Functionals 189

8.4 The ZORA Hamiltonian 190

Chapter 9: The Jaguar Input File 195

9.1 General Description of the Input File 195

9.1.1 Input File Format 195

9.1.2 Sections Describing the Molecule and Calculation 196

9.2 The zmat, zmat2, and zmat3 Sections 198

9.3 The zvar, zvar2, and zvar3 Sections 200

9.4 The coord and connect Sections 201

9.4.1 Constrained Coordinates 202

9.4.2 Specifying Bonds for Internal Coordinates with a connect Section 203

9.5 The gen Section 204

9.5.1 Units Keywords 204

9.5.2 Covalent Bonding Keyword 205

9.5.3 Molecular State Keywords (Charge and Multiplicity) 205

9.5.4 Atomic Mass Keyword 205

9.5.5 Symmetry-Related Keywords 206

9.5.6 LMP2 Keywords 206

9.5.7 Relativistic Hamiltonian Keyword 208

9.5.8 DFT Keywords 208

9.5.9 TDHF and TDDFT Keywords 216

9.5.10 Geometry Optimization and Transition-State Keywords 217

9.5.11 Geometry Scan Keywords 227

9.5.12 Intrinsic Reaction Coordinate (IRC) Keywords 228

9.5.13 Solvation Keywords 230

9.5.14 pKa Keywords 235

9.5.15 Properties Keywords ... 236
9.5.16 Frequency-Related Keywords ... 241
9.5.17 Electron Transfer Keywords ... 244
9.5.18 Basis Set Keywords ... 244
9.5.19 Keywords for SCF Methods ... 245
9.5.20 Initial Guess Keywords ... 251
9.5.21 Localization Keywords ... 253
9.5.22 File Format Conversion Keywords ... 254
9.5.23 Standard Output Keywords ... 257
9.5.24 File Output Keywords ... 259
9.5.25 Output Keywords for Each Iteration ... 261
9.5.26 Orbital Output Keywords ... 261
9.5.27 Grid and Dealiasing Function Keywords ... 263
9.5.28 Memory Use Keywords ... 265
9.5.29 Plotting Keywords ... 267

9.6 The lmp2 Section ... 270

9.7 The relativity Section ... 272

9.8 The atomic Section ... 273

9.8.1 General Format of the **atomic** Section ... 273
9.8.2 Keywords That Specify Physical Properties ... 274
9.8.3 Basis, Grid, Dealiasing Function, and Charge Usage for Individual Atoms ... 276
9.8.4 Defining Fragments ... 281

9.9 The hess Section ... 282

9.10 The guess and guess_basis Sections ... 283

9.11 The pointch Section ... 284

9.12 The efields Section ... 285

9.13 The egridpt Section ... 285

9.14 The ham Section ... 286

9.15 The orbman Section ... 286

9.16 The cdft Section ... 287

9.17 The loc_ib Section 288

9.18 The echo Section 288

9.19 The path Section 288

9.20 NBO Sections 291

Chapter 10: Other Jaguar Files **293**

10.1 The Basis Set File 293

10.1.1 Basis Set Format 293

10.1.2 Effective Core Potential Format 295

10.1.3 Customizing Basis Sets 297

10.2 The Initial Guess Data File 298

10.3 The Dealiasing Function File 300

10.3.1 File Format and Description 301

10.3.2 Sample File 303

10.4 The Grid File 305

10.4.1 File Format and Description 305

10.5 The Cutoff File 308

10.6 The PBF Radii File 311

10.6.1 Summary of Ark Syntax 311

10.6.2 The PBF Radii File Structure 312

Chapter 11: Running Jobs **315**

11.1 The jaguar Command 315

11.1.1 Running a Jaguar Job From the Command Line 317

11.1.2 Converting File Formats 318

11.2 Running Multiple Jobs with Batch Scripts 322

11.2.1 Batch Input File Format 322

11.2.2 Running Batch Jobs 326

11.2.3 Batch Input File Examples 327

11.2.3.1 Pipelined Jobs 327

11.2.3.2 Running Jobs from Input in a Specified Directory 328

11.3 Running Complex Jobs with Python Scripts 329

11.3.1 counterpoise.py 329

11.3.2 hydrogen_bond.py 330

11.3.3 h_abstraction.py 331

11.3.4 distributed_scan.py 332

11.3.5 fukui.py 333

11.3.6 vcd.py 333

11.3.7 canonical.py 334

11.3.8 jreactions.py 334

11.4 Running Parallel Jobs 335

11.4.1 Automatic Selection of Distribution and Parallel Options 335

11.4.2 Distribution of Jobs 335

11.4.3 Message Passing Interface (MPI) 336

11.4.4 OpenMP threads 337

11.4.5 Considerations for Running Parallel Jobs 338

Chapter 12: Troubleshooting **341**

12.1 Problems Getting Started 341

12.1.1 The SCHRODINGER Environment Variable 341

12.1.2 Problems Starting Maestro 342

12.1.3 Problems Related to Your Temporary Directory 343

12.1.4 Problems Running Jaguar Calculations on Other Hosts 344

12.2 Other Problems 345

Chapter 13: The pK_a Prediction Module **347**

13.1 Theory of pK_a Calculation 348

13.1.1 Ab initio Quantum Chemical Calculation of pK_a Values 348

13.1.2 Empirical Corrections 350

13.2 Predicting pK_a Values in Complex Systems 351

13.2.1 Conformational Flexibility 351

13.2.2 Equivalent Sites 352

13.2.3 Multiple Protonation Sites 353

13.3 Training Set Results 353

13.3.1 Jaguar pK_a Shell Model 358

13.3.2 Zwitterions 359

13.4 Running pK_a Calculations 360

13.4.1 Running pK_a Calculations from Maestro 360

13.4.2 Jaguar Input Files for pK_a Calculations 362

13.4.3 Running pK_a Calculations from the Command Line 363

13.4.4 Monitoring pK_a Calculations 365

13.4.5 Choice of Initial Geometry 366

13.4.6 Recalculating pKa Values with New Parameters 367

13.5 Developing Your Own pK_a Correction Parameters 368

Getting Help **371**

References **375**

Index **389**

Keyword Index **403**

Document Conventions

In addition to the use of italics for names of documents, the font conventions that are used in this document are summarized in the table below.

Font	Example	Use
Sans serif	Project Table	Names of GUI features, such as panels, menus, menu items, buttons, and labels
Monospace	`$SCHRODINGER/maestro`	File names, directory names, commands, environment variables, command input and output
Italic	*filename*	Text that the user must replace with a value
Sans serif uppercase	CTRL+H	Keyboard keys

Links to other locations in the current document or to other PDF documents are colored like this: Document Conventions.

In descriptions of command syntax, the following UNIX conventions are used: braces { } enclose a choice of required items, square brackets `[ ]` enclose optional items, and the bar symbol | separates items in a list from which one item must be chosen. Lines of command syntax that wrap should be interpreted as a single command.

File name, path, and environment variable syntax is generally given with the UNIX conventions. To obtain the Windows conventions, replace the forward slash / with the backslash \ in path or directory names, and replace the $ at the beginning of an environment variable with a % at each end. For example, `$SCHRODINGER/maestro` becomes `%SCHRODINGER%\maestro`.

Keyboard references are given in the Windows convention by default, with Mac equivalents in parentheses, for example CTRL+H (⌘H). Where Mac equivalents are not given, COMMAND should be read in place of CTRL. The convention CTRL-H is not used.

In this document, to *type* text means to type the required text in the specified location, and to *enter* text means to type the required text, then press the ENTER key.

References to literature sources are given in square brackets, like this: [10].

Introduction

1.1 About This Manual

The *Jaguar User Manual* is intended to help you perform ab initio calculations for a variety of methods, parameters, and calculated properties. Jaguar can be run from the command line or from the Maestro graphical user interface (GUI). Online help is available in the GUI, although the information in this manual is generally more comprehensive.

A set of tutorial exercises that introduces many Jaguar features is available in the *Jaguar Quick Start Guide*. For a summary of recommended settings for various calculations, see the quick reference sheet, *Recommended Settings for Jaguar*.

Chapter 2 contains information you will need to run Jaguar, including information about using the GUI, geometry input formats, specifying file names for input and output, displaying molecular geometries, symmetrizing geometries, and setting run-time parameters, such as the machine that will perform the calculation. We suggest you start by trying the sample calculation in Section 2.1. If the calculation runs successfully, you can proceed to the rest of the chapter to learn how to input molecular structures and run jobs. If you have problems starting Maestro or running the sample calculation, see the troubleshooting information in Chapter 12.

Chapters 3 and 4 describe the available calculation options, which allow you to specify which properties you want the program to calculate and which methods you want it to use. Chapter 3 includes information on using Møller-Plesset second-order perturbation theory and density functional theory (DFT) techniques; calculating solvation energies, vibrational frequencies, hyperpolarizabilities, multipole moments, and other properties; generating molecular orbitals; fitting charges; specifying basis sets; the output or printing options; and various other options. Chapter 4 describes optimizations of the molecular structure, transition-state searches, and geometry scans.

Chapter 5 describes a set of workflows that calculate properties requiring multiple Jaguar calculations, such as counterpoise-corrected dissociation, hydrogen bonding, VCD spectra, Fukui functions, and more.

Chapter 6 describes how to summarize Jaguar output. The output file containing the primary Jaguar output is first described for cases where no Output options have been selected. Next, the output given when various Output settings are turned on is explained. Finally, the log file is described.

Chapter 7 contains tips and suggestions for using Jaguar. The chapter includes some general tips for different sorts of calculations: a description of how to restart calculations, how to incorporate results from previous runs, and some tips on using both Jaguar and GAUSSIAN.

Chapter 8 describes some of the theory behind the pseudospectral method and the electron correlation methods used in Jaguar. This chapter includes information on pseudospectral implementations of local MP2 techniques, and a brief description of density functional theory.

Chapter 9 describes the Jaguar input file in detail. You may find this chapter especially useful if you want to run some jobs without using the GUI. Chapter 10 describes other Jaguar files that are necessary for calculations. You may skip Chapter 9 and Chapter 10 if you want to run all jobs from the GUI, but you might want to skim them anyway to find out more about Jaguar and the methods it uses.

Chapter 11 provides information on submitting jobs from the command line, running multiple Jaguar jobs using batch scripts, and running calculations on parallel computers.

Chapter 12 contains troubleshooting hints concerning various problems you might encounter, especially when first setting up Jaguar on your system. Chapter 13 describes the pK_a calculation module.

1.2 Running Schrödinger Software

Schrödinger applications can be run from a graphical interface or from the command line. The software writes input and output files to a directory (folder) which is termed the *working directory*. If you run applications from the command line, the directory from which you run the application is the working directory for the job.

Linux:

To run any Schrödinger program on a Linux platform, or start a Schrödinger job on a remote host from a Linux platform, you must first set the `SCHRODINGER` environment variable to the installation directory for your Schrödinger software. To set this variable, enter the following command at a shell prompt:

csh/tcsh:	`setenv SCHRODINGER` *installation-directory*
bash/ksh:	`export SCHRODINGER=`*installation-directory*

Once you have set the `SCHRODINGER` environment variable, you can run programs and utilities with the following commands:

```
$SCHRODINGER/program &
$SCHRODINGER/utilities/utility &
```

You can start the Maestro interface with the following command:

```
$SCHRODINGER/maestro &
```

It is usually a good idea to change to the desired working directory before starting the Maestro interface. This directory then becomes the working directory.

Windows:

The primary way of running Schrödinger applications on a Windows platform is from a graphical interface. To start the Maestro interface, double-click on the Maestro icon, on a Maestro project, or on a structure file; or choose Start → All Programs → Schrodinger-2015-4 → Maestro. You do not need to make any settings before starting Maestro or running programs. The default working directory is the Schrodinger folder in your Documents folder.

If you want to run applications from the command line, you can do so in one of the shells that are provided with the installation and have the Schrödinger environment set up:

- Schrödinger Command Prompt—DOS shell.
- Schrödinger Power Shell—Windows Power Shell (if available).

You can open these shells from Start → All Programs → Schrodinger-2015-4. You do not need to include the path to a program or utility when you type the command to run it. If you want access to Unix-style utilities (such as `awk`, `grep`, and `sed`), preface the commands with `sh`, or type `sh` in either of these shells to start a Unix-style shell.

Mac:

The primary way of running Schrödinger software on a Mac is from a graphical interface. To start the Maestro interface, click its icon on the dock. If there is no Maestro icon on the dock, you can put one there by dragging it from the SchrodingerSuite2015-4 folder in your Applications folder. This folder contains icons for all the available interfaces. The default working directory is the Schrodinger folder in your Documents folder (`$HOME/Documents/Schrodinger`).

Running software from the command line is similar to Linux—open a terminal window and run the program. You can also start Maestro from the command line in the same way as on Linux. The default working directory is then the directory from which you start Maestro. You do not need to set the `SCHRODINGER` environment variable, as this is set in your default environment on installation. To set other variables, on OS X 10.7 use the command

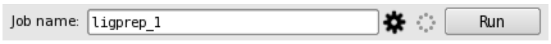
```
defaults write ~/.MacOSX/environment variable "value"
```

and on OS X 10.8, 10.9, and 10.10 use the command

```
launchctl setenv variable "value"
```

1.3 Starting Jobs from the Maestro Interface

To run a job from the Maestro interface, you open a panel from one of the menus (e.g. Tasks), make settings, and then submit the job to a host or a queueing system for execution. The panel settings are described in the help topics and in the user manuals. When you have finished making settings, you can use the Job toolbar to start the job.



Job name: ligprep_1

You can start a job immediately by clicking Run. The job is run on the currently selected host with the current job settings and the job name in the Job name text box. If you want to change the job name, you can edit it in the text box before starting the job. Details of the job settings are reported in the status bar, which is below the Job toolbar.

If you want to change the job settings, such as the host on which to run the job and the number of processors to use, click the Settings button. (You can also click the arrow next to the button and choose Job Settings from the menu that is displayed.)

You can then make the settings in the Job Settings dialog box, and choose to just save the settings by clicking OK, or save the settings and start the job by clicking Run. These settings apply only to jobs that are started from the current panel.

If you want to save the input files for the job but not run it, click the Settings button and choose Write. A dialog box opens in which you can provide the job name, which is used to name the files. The files are written to the current working directory.

The Settings button also allows you to change the panel settings. You can choose Read, to read settings from an input file for the job and apply them to the panel, or you can choose Reset Panel to reset all the panel settings to their default values.

You can also set preferences for all jobs and how the interface interacts with the job at various stages. This is done in the Preferences panel, which you can open at the Jobs section by choosing Preferences from the Settings button menu.

Note: The items present on the Settings menu can vary with the application. The descriptions above cover all of the items. Jaguar has an Edit item and extra functions for the Read and Write items, which are described later in the manual.

The icon on the Job Status button shows the status of jobs for the application that belong to the current project. It starts spinning when the first job is successfully launched, and stops spinning when the last job finishes. It changes to an exclamation point if a job is not launched successfully.

Clicking the button shows a small job status window that lists the job name and status for all active jobs submitted for the application from the current project, and a summary message at the bottom. The rows are colored according to the status: yellow for submitted, green for launched, running, or finished, red for incorporated, died, or killed. You can double-click on a row to open the Monitor panel and monitor the job, or click the Monitor button to open the Monitor panel and close the job status window. The job status is updated while the window is open. If a job finishes while the window is open, the job remains displayed but with the new status. Click anywhere outside the window to close it.

Jobs are run under the Job Control facility, which manages the details of starting the job, transferring files, checking on status, and so on. For more information about this facility and how it operates, as well as details of the Job Settings dialog box, see the *Job Control Guide*.

1.4 Citing Jaguar in Publications

The use of this product should be acknowledged in publications as:

Jaguar, version 9.0, Schrödinger, LLC, New York, NY, 2015.

Running Jaguar From Maestro

The Maestro interface to Jaguar can simplify the preparation and submission of jobs. You can run Maestro on one machine, display it to another, and start a Jaguar calculation on yet another machine. The main part of the interface is the Jaguar panel, which you use to prepare job input. Without the Jaguar panel, you would have to create input files with particular formats in order to run Jaguar. Maestro creates these input files for you, based on information you provide, and submits the job. This frees you from learning the input format and program sequences, and instead allows you to concentrate on the science.

Try the sample calculation in Section 2.1 to get some experience running Jaguar and to make sure your system is set up properly. If you have problems starting or using Maestro or performing the calculation, you may be able to solve them using the troubleshooting suggestions in Chapter 12. If any problems persist, contact your system manager or Schrödinger.

The rest of this chapter describes the basics of running Jaguar from Maestro, including entering a geometry and submitting a job. Jaguar has a wide range of options for different kinds of calculations. Setting these from Maestro is described in Chapter 3. Jaguar can perform many kinds of calculations, such as geometry optimizations, transition-state searches, and various coordinate scans. Setting up these calculations is described in Chapter 4. Jaguar also provides special capabilities for various kinds of calculations: binding energies of hydrogen-bonded complexes and other noncovalent complexes, evaluation of Fukui functions (described in Chapter 5), and pK_a calculations (described in Chapter 13.)

The footnotes describe Jaguar input file keywords and sections that correspond to particular GUI settings. If you are working from Maestro, you can ignore these footnotes, but you may later find them helpful if you decide to use input files to submit jobs without using Maestro, or if you want to edit keywords directly using the Edit Job dialog box.

2.1 Sample Calculation

This section provides instructions for running a sample calculation on the water molecule. The sample calculation runs only if Jaguar has been correctly installed. If the calculation does not run, try the suggestions in Chapter 12, or see your system manager or the person who installed Jaguar at your site. Contact Schrödinger if you cannot resolve the installation problems.

First, log on to a machine where the Maestro and Jaguar software is installed, and start Maestro according to the instructions in Section 1.2 on page 2. Once Maestro is running, choose Applications → Jaguar → Single Point Energy or Tasks → Quantum → Mechanics Single Point Energy to open the Jaguar panel for a single point energy calculation (see Figure 2.1 on page 10).

The next step is to enter a molecular geometry (structure). You can enter the structure by hand or read it from a file.

To enter the structure by hand, you can use the Edit Job window. Click the Edit button, select the Structure option, then click in the text area and type the following lines:

```
O     0.0         0.0    -0.1135016
H1    0.753108    0.0     0.4540064
H2   -0.753108    0.0     0.4540064
```

The labels begin with element symbols, `O` and `H`. The numerals `1` and `2` appended to the hydrogen labels distinguish between the atoms. The next three numbers on each line give the *x*, *y*, and *z* Cartesian coordinates of the atoms in the geometry, in angstroms. The number of spaces you type does not matter, as long as you use at least one space to separate each item. When you finish entering the water geometry, click OK.

To read in the structure, click Read, then navigate to the following directory:

`$SCHRODINGER/jaguar-v`*version*`/samples`

where *version* is the version number of your Jaguar software. Select `H2O.in` from the file list, and click OK.

Whichever method of entry you chose, the molecular structure should now be shown in the Workspace. If you entered the geometry by hand, the structure is a scratch entry. You can run the job with a scratch entry, but the results will not be incorporated into the Project Table unless you make it a named entry. To do so, click the Create Entry button on the Workspace toolbar, and name the entry H2O.

When you finish setting up your calculation, click the Settings (gear) button. The Start dialog box is displayed (see Figure 2.6 on page 26). In this dialog box you can make settings for running the job. In the Output section, you can select an option for incorporation of results. Choose Replace existing entries.

In the Job section, you can provide a job name, select the execution host, the number of processors, and the scratch directory. The entry name, H2O, appears in the Name text box. The names of the input, output, and log files for your job depend on the name you provide: the Jaguar input file is named *jobname*`.in`, the output file is named *jobname*`.out`, and the log file is named *jobname*`.log`, where *jobname* is the text that appears in the Name text box.

The default execution host (the machine that the job will run on) is selected in the Host option menu. This default is `localhost`, which means the machine on which Maestro is running. The host name is followed by the number of available processors in parentheses. If Jaguar is installed on more than one machine at your site, you can change the choice of execution host by selecting another host from the option menu. The Scratch directory option menu displays the directory on the execution host that will be used during the calculation to store temporary files. You should check that the directory already exists on the execution host. If it does not exist, you should create it.

You should not need to change any other settings. Click the Run button to start the job. If you want to monitor the progress of the job, you can open the Monitor panel (Applications → Monitor Jobs, or click the Jobs button in the status bar). This panel is automatically updated to show the progress of your job. As each separate program in the Jaguar code finishes running, its completion is noted in the log text area. When the program `scf` is running, the Monitor panel displays the energy and other data of each iteration. See Section 6.4 on page 164 on the log file for more information on this data. If you want the Monitor panel to open automatically when you start a job, you can set a preference in the Preferences panel (Maestro → Preferences) under Jobs → Monitoring.

When the job finishes, its output file is copied to the directory from which you started Maestro. The output file ends with the extension `.out`. For instance, if you entered the job name `h2o`, the output file would be `h2o.out`.

If you want to exit Maestro, choose Quit from the Maestro menu in the Maestro main window. The Quit dialog box permits you to save a log file of the Maestro session. A warning dialog box is displayed, which permits you to save the Maestro scratch project. For this exercise, choose Discard.

To check that the job ran correctly, change to the directory where the output file was stored and enter the following command:

```
diff -w jobname.out $SCHRODINGER/jaguar-vversion/samples/H2O.out
```

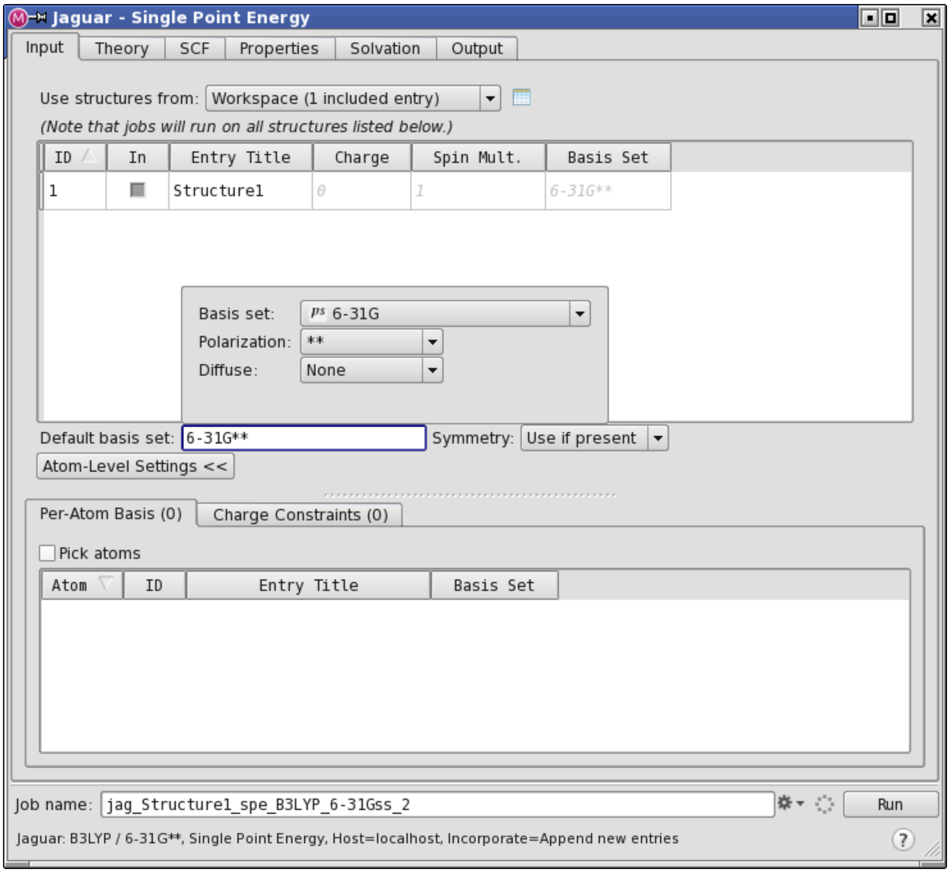
If there is no output from this command, the job ran correctly. If there is output, examine the differences between the two files to see if the differences are significant.

If you are satisfied with the results of this sample run, continue this chapter to learn more about using Maestro. If you were unable to run the sample calculation, see the troubleshooting suggestions in Chapter 12.

2.2 The Jaguar Panel

The Jaguar panel is the main interface between Maestro and Jaguar. In this panel, you can set up input files for a range of Jaguar jobs, and start the jobs. To open the panel, choose the task or calculation type from the Jaguar submenu of the Applications menu in the main window. The available tasks are:

- Single Point Energy
- Optimization
- Relaxed Coordinate Scan
- Rigid Coordinate Scan
- Transition State Search
- Reaction Coordinate
- Initial Guess Only



Use structures from: Workspace (1 included entry)

(Note that jobs will run on all structures listed below.)

ID	In	Entry Title	Charge	Spin Mult.	Basis Set
1	■	Structure1	0	1	6-31G**

Basis set: 6-31G

Polarization: **

Diffuse: None

Default basis set: 6-31G** Symmetry: Use if present

☐ Pick atoms

Atom	ID	Entry Title	Basis Set

Job name: jag_Structure1_spe_B3LYP_6-31Gss_2

Jaguar: B3LYP / 6-31G**, Single Point Energy, Host=localhost, Incorporate=Append new entries

Figure 2.1. The Jaguar panel.

Below these tasks in the menu are several calculation types that are run as Jaguar batch jobs:

- pKa
- Hydrogen Bond
- Counterpoise
- Fukui Functions
- VCD Spectra
- Heat of Formation
- Reaction

The input for these calculations is described in Chapter 5 and Chapter 13.

Most of the Jaguar panel is occupied by a set of tabs in which you can make settings for jobs. These tabs are described in Chapter 3 and Chapter 4.

You can select multiple structures as input for many kinds of Jaguar calculations. When you do, Maestro creates a Jaguar batch script that runs the calculation for each structure independently. These independent calculations can be run across multiple processors. For more information, see Section 2.9 on page 25.

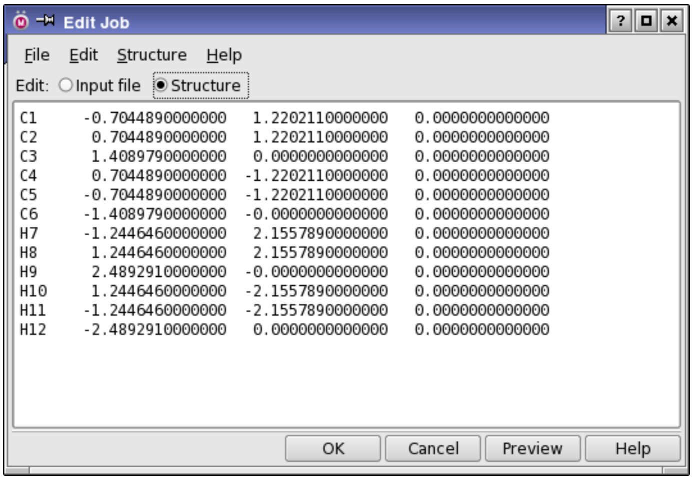
Below the tabs is the Job toolbar, which is described in Section 1.3 on page 4. Some of the items on the Settings button menu are described in later sections of this chapter.

2.3 The Edit Job Dialog Box

While most of the common settings for Jaguar jobs can be made in the Jaguar panel, you might need to make changes to the settings, add keywords to the input file for options that are not available from the Jaguar panel, or make changes to the geometry. You can make these changes in the Edit Job dialog box, which you open by choosing Edit from the Settings button menu on the Job toolbar. Apart from standard editing tools, this dialog box has special tools for editing Jaguar input files.

The basic editing tools are contained in the File and Edit menus. The File menu allows you to save the current state of the input file to disk (Write), and to cancel all your changes without closing the dialog box (Revert). The Edit menu provides Cut, Copy, and Paste options for cutting and pasting, which you can use to copy text from elsewhere in Maestro or from another application. You can also use the usual keystrokes (CTRL+X, CTRL+C, CTRL+V) to copy and paste text. The text in the paste buffer is saved when you close the dialog box, so you can copy text between input files. You can copy settings from a previous job that was incorporated into Maestro, by clicking the link to the input file in the Project Table and choosing View to open it in a text viewer, then selecting the text and copying it with CTRL+C. (See Section 2.11 on page 29 for more information on the links.)

The Edit Job dialog box has two editing modes: Input File and Structure. By default, the entire input file is available for editing. In Structure mode, only the geometry is displayed, and a Structure menu is added to the menu bar. If you are editing geometries for a transition state search or an IRC scan, all three geometries can be edited. The editing area has tabs for each of these geometries, labeled Reactant, Product, and Transition State.



```
C1     -0.7044890000000   1.2202110000000   0.0000000000000
C2      0.7044890000000   1.2202110000000   0.0000000000000
C3      1.4089790000000   0.0000000000000   0.0000000000000
C4      0.7044890000000  -1.2202110000000   0.0000000000000
C5     -0.7044890000000  -1.2202110000000   0.0000000000000
C6     -1.4089790000000  -0.0000000000000   0.0000000000000
H7     -1.2446460000000   2.1557890000000   0.0000000000000
H8      1.2446460000000   2.1557890000000   0.0000000000000
H9      2.4892910000000  -0.0000000000000   0.0000000000000
H10     1.2446460000000  -2.1557890000000   0.0000000000000
H11    -1.2446460000000  -2.1557890000000   0.0000000000000
H12    -2.4892910000000   0.0000000000000   0.0000000000000
```

Figure 2.2. The Edit Job dialog box with Structure selected.

The Structure menu provides options for modifying the geometry input. The Convert to Z-matrix and Convert to Cartesians options switch between Z-matrix format and Cartesian format. The option Assign Unique Atom Labels converts all atom labels to the form *El#*, where *El* is the standard element symbol (Fe for iron, for instance) and # is the atom number in the input list (1 for the first atom, 2 for the second, and so on). This option guarantees that all atoms have unique atom labels, which is required by Jaguar. Unique atom labels are assigned automatically if Jaguar detects any ambiguity in the labels. You can display the atom labels in the Workspace by choosing View Atom Labels. The remaining option on the Structure menu is useful for transition state searches, but not for other Jaguar jobs. This option is described in Section 4.3 on page 100.

The changes you make are automatically saved in the Maestro project, and are reflected in the panel settings. You can view the changes to the geometry in the Workspace by clicking Preview.

Note: Counterpoise atoms, constraints and coordinates for Hessian refinement should not be added in this dialog box: they are removed when you close the dialog box.

2.4 Molecular Structure Input

After you start Maestro, the first task for setting up any Jaguar calculation is to enter a molecular structure (geometry).[1] You can create a structure using Maestro's Build panel, you can use the Jaguar panel to read in a file as described in Section 2.5 on page 20, or you can enter and edit the geometry yourself, in Cartesian (*x*, *y*, *z*) coordinates or in Z-matrix format, using the Edit Job dialog box. This section describes the input formats for Cartesian and Z-matrix geometries. The geometry input is used to set constraints of bond lengths or angles for geometry optimization and to specify atoms for a counterpoise calculation. These aspects of geometry input are explained in this section as well.

The geometries that you enter are displayed in the Workspace, in which you can rotate and translate the structure, change the geometry, display in various representations, and perform many other tasks. For information on Maestro, see the *Maestro User Manual*.

2.4.1 Cartesian Format for Geometry Input

The Cartesian geometry input format can consist of a simple list of atom labels and the atomic coordinates in angstroms in Cartesian (*x*, *y*, *z*) form. For example, the input

```
O   0.000000    0.000000   -0.113502
H1  0.000000    0.753108    0.454006
H2  0.000000   -0.753108    0.454006
```

describes a water molecule. Each atomic label must start with the one- or two-letter element symbol, and may be followed by additional characters, as long as the atomic label has eight or fewer characters and the atomic symbol remains clear. For example, `HE5` would be interpreted as helium atom 5, not hydrogen atom E5. The atom label is case-insensitive. The coordinates may be specified in any valid C format, but each line of the geometry input should contain no more than 80 characters.

2.4.2 Variables in Cartesian Input

Coordinates can also be specified as variables, whose values are set below the list of atomic coordinates. This makes it easier to enter equal values and also makes it possible to keep several atoms within the same plane during a geometry optimization.

To use variables, type the variable name (`zcoor`, for instance) where you would normally type the corresponding numerical value for each relevant coordinate. You can prefix any variable with a + or – sign. When you have entered the full geometry, add one or more lines setting the variables. For instance, the Cartesian input

1. The geometry input is in the **zmat** and **zvar** sections of the input file.

```
O   0.000000     0.000000  -0.113502
H1  0.000000     ycoor      zcoor
H2  0.000000    -ycoor      zcoor
ycoor=0.753108   zcoor=0.454006
```

describes the same water coordinates as the previous Cartesian input example. If you performed a geometry optimization using this input structure, its `ycoor` and `zcoor` values might change, but their values for one hydrogen atom would always be the same as those for the other hydrogen atom, so the molecule would retain C_{2v} symmetry.

The variable settings can also be separated from the coordinates by a line containing the text `Z-variables`. For instance, the following input is equivalent to the previous example:

```
O   0.000000     0.000000  -0.113502
H1  0.000000     ycoor      zcoor
H2  0.000000    -ycoor      zcoor
Z-variables
ycoor=0.753108
zcoor=0.454006
```

Note that if Cartesian input with variables is used for an optimization, Jaguar performs the optimization using Cartesian coordinates instead of generating redundant internal coordinates, and the optimization will not make use of molecular symmetry.

2.4.3 Constraining Cartesian Coordinates

As described in the previous section, you can force Cartesian coordinates to remain the same as each other during an optimization by using variables. You can also specify Cartesian coordinates that should be frozen during a geometry optimization by adding a "#" sign after the coordinate values. For example, if you add constraints to the `zcoor` variables in the water input example, as listed below,

```
O   0.000000     0.000000  -0.113502
H1  0.000000     ycoor      zcoor#
H2  0.000000    -ycoor      zcoor#
ycoor=0.753108   zcoor=0.454006
```

and perform a geometry optimization on this molecule, the H atoms would be allowed to move only within the xy plane in which they started.

If frozen Cartesian coordinates are included in the input for an optimization, Jaguar uses Cartesian coordinates for the optimization rather than generating redundant internal coordinates, and the optimization does not make use of molecular symmetry.

2.4.4 Z-Matrix Format for Geometry Input

Like Cartesian geometries, Z-matrix-format geometries also specify atoms by atom labels that begin with the one- or two-letter element symbol. The atom label is case-insensitive. The element symbol may be followed by additional characters, as long as the atom label has eight or fewer characters and the element symbol is still clear.

The first line of the Z-matrix should contain only one item: the atom label for the first atom. For example,

```
N1
```

This atom is placed at the origin. The second line contains the atom label for atom 2, the identifier of atom 1, and the distance between atoms 1 and 2. Identifiers can either be atom labels or atom numbers (the position in the list: 1 for the first atom, 5 for the fifth atom listed, and so on). In this example, the identifier for the first atom could be either “N1” or “1.” The second atom is placed along the positive *z*-axis. For example,

```
N1
C2   N1  1.4589
```

places the carbon atom (C2) at (0.0, 0.0, 1.4589) in Cartesian coordinates. Distances between atoms must be positive.

The third line is made up of five items: the atom label for atom 3, the identifier of one of the previous atoms, the distance between this atom and atom 3, the identifier of the other previous atom, and the angle defined by the three atoms. In this example,

```
N1
C2   N1  1.4589
C3   C2  1.5203   N1  115.32
```

the final line states that atoms C3 and C2 are separated by 1.5203 Å and that the C3–C2–N1 bond angle is 115.32°. The bond angle must be between 0° and 180°, inclusive. The third atom (C3 in this case) is placed in the *xz* plane (positive *x*).

The fourth line contains seven items: the atom label for atom 4, an atom identifier, the distance between this atom and atom 4, a second atom identifier, the angle defined by these three atoms, a third atom identifier, and a torsional angle. In this example,

```
N1
C2   N1  1.4589
C3   C2  1.5203   N1  115.32
O4   C3  1.2036   C2  126.28   N1  150.0
```

the last line states that atoms O4 and C3 are 1.2036 units apart, that the O4–C3–C2 bond angle is 126.28°, and that the torsional angle defined by O4–C3–C2–N1 is 150.0°. This information is sufficient to uniquely determine a position for O4. If the first three atoms in the torsional angle definition were collinear or very nearly collinear, O4's position would be poorly defined. You should avoid defining torsional angles relative to three collinear (or nearly collinear) angles. In such a case you should use dummy atoms to define the torsional angle (see Section 2.4.5 on page 16).

The torsional angle is the angle between the plane formed by the first three atoms (in this case N1–C2–C3) and the plane formed by the last three atoms (in this case C2–C3–O4). Looking from the second to the third atom (C2 to C3), the sign of the angle is positive if the angle is traced in a clockwise direction from the first plane to the second plane, and negative if the angle is traced counterclockwise.

An alternative for specifying the fourth atom's position is to use a second bond angle instead of a torsional angle. To specify another bond angle, add 1 or –1 to the end of the line. The second bond angle is the angle between the first, second, and fourth atoms (in the example above, the O4–C3–N1 angle). Since there are two possible positions for the atom which meet the angle specifications, the position is defined by the scalar triple product $\mathbf{r}_{12} \cdot (\mathbf{r}_{23} \times \mathbf{r}_{24})$, where $\mathbf{r}_{ij} = \mathbf{r}_i - \mathbf{r}_j$ is the vector pointing from atom j to atom i. If this product is positive, the value at the end of the line should be 1. If it is negative, the value should be 1. You should use torsional angles instead of second bond angles if you want to perform a constrained geometry optimization, however, since Jaguar cannot interpret *any* constraints on bond lengths or angles for geometries containing second bond angles.

All additional lines of the Z-matrix should have the same form as the fourth line. The complete Z-matrix for the example molecule (the 150° conformation of glycine) is

```
N1
C2   N1  1.4589
C3   C2  1.5203   N1  115.32
O4   C3  1.2036   C2  126.28   N1  150.0
O5   C3  1.3669   C2  111.39   N1  -31.8
H6   N1  1.0008   C2  113.55   C3  -69.7
H7   N1  1.0004   C2  112.77   C3   57.9
H8   C2  1.0833   N1  108.89   H6  170.0
H9   C2  1.0782   N1  110.41   H6   52.3
H10  O5  0.9656   C3  111.63   C2 -178.2
```

2.4.5 Variables and Dummy Atoms in Z-Matrix Input

Bond lengths or angles can also be specified as variables below the Z-matrix itself. This feature makes it easier to input equal values (such as C–H bond lengths or H–C–H bond angles for methane), and also makes it possible to keep several distances or angles the same as each other during an optimization.

To use variables, type the variable name (chbond, for instance) where you would type the corresponding value (such as a C–H bond length in Å) for each relevant occurrence of that value. You can prefix any variable with a + or – sign. After you type the full Z-matrix, define the variables by adding one or more lines at the bottom, such as

```
chbond=1.09  HCHang=109.47
```

As for Cartesian input, you can separate the variable settings from the coordinates by a line containing the text Z-variables.

Defining dummy atoms can make the assignment of bond lengths and angles easier. Dummy atoms are a way of describing a point in space in the format used for an atomic coordinate without placing an atom at that point. The symbols allowed for dummy atoms are X or Du. An example of the use of a dummy atom for CH_3OH input follows:

```
C
O    C   1.421
H1   C   1.094    O   107.2
X1   C   1.000    O   129.9    H1  180.0
H2   C   1.094    X1   54.25   H1   90.0
H3   C   1.094    X1   54.25   H1  -90.0
H4   O   0.963    C   108.0    H1  180.0
```

2.4.6 Constraining Z-Matrix Bond Lengths or Angles

To freeze bond lengths or angles during a geometry optimization, add a # sign after the coordinate values. For example, to fix the HOH bond angle of water to be 106.0°, you could enter the following Z-matrix:

```
O
H1  O   0.9428
H1  O   0.9428   H1   106.0#
```

In a geometry optimization on this input geometry, the bond angle remains frozen at 106° throughout the optimization, although the bond lengths would vary. For more details, see Section 4.2 on page 96, which describes how to set up constraints for optimizations.

To constrain two geometric parameters to be the same during a geometry optimization, use variables in Z-matrix input (see Section 2.4.5 on page 16). To freeze variables during an optimization, add a # sign to the end of the variable setting in the variable definition section. In this example, the C–H bond is frozen at 1.09 Å:

```
chbond=1.09#  HCHang=109.47
```

2.4.7 Counterpoise Calculations

Following the procedure of Boys and Bernardi [29], a counterpoise calculation is often used to correct for the problem of basis set superposition error (BSSE), which arises when an incomplete basis set is used in the calculation of the binding energy of a complex consisting of two or more molecules. The calculation of a counterpoise-corrected binding energy for a dimeric complex actually consists of seven calculations:

- Geometry optimization of the complex (calculation 1)
- Geometry optimization of each of the two molecular fragments in their own basis sets (calculations 2, 3)
- Single-point calculations of each of the fragments in their own basis sets at the geometries that they adopt in the complex (calculations 4, 5)
- Single-point counterpoise calculations on each fragment at the geometries that they adopt in the complex using the basis set of the complex (calculations 6, 7)

The usual, uncorrected binding energy would be calculated as:

$$\Delta E_{\text{bind}} = E_1 - (E_2 + E_3)$$

where the energy subscripts refer to the calculations listed above. The counterpoise correction to the binding energy expresses the artificial gain in energy of each molecular fragment when it can use the basis functions of the other fragment in addition to its own basis functions:

$$\Delta E_{\text{cp}} = (E_4 - E_6) + (E_5 - E_7)$$

Calculated in this way, the counterpoise correction is a positive number, and it is added to ΔE_{bind} to yield the final binding energy. Counterpoise corrections are often several kilocalories per mole in magnitude, and decrease as the size of the basis set increases.

In the input files for jobs 6 and 7, the atoms of one fragment must be marked as counterpoise atoms (also called “ghost atoms”) so that only their basis functions are used. In Jaguar, a counterpoise atom is indicated by appending a `@` to the atom label. For example, to calculate the interaction energy of a water molecule with a methanol molecule, the **zmat** section for one counterpoise job would have the atoms of methanol marked as counterpoise atoms:

```
&zmat
 O1@        -0.3380316687          0.9068671477          0.0000000000
 H2@        -0.3206434752         -0.0520359937          0.0000000000
 C3@         0.9752459717          1.3666159794          0.0000000000
 H4@         0.9478196867          2.4513855069          0.0000000000
 H5@         1.5357440779          1.0497731323         -0.8817844743
 H6@         1.5357440779         1.0497731323         0.8817844743
 O7         -0.4959747210        -1.9447535985         0.0000000000
 H8         -1.0372322234        -2.1494734847         0.7574958845
 H9         -1.0372322234        -2.1494734847        -0.7574958845
&
```

In the other counterpoise job, the **zmat** section would have the atoms of the water molecule marked as counterpoise atoms:

```
&zmat
 O1         -0.3380316687         0.9068671477         0.0000000000
 H2         -0.3206434752        -0.0520359937         0.0000000000
 C3          0.9752459717         1.3666159794         0.0000000000
 H4          0.9478196867         2.4513855069         0.0000000000
 H5          1.5357440779         1.0497731323        -0.8817844743
 H6          1.5357440779         1.0497731323         0.8817844743
 O7@        -0.4959747210        -1.9447535985         0.0000000000
 H8@        -1.0372322234        -2.1494734847         0.7574958845
 H9@        -1.0372322234        -2.1494734847        -0.7574958845
&
```

You can also indicate counterpoise atoms in an **atomic** section by setting their nuclear charge to zero in the 'charge' column (see Section on page 275).

To automate the calculation of a counterpoise-corrected binding energy for a complex consisting of two non-covalently bound molecules, you can choose Counterpoise from the Jaguar submenu of the Applications menu. The panel that opens can be used to set up and run the counterpoise job, which uses the Jaguar batch script `counterpoise.py`. This panel contains only the Input, Theory, SCF, and Optimization tabs. See Section 11.3 on page 329 for details on the `counterpoise.py` script, which you can also use from the command line. Counterpoise calculations can be distributed across multiple processors.

For LMP2 calculations (see Section 3.5 on page 50), the LMP2 correction is already designed to avoid basis set superposition error, so we advise computing only the SCF counterpoise correction term.

2.4.8 Specifying Coordinates for Hessian Refinement

If you are optimizing a molecular structure to obtain a transition state, you might want to refine the Hessian used for the job. Section 4.3 on page 100 explains the methods used for transition-state optimizations, including Hessian refinement. This subsection explains only how to edit your input to specify particular coordinates for Hessian refinement. (Whether or not you refine particular coordinates, you can specify a certain number of the lowest eigenvectors of the Hessian for refinement, as described in Section 4.3.5 on page 104—the Hessian can be refined in both ways in the same job.)

If you type an asterisk (*) after a coordinate value, Jaguar computes the gradient of the energy both at the original geometry and at a geometry for which the asterisk-marked coordinate has been changed slightly, and uses the results to refine the initial Hessian to be used for the optimization. To request refinement of a coordinate whose value is set using a variable, add an asterisk to the end of the variable setting in the variable definition section.

For instance, either of the following two input geometries will have the same effect: a job that includes Hessian refinement will use both O–H bonds and the H–O–H angle in the refinement.

```
O1
H2   O1   1.1*
H3   O1   1.1*   H2   108.0*
```

or

```
O1
H2   O1   ohbond
H3   O1   ohbond   H2   108.0*
ohbond = 1.1*
```

Molecular symmetry or the use of variables, either of which may constrain several coordinate values to be equal to each other, can reduce the number of coordinates actually used for refinement. For example, for the second water example shown above, only two coordinates are actually refined (the O–H bond distance, which is the same for both bonds, and the H–O–H angle). The same would be true for the first example if molecular symmetry were used for the job.

2.5 Reading Files

If you already have Jaguar input files containing geometries (either with or without information on the type of calculation to perform), you can read them using the Jaguar Read dialog box, which you open by choosing Read from the Settings button menu on the Job toolbar. This dialog box is a file selector with the usual file browsing tools: a Filter text box, a Directories list, a Files list, and a Selection text box. By default, information is displayed for the current working directory.

When you read a Jaguar input file, you can read the geometry only, or you can read the entire input file. To read just the geometry, choose Geometry only from the Read as option menu. To read the entire input file, choose Geometry and settings from the Read as option menu. If you read in a geometry only from a file, Jaguar also tries to obtain information on the molecular charge. If you read the geometry and settings, the settings are used to determine the Jaguar task, which might not be the task with which you opened the Jaguar panel. For example, if you chose Single Point Energy, then read an input file for a geometry optimization including the settings, the task is reset to Optimization.

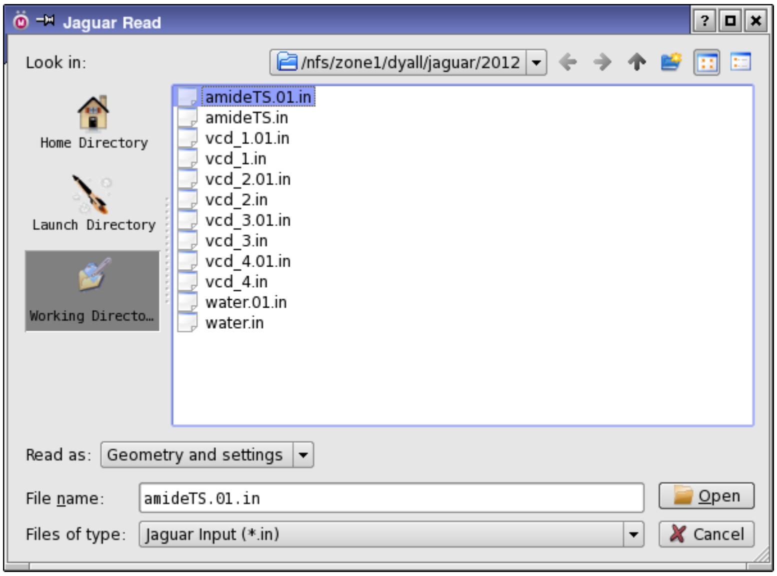


Figure 2.3. The Jaguar Read dialog box.

The structures in the input file are added as entries in the Project Table, named with the stem of the input file name by default. For example, reading `h2o.in` creates an entry named `h2o`.

To read geometries from files generated or used by other programs, you must import them into Maestro using the Import panel. The files are imported using the file format conversion program, Babel [26], and must be in a format recognized by Babel. Maestro does not read any information other than the geometry from these files. If you want other information, such as a Hessian, you can cut and paste it into a Jaguar input file.

2.6 Setting Charge and Multiplicity

Apart from the geometry, the main setting that you might want to make in an otherwise default calculation is to set the molecular charge and the spin multiplicity. You can set these quantities in the Input tab. The default molecular charge is determined by the formal charges on the atoms in the Workspace. The default spin multiplicity is set to 1 (singlet). You can change the charge by entering a value in the Charge column,[2] and you can change the spin multiplicity by entering a value in the Spin Mult. column.[3] If the molecular charge and spin multiplicity settings you make do not agree with your molecular input—for instance, if your molecule has

2. Keyword **molchg** in the **gen** section.
3. Keyword **multip** in the **gen** section.

an odd number of electrons and you set the spin multiplicity to 1—Maestro warns you of the inconsistency by default, and you must choose consistent values to submit a job.

The charge and multiplicity are added as properties to the output Maestro file and are added to the Project Table when the job is incorporated.

2.7 Cleaning up Molecular Geometries

The molecular geometry sometimes needs improvement before you perform calculations. For example, it might not have the desired molecular symmetry, or it might be far from the minimum (or transition state). Maestro has options to clean up the geometry for calculations in both of these cases. The options are available from the Edit menu.

2.7.1 Quick Geometry Optimization

You can clean up the geometry by clicking the Clean up geometry button on the Build toolbar:

By default, the minimization is performed with the OPLS_2005 force field, which is fast and is suitable for most organic molecules. However, this force field does not cover all atoms. For molecules that contain other atoms, you can instead choose to do cleanup based on Goddard and Rappe's Universal Force Field (UFF), by choosing the UFF option under Geometry cleanup method in the Builder tab of the Preferences panel. Because UFF includes parameters for all elements in the periodic table, it can be used for inorganic complexes as well as organic compounds. During the UFF minimization, a status box is displayed. To stop the minimization, click Stop in this status box.

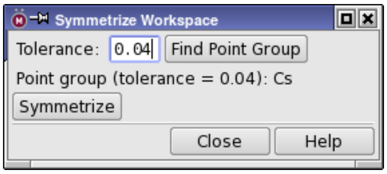
The convergence criteria for the cleanup minimization are deliberately set fairly loose, so that even fairly large systems can be optimized interactively. In addition, a time limit is imposed on the minimization to keep it from running excessively long. As a result, you might find that the geometry continues to change if you perform a second cleanup on a cleaned-up structure.

UFF cleanup minimization is useful for quickly bringing a distorted molecule back into the neighborhood of the ab initio minimum-energy geometry, in preparation for full ab initio geometry optimization. However, it is no substitute for ab initio optimization because UFF is a relatively simple force field. It is probably a good idea to perform a cleanup minimization after building a new molecule. On the other hand, performing a cleanup minimization on a molecule that has already undergone ab initio minimization is likely to move the molecule away from the ab initio minimum. Also, you should be careful to avoid cleaning up a structure that has been prepared as an initial guess for a transition-state search.

2.7.2 Symmetrization

By default, Jaguar takes advantage of molecular symmetry[4] whenever possible, in order to save CPU time. Both Abelian and non-Abelian point groups are recognized. Generally, you should symmetrize the geometry if you plan to use symmetry in the calculation itself. Otherwise, the input coordinates may not be accurate enough for the desired symmetry to be recognized.

You can symmetrize the molecule by choosing Symmetrize Workspace from the Edit menu. The Symmetrize Workspace dialog box opens.



Point group (tolerance = 0.04): Cs

Figure 2.4. The Symmetrize Workspace dialog box.

The point group symmetry is determined as follows. After the molecule is translated so that the center of mass is at the origin of the coordinate system and rotated so that the principal axes of inertia are aligned on the coordinate axes, symmetry operations (reflections, rotations, and inversions) are applied to determine the point group of the molecule.

When Maestro checks whether a symmetry operation produces an equivalent structure, the coordinates of the two structures only have to be the same to within a prescribed tolerance, that is, each pair of symmetry-related atoms is within a distance specified by the tolerance. The value of the tolerance can be specified in the Tolerance text box, and is 0.04 Å by default. This value ensures that the highest symmetry is found in most cases. By changing the value and clicking the Find Point Group button, you can determine whether there is a lower (or higher) symmetry point group that approximately describes the structure, and use that group to symmetrize the molecule instead of the default.

The tolerance is also used when the molecule is symmetrized. After translation and rotation, the coordinates of the atoms are adjusted to reflect the symmetry group accurately. The maximum displacement permitted is the tolerance specified. A large tolerance yields the highest symmetry, but may cause the coordinates to be changed significantly. A small tolerance may yield a lower symmetry, but results in smaller coordinate changes. The main Jaguar programs use a small tolerance (1.0×10^{-6} bohr), which should result in molecular energy changes of 1 microHartree or less.

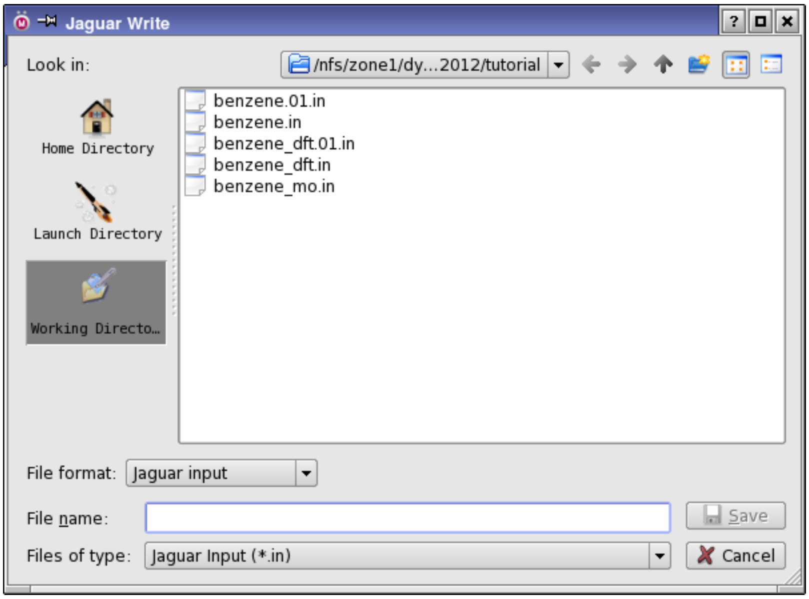
4. Keyword **isymm** = 8 in the **gen** section.

You can turn the use of symmetry off[5] in the Molecule tab. For methods such as LMP2 and for some properties, such as IR intensities or hyperpolarizabilities, symmetry is not implemented and is disabled automatically for the job.

If you are comparing calculations from geometries that differ only slightly, you must use caution when symmetrizing coordinates. For example, a small symmetry-breaking change can be removed if its magnitude is smaller than the tolerance you have set, which establishes what changes are acceptable. In this case, you should inspect the symmetrized coordinates in the Edit Job dialog box to insure that symmetrizing had the desired effect and did not discard any important information about the molecular geometry.

2.8 Writing Files

When you are satisfied with the molecular geometry and the settings that you have made, you can either run the job, or save the geometry and settings for later use. Running jobs is described in the next section. To save your input, choose Write from the Settings button on the Job toolbar. The Jaguar Write dialog box is displayed. This dialog box is a file selector with the usual file browsing tools.



benzene.01.in
benzene.in
benzene_dft.01.in
benzene_dft.in
benzene_mo.in

File format: Jaguar input
File name:
Files of type: Jaguar Input (*.in)

Figure 2.5. The Jaguar Write dialog box.

5. Keyword **isymm** = 0 in the **gen** section.

If you want to write a Jaguar input file containing the geometry and the settings, choose Jaguar input from the File Format option menu. When you click OK, a Jaguar input file is created that is suitable for running a calculation, whether from Maestro or from the command line.

If you want to use the settings for a series of calculations on different molecules, you can write a batch script, by choosing Jaguar batch script from the File Format option menu. When you click OK, a Jaguar batch script is created that contains settings (**gen** section) and batch commands, but no geometry, so that you can run this script with any geometry. To make use of this script, you can select Run Batch File from the Jaguar submenu of the Applications menu in the main window.

You can also use Maestro to export structures to a variety of formats, using the Babel file conversion program (see Section 11.1.2 on page 318). See Chapter 3 of the *Maestro User Manual* for more information on exporting structures.

2.9 Running Jobs

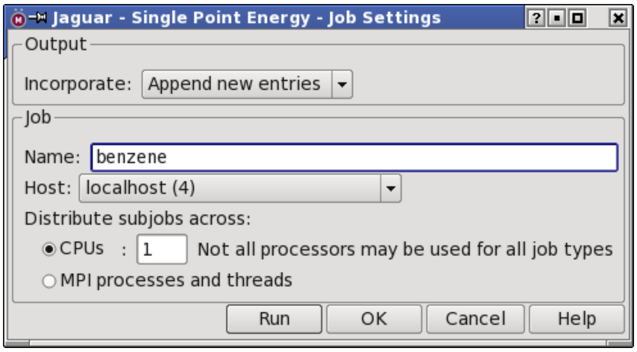
Maestro provides several ways of running Jaguar jobs. You can select a task from the Jaguar submenu of the Applications menu, make settings, then start the job; you can select a set of preexisting input files and run them as a single job by selecting Run Input Files from the Jaguar submenu; or you can run a job using a preexisting Jaguar batch file with one or more structures as input by selecting Run Batch File from the Jaguar submenu.

Whenever you run a calculation with multiple structures as input that does not have any geometry-dependent settings, a Jaguar batch script is created to run the calculation on each structure.

You can submit a job either from Maestro or from the command line. Information on submitting jobs from the command line with the `jaguar run` command can be found in Section 11.1 on page 315. This section describes the submission of jobs from Maestro.

You can submit the Jaguar job by clicking the Run button in the relevant panel. The job is run with the current job settings. To change the job settings, you can use the Settings button to open the Job Settings dialog box. See Section 1.3 on page 4 for more information. For more detailed information on this dialog box, see Section 2.2 of the *Job Control Guide*.

If you want to run a parallel job, you must select a multiprocessor host, and specify the number of CPUs to use. Batch jobs with multiple input structures can be distributed over multiple CPUs, so you can select a multiprocessor host and specify the number of CPUs to use for these jobs also. For jobs that can be parallelized over MPI processes and OMP threads, you can specify the number of both, and the number of CPUs is the product of these two. See Section 11.4 on page 335 for more information about parallel jobs.


Jaguar - Single Point Energy - Job Settings

Output

Incorporate: Append new entries

Job

Name: benzene

Host: localhost (4)

Distribute subjobs across:

[✓ CPUs] : 1 Not all processors may be used for all job types

MPI processes and threads

Figure 2.6. The Job Settings dialog box.

If you want to monitor the progress of a job, you can do so in the Monitor panel, which you open from the Applications menu. See Chapter 3 of the *Job Control Guide* for more information on job control and monitoring.

2.10 Running Jaguar Batch Jobs

You can run multiple Jaguar calculations in a single run using Jaguar batch scripts. Some of the kinds of calculations you can run with a batch script are:

- Multiple independent jobs with predetermined input files
- The same type of job for several input geometries
- A series of jobs in which later jobs use files generated during earlier jobs

Several batch scripts are included with Jaguar. You can write your own batch scripts, or save job settings as a batch script in the Jaguar Write dialog box. Maestro writes temporary batch scripts and runs a batch job whenever you run jobs with multiple structures as input that do not have geometry-dependent settings. Section 11.2 on page 322 provides details on batch scripts.

Jaguar batch jobs can be run from the Run Batch File panel, which you open by choosing Applications → Jaguar → Run Batch File. To run a batch job, you must select a script and a source of structures, and start the job. If you select Jaguar input files for the structural input, you can choose whether to use or ignore the settings in the input files.

To select a script, you can enter the path in the Script text box, or click Browse and choose a script in the Select Batch Script panel. This panel is a file selector with the usual file browsing tools. By default, information is displayed in the lists and the filter for the current working directory. To select one of the supplied Jaguar scripts, click Jaguar Scripts in the Directory section, then select the script. When you select a script, the Notes text area shows comments from the batch script.

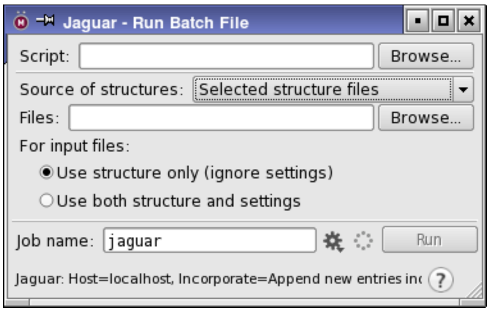

Jaguar - Run Batch File
Script:
Browse...
Source of structures: Selected structure files
Files:
Browse...
For input files:
Use structure only (ignore settings)
Use both structure and settings
Job name: jaguar
Run
Jaguar: Host=localhost, Incorporate=Append new entries in

Figure 2.7. The Run Batch File panel.

When you have chosen the directory, choose a script from the list of scripts, then click OK. The built-in scripts are described briefly in Table 2.1.

The input files that you select can be pre-existing input files, or files created from the current structure in the Workspace or from Project Table entries. These choices are available from the Source of structures option menu in the Run Batch File panel. If you choose Workspace or Selected entries, the structures are written to a single Maestro file, and the Jaguar input files are created later by the batch facility.

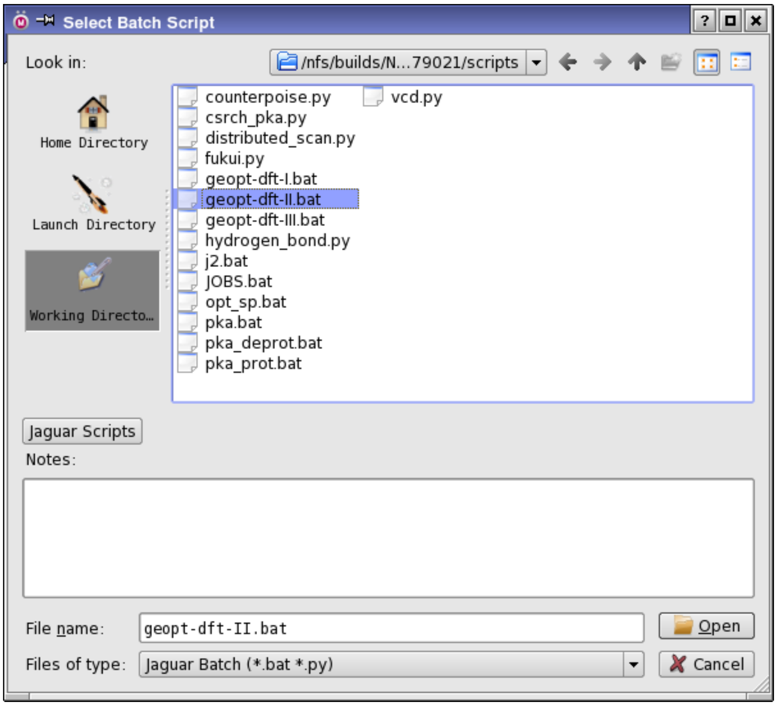


Figure 2.8. The Select Batch Script panel.

Table 2.1. Description of built-in batch scripts

Script	Description
`JOBS.bat`	Run a sequence of jobs specified by the input files.
`geopt-DFT-I.bat`	Preoptimize a geometry at the BLYP/6-31G level, then optimize at the BLYP/6-31G* level.
`geopt-DFT-II.bat`	Do geometry preoptimizations at the HF/6-31G and BLYP/6-31G* level, then optimize at the B3LYP/cc-pVTZ(-f) level.
`geopt-DFT-III.bat`	Do geometry preoptimizations at the HF/6-31G, BLYP/6-31G* and B3LYP/6-31G* level, then optimize at the B3LYP/cc-pVTZ(-f) level.
`pka.bat`	Run a pKa calculation. See Chapter 13 for details.
`canonical.py`	Run a calculation on a set of isomers, enforcing the same number of canonical orbitals. See Section 3.7.5 and Section 11.3.7.
`counterpoise.py`	Run a counterpoise calculation on a dimer. See Section 2.4.7 and Section 11.3.1.
`distributed_scan.py`	Distribute a 1D or 2D scan job over multiple processors. Automatically used from Maestro when multiple processors are requested. See Section 11.3.4.
`hydrogen_bond.py`	Calculate binding energy of two molecules that are hydrogen-bonded. See Section 5.1 and Section 11.3.2.
`jreactions.py`	Calculate the reaction energy and enthalpy for specified reactants and products. See Section 5.5 and Section 11.3.8.
`fukui.py`	Calculate Fukui functions. See Section 5.2 and Section 11.3.5.
`vcd.py`	Compute vibrational circular dichroism (VCD) properties. See Section 5.3 and Section 11.3.6.

To select pre-existing files, choose Selected structure files from the Source of structures option menu, then enter a comma-separated list of file names in the Files text box, or click Browse. The Browse button opens a standard file selector, labeled Select Batch Inputs, with the current directory and its files listed. You can select either Jaguar input files or Maestro files for input. If you select Maestro files, Jaguar input files are constructed later. To select multiple files, use SHIFT to select a range of items and CTRL to select or deselect a single item without affecting other items. When you have made a selection, click OK. The input files are passed to the batch script in the order in which they appear in the list. To process input files in a particular order, you must name them so that they appear in the correct order in the list.

After you finish selecting the batch script and input files, click Start to set the job parameters and launch the batch job. You can distribute the job over multiple processors by selecting a multiprocessor host and specifying the number of CPUs. When the job is submitted, the

Monitor panel opens (depending on the preferences you set for job monitoring—see Section 14.13 of the *Maestro User Manual*). This panel shows the batch log file (`.blog`) for the batch job, which logs the completion of each Jaguar job launched from the batch script. The information is automatically updated as the Jaguar jobs run.

2.11 Output

A Jaguar log file contains comments on the progress of a job. If the job was started from Maestro, the log file is written to the local job directory. The log file notes when each section of Jaguar is complete, as well as noting data from each iteration in an SCF calculation as it is calculated. You can view this file in the Monitor panel, which you can open by choosing Monitor Jobs from the Applications menu or the Tasks menu in the main window. See Section 6.4 on page 164 for more information on this file.

The primary Jaguar text output is contained in the output file, which is created in the scratch directory of the host on which the calculation is run, and is copied back to the local host when the job finishes. The output file is described in Chapter 6.

The structure output from a Jaguar job is written to a Maestro file, along with any properties that are calculated. These properties are shown in the Project Table when the job is incorporated into the Maestro project.

You can examine the text output files and the input files for any completed Jaguar job that has been incorporated into the project. The names of the files are recorded as Maestro properties, and are shown in blue with an underline, like a hyperlink in a web page, to indicate that they are links to the files. It is likely that you will need to choose Property → Show → All to see these properties, as they are hidden by default. Click on the link, and choose View to open the file in a read-only text viewer (the View File panel).

You can select text and copy it (with CTRL+C, ⌘C) if you want to use it somewhere else, such as in the input for another job. This feature allows you to examine the settings used for a particular job, and reproduce them in another job by pasting them into the Edit Job panel, all from within Maestro.

You can search for text using the search box next to the file name. The search takes place as you type. To step through the hits, use the arrow buttons to the right of the search box.

Options

Jaguar provides a wide range of options for performing different kinds of calculations, for controlling the convergence of calculations, and for controlling the output of calculations. Most of these options can be set in the various tabs of the Jaguar panel. The options that are common to nearly all calculations are set in the following six tabs:

- Input—project entries, charge, spin, symmetry, basis set.
- Theory—HF, DFT, LMP2, CIS, restricted/unrestricted SCF.
- SCF—SCF convergence
- Properties—frequencies, surfaces, ESP charges, Mulliken populations, NBO analysis, multipole moments, polarizabilities, NMR shielding constants, VCD spectra
- Solvation—solvent, reference energy
- Output—printout options, output files

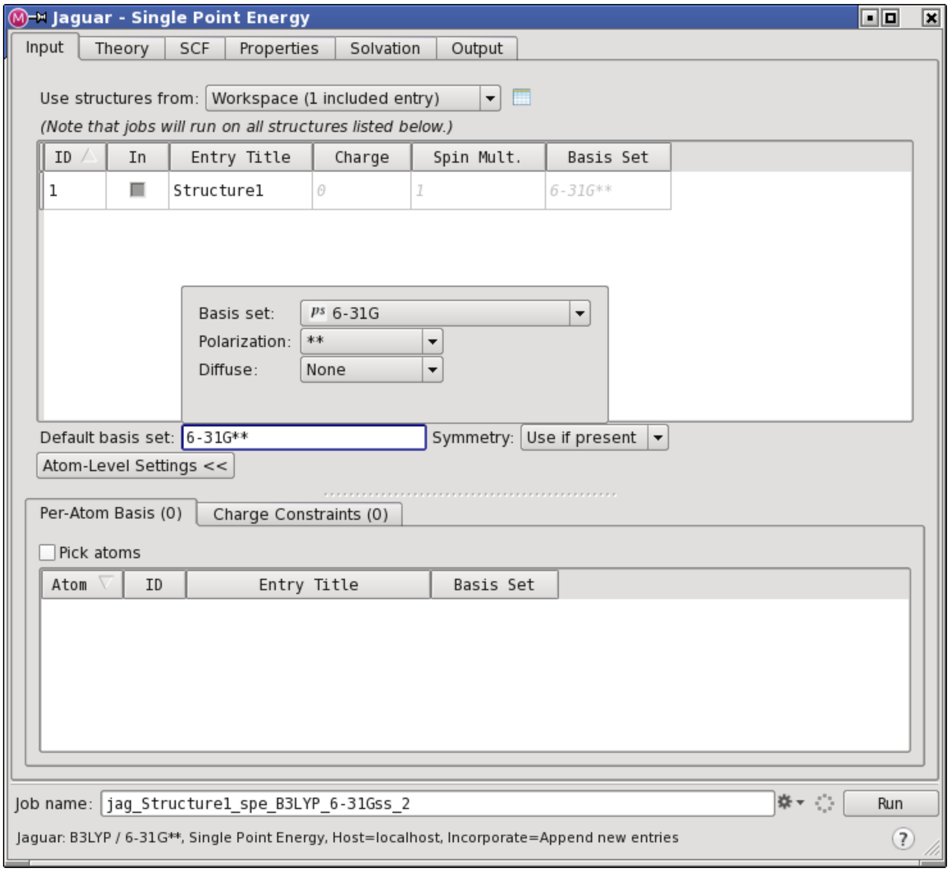
Several other tabs can appear in the Jaguar panel according to the task: Optimization, Transition State, IRC, Scan. These tabs are described in Chapter 4, along with other information about optimizations and scans. Output options are described in Chapter 6. This chapter describes the settings that can be made in the Input, Theory, SCF, Properties, and Solvation tabs.

The footnotes in this chapter indicate the Jaguar input file keywords and sections that correspond to settings made in the GUI. If you are working from the GUI, you can ignore these footnotes, but you may find them helpful if you decide to use input files to submit jobs without using the GUI, or if you want to edit keywords directly by using the Edit Job window described in Section 2.3 on page 11.

3.1 Input Structures

The source of the structures that are used in a Jaguar calculation can be chosen from the Use structures from option menu. The choices are to use the structures in the Workspace (the included entries) or the structures that are selected in the Project Table. The “table” button to the right of this option menu opens the Project Table panel, so you can select entries. If the structures you want to use are in a file, you should import the file into the project first. By default, these structures are automatically selected in the Project Table on import.

The structures are displayed in a table in the Input tab, which shows the entry ID and the entry title, and also has an In column, which you can use to display the structures in the Workspace, just as in the Project Table panel or the Entry List panel.


Jaguar - Single Point Energy

Input | Theory | SCF | Properties | Solvation | Output

Use structures from: Workspace (1 included entry)

(Note that jobs will run on all structures listed below.)

ID	In	Entry Title	Charge	Spin Mult.	Basis Set
1	■	Structure1	*0*	*1*	*6-31G***

Basis set: 6-31G
Polarization: **
Diffuse: None

Default basis set: 6-31G**   Symmetry: Use if present

Atom-Level Settings <<

Per-Atom Basis (0) | Charge Constraints (0)

Pick atoms

Atom	ID	Entry Title	Basis Set

Job name: jag_Structure1_spe_B3LYP_6-31Gss_2   Run

Jaguar: B3LYP / 6-31G**, Single Point Energy, Host=localhost, Incorporate=Append new entries

Figure 3.1. The Input tab.

The remaining columns in the table specify the charge[1], the spin multiplicity[2], and the basis set[3] for each structure. The defaults for the charge and spin multiplicity are determined from the structure. Usually these are 0 for the charge, 1 for the spin multiplicity (singlet) for a closed-shell molecule or 2 for an open-shell molecule. The default basis set is 6-31G**. The defaults are displayed in italics (to indicate that they are the defaults).

You can edit the table cells to set the charge, multiplicity, and basis set. When you change the charge, the multiplicity is changed to maintain consistency. If the new value is not what you want, you can change it to another consistent value. When you click in the Basis Set cell, a small window is displayed with controls for selecting the basis set (see Figure 3.1). This

1. Keyword **molchg** in the **gen** section.
2. Keyword **multip** in the **gen** section.
3. Keyword **basis** in the **gen** section.

window also shows the number of basis functions for the structure. You can change the default basis set in the Default basis set text box, below the table. Clicking in the text box opens a small window with controls for selecting the basis set, as for clicking in the cell. The basis set controls are discussed in Section 3.2 on page 33.

By default, Jaguar takes advantage of molecular symmetry in order to obtain CPU savings. Both Abelian and non-Abelian point groups are recognized. You can choose whether to use the full symmetry,[4] Abelian symmetry (D2h and subgroups),[5] or turn the use of symmetry off[6] from the Symmetry option menu. For information on how to make sure the symmetry of your input structure is treated as you expect, see Section 2.7.2 on page 23.

For some calculations, including LMP2 and IR intensities or hyperpolarizabilities, symmetry is not implemented and is disabled automatically during the job.

3.2 Basis Sets

You can choose a basis set from the three option menus labeled Basis set, Polarization, and Diffuse.These option menus are shown in the Transition State tab, the IRC tab, the Molecules tab of the Reaction panel, and the popup window that is displayed in the Input tab when you edit the Basis Set table cell or click in the Default basis set text box. For single-point energies, you can choose basis sets for individual atoms. Click Atom-Level Settings to show the atom-level settings, pick the atoms in the Workspace structure whose basis set you want to change, and set the basis set for those atoms in the table in the Per-Atom Basis tab.

The Basis set option menu lists all the basis sets available to Jaguar. If an option or menu is dimmed, it is incompatible with the rest of your input (for instance, the basis set could be missing basis functions for some atom or atoms in your molecule). Basis sets that have pseudo-spectral grids and dealiasing functions are preceded by *ps* in the Basis set option menu.

If you do not choose a basis set for a calculation, Jaguar uses the 6-31G** basis set if 6-31G** basis functions are available for all atoms in the input, and otherwise uses the LACVP** basis set by default. These basis sets are described in more detail below.

The Polarization option menu provides the choices none, *, and **. In general, the ** option places polarization functions on all atoms except for transition metals, and the * option places polarization functions on all atoms except for transition metals, H, and He. The tables below describe in detail the atoms that have polarization functions in each basis set. The cc-pV*n*Z and dyall basis sets include polarization functions by definition.

4. Keyword **isymm** = 8 in the **gen** section.
5. Keyword **isymm** = 8 and **idoabe**=1 in the **gen** section.
6. Keyword **isymm** = 0 in the **gen** section.

The Diffuse option menu provides the choices + and ++. The ++ option places diffuse functions on all atoms, while the + option places diffuse functions on all atoms except H and He. Diffuse functions are useful for calculations on van der Waals complexes or molecules that include atoms with negative charges.

For the basis set that you select, a message giving the number of functions and stating whether pseudospectral grids are available and whether ECPs are used is displayed below the controls.

Table 3.1 lists the available basis sets in Jaguar that do not use effective core potentials. The table indicates the atoms these basis sets can describe and shows which sets include the options of polarization and diffuse functions. The sarc-zora, dyall_v2z, and dyall_2zcvp basis sets are contracted for relativistic calculations with the ZORA Hamiltonian.

Table 3.1. Available basis sets that do not include effective core potentials

Basis Set	Atoms Included	Options	Method	# of d fns.	Refs.
STO-3G	H-Xe	* (Na-Xe)	analytic	5	112-116
3-21G	H-Xe	* (Na-Ar), + (Li-Ar), ++ (H-Ar)	H–Ar pseudospectral, K–Xe analytic; +, ++ analytic	6	117-119
4-21G	H-Ne	*, **	analytic	6	120
6-21G	H-Ar	*, **	analytic	6	117-119
4-31G	H-Ne	*, **	analytic	6	121-126
6-31G	H-Ar	*, **; +, ++	pseudospectral	6	122-127
6-311G	H-Ar	*, **, +, ++	pseudospectral	5	129-132
6-311G-3df-3pd[a]	H-Ar	+, ++	analytic	5	129-132
6-31G(tm)	H-Zn	*, **; +, ++ for H-Ar	H-Ar pseudospectral, K-Zn analytic	6	122-128
m6-31G(tm)	K-Zn		analytic	6	133
D95V	H, Li-Ne	*, **	analytic	6	134
D95	H, Li-Ne, Al-Cl	*, **	H, Li, C-F, Si-Cl pseudospectral, others analytic	6	134

Table 3.1. Available basis sets that do not include effective core potentials (Continued)

Basis Set	Atoms Included	Options	Method	# of d fns.	Refs.
MSV	H-Ru, Pd-Xe		analytic	5	135
cc-pVDZ, cc-pVDZ(-d) (without d functions)	H-Ar, Ga-Kr	+, ++	H, C-F, Si-Cl, Br pseudospectral, others analytic; +, ++ pseudospectral for H, He, B-Ne, Al-Ar, Br, others analytic	5	136-139
cc-pVTZ, cc-pVTZ(-f) (without f functions)	H-Ar, Ca, Ga-Kr	+, ++	H, He, B-Ar, As-Br pseudospectral, others analytic; +, ++ pseudospectral for H, He, B-Na, Al-Ar, As-Br, others analytic	5	136-139
cc-pVQZ(-g) (without g functions)	H-Ar, Ca, Ga-Kr	+, ++	H, Li, B-F, Na-Cl, Ca, Br pseudospectral, others analytic; +, ++ analytic	5	136
MIDIX	H, Li, C-F, Si-Cl, Br, I		H, C-F, P-Cl pseudospectral; Li, Si, Br, I analytic	5	140-142
TZV	H-Kr	*,** (Sc-Zn p only)	analytic	5	143
TZV(f)	H-Kr	*,* (Sc-Zn p and f)	analytic	5	143
Rappoport-SVPD or def2-svpd	H-Kr		pseudospectral	5	144
sarc-zora	La-Rn		analytic	5	145-147
dyall-v2z_zora-j-pt-gen	H-Rn	+, ++	analytic	5	148 - 155
dyall-2zcvp_zora-j-pt-gen	H-Rn	+, ++	analytic	5	148 - 155

a. This basis is referred to in the literature as 6-311G(3df-3pd).

Table 3.1 also gives the method used for integral evaluation: the fast pseudospectral method or the slower analytic method, in which all integrals are computed explicitly. The analytic method is used only when pseudospectral grids and dealiasing function sets for any atom in the molecule are not available. For molecules containing only atoms H–Ar, we recommend the 6-31G** basis set, which permits pseudospectral calculations (and is the default).

The column headed “# of d fns.” indicates whether d shells include the five real spherical functions d_{xy} , d_{xz} , d_{yz} , $d_{x^2-y^2}$, and $d_{2z^2-x^2-y^2}$, all with the same angular momentum ($l = 2$), or whether d shells include the six Cartesian d functions d_{x^2} , d_{y^2} , d_{z^2} , d_{xy} , d_{xz} , and d_{yz} . This choice also affects the dimension of the Fock matrix for diagonalization. To override this selection, set the keyword **numd** in the **gen** section of the input file, as described in Section 9.5.18 on page 244. The orbital coefficients are always printed out in terms of the six Cartesian functions. For basis sets with f functions, the real spherical set of 7 f functions is always used.

The references describing the basis sets are in the References list at the back of this manual.

The other available basis sets, which are listed in Table 3.2, include effective core potentials (ECPs). The names of eight of these basis sets begin with “LA” to indicate they were developed at Los Alamos National Laboratory. If the next character in the name is a “V”, the basis set is valence-only, containing only the highest s and p shells for main group atoms and the highest s, p, and d shells for transition metals. For example, 5s and 5p would be included for tellurium, and 6s, 5d, and 6p for tungsten. “LAV1” indicates that the basis set has been fully contracted to form a minimal basis set, “LAV2” that the last Gaussian has been uncontracted to form a double zeta basis, and “LAV3” that all of the s functions and the last p and d Gaussian have been uncontracted.

Table 3.2. Basis sets contained in Jaguar that include effective core potentials

Basis Set	Atoms in ECP	Other Atoms	Options	Refs.
LAV1S	Na-La, Hf-Bi	H-Ne (STO-3G)	* (H-Ne)	156-157
LAV2D	Na-La, Hf-Bi	H, Li-Ne (D95V)	*, ** (H, Li-Ne)	156-157
LAV2P	Na-La, Hf-Bi	H-Ne (6-31G)	*, ** (H-Ne); +, ++ (H-Ne)	156-157
LAV3D	Na-La, Hf-Bi	H, Li-Ne (D95V)	*, ** (H, Li-Ne)	156-157
LAV3P	Na-La, Hf-Bi	H-Ne (6-31G)	*, ** (H-Ne); +, ++ (H-Ne)	156-157
LACVD	K-Cu, Rb-Ag, Cs-La, Hf-Au	H, Li-Ne (D95V); Na-Ar, Zn-Kr, Cd-Xe, Hg-Bi (LAV3D)	*, ** (H, Li-Ne)	158
LACVP	K-Cu, Rb-Ag, Cs-La, Hf-Au	H-Ar (6-31G); Zn-Kr, Cd-Xe, Hg-Bi (LAV3P)	*, ** (H-Ar); +, ++ (H-Ar)	158
LACV3P	K-Cu, Rb-Ag, Cs-La, Hf-Au	H-Ar (6-311G); Zn-Kr, Cd-Xe, Hg-Bi (LAV3P)	*, ** (H-Ar); +, ++ (H-Ar, plus metal diffuse d)	159
cc-pVTZ-pp	Ga-Kr, In-Xe	H-Ar, Ca (cc-pVTZ)	+, ++	160-162

Table 3.2. Basis sets contained in Jaguar that include effective core potentials (Continued)

Basis Set	Atoms in ECP	Other Atoms	Options	Refs.
cc-pVTZ-pp(-f)	Ga-Kr, In-Xe	H-Ar, Ca (cc-pVTZ(-f))	+, ++	160-162
CSDZ	Ce-Lu	H-Ar (6-31G); Zn-Kr, Cd-Xe, Hg-Bi (LAV3P); K-Cu, Rb-Ag, Cs-La, Hf-Au (LACVP)	*, ** (H-Ar); +, ++ (H-Ar)	163
ERMLER2	K-Lr	H-Ar: 6-31G	*, ** (H-Ar, Sc-Kr, Y-Xe, Hf-Rn) +, ++ (H-Ar, Ga-Kr, In-Xe, Tl-Rn)	164-171

Names starting with "LACV" indicate that the basis set also includes the outermost core orbitals (e.g., 5s5p6s5d6p for W). The last letter in each LA basis set name refers to the basis set used for atoms *not* described by ECPs: S indicates the STO-3G basis set, D indicates the D95V basis set, and P indicates the 6-31G set developed by Pople and coworkers. (Note that in addition, for some atoms, the LACVD and LACVP basis sets use the same basis functions as the LAV3D and LAV3P basis sets, respectively.)

The Los Alamos effective core potentials, which were developed by Hay and Wadt, include one-electron mass-velocity and Darwin relativistic corrections for elements beyond Kr.

The Cundari-Stevens ECP basis set [163], named CSDZ, has been provided to cover the lanthanides. This basis set uses a relativistic effective core potential for the inner core electrons and treats the outer core and valence electrons with a 4s/4p/2d/2f basis set.

The ECP basis set developed by Ermler and coworkers [164-169], named ERMLER2, is also available. The basis set provided is the "small core" set that includes the outer core orbitals in the valence space, in the same way as the LACV basis sets. The basis set is a double-zeta contraction in which the outermost primitive function in each symmetry has been uncontracted. The core is treated by a relativistic effective core potential for all elements. For Tl-Rn the refitted ECPs have been taken from Wildman et al. [170], but the basis set from the original ECPs has been retained, because the new basis sets are much larger, and do not match the basis sets for the other elements. Polarization and diffuse functions for the 4p, 5p, and 6p elements and polarization functions for the 3d, 4d, and 5d elements have been added from the relativistic all-electron double-zeta basis sets of Dyall [171].

In Table 3.2, the atoms described by the effective core potential are listed first, followed by the atoms described by the alternate basis set or sets. The other table entries provide the same information as that given in the previous table, except that the polarization functions are only applied to atoms obtained from the non-ECP basis sets, with the exception of the ERMLER2 basis sets. All ECP basis sets use five d functions, as described earlier in this section.

Currently, the LACVP, LAV3P, LACV3P, and CSDZ basis sets use the pseudospectral method, while all other ECP basis sets use the analytic method, with the exception of Br and I in the cc-pVTZ-pp and cc-pVTZ-pp(-f) basis sets. We strongly recommend using either the LACVP or the LACV3P basis set for non-lanthanide molecules containing atoms beyond Ar in the periodic table, especially for studies involving charge transfer, d^0 metals, or correlated wave functions. The LACV3P basis set seems to give substantial improvements over the LACVP basis set for HF, LDA, and B3LYP atomic state splittings. The LACV3P++ basis set, which includes a diffuse d function on any metal atoms, is useful for calculations on anions or low-spin M(0) complexes of the late first row metals.

Note: You should not use ECP basis sets to calculate NMR shieldings on the ECP atom, as the ECP orbitals do not have the necessary density at the nucleus for the shielding calculation.

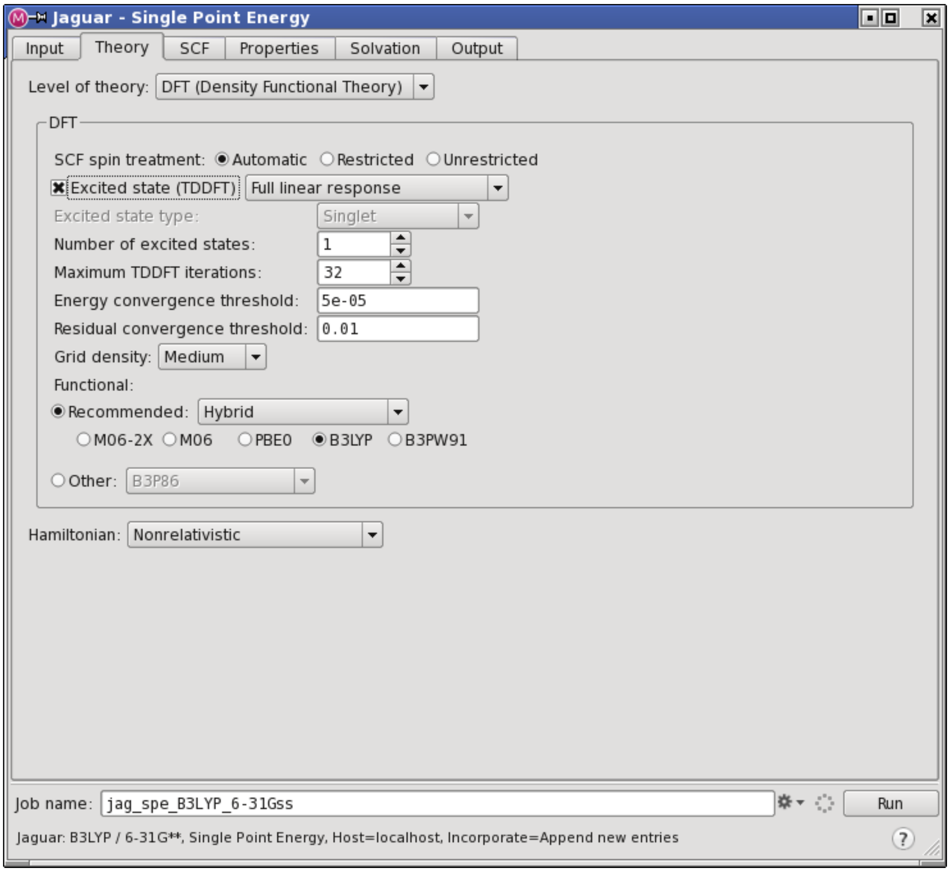
3.3 Density Functional Theory (DFT) Settings

The density functional theory module in Jaguar allows you to employ a variety of functionals to describe exchange and correlation for either open or closed shell systems. The theory is described in Section 8.3 on page 188. This section describes how to set up a DFT calculation in Jaguar. You can perform DFT geometry optimizations, solvation calculations, charge fitting, and all other calculations and properties available for Hartree-Fock wave functions.

DFT calculations are selected by choosing DFT (Density Functional Theory) from the Level of theory option menu in the Theory tab. When you make this choice, controls for DFT calculation are displayed below this menu.

The default spin treatment is spin-restricted for closed shell molecules and spin-unrestricted[7] (UDFT) for open-shell molecules, which is selected with the Automatic option. You can run a spin-restricted[8] (RODFT) calculation for open-shell molecules by selecting Restricted, and a spin-unrestricted calculation on closed-shell molecules by selecting Unrestricted. The latter will produce identical alpha and beta orbitals close to equilibrium, but may differ in regions where a closed shell is no longer a good description of the electronic structure.

7. Keyword **iuhf** = 1 in the **gen** section.
8. Keyword **iuhf** = 0 in the **gen** section.


Jaguar - Single Point Energy

Input | Theory | SCF | Properties | Solvation | Output

Level of theory: DFT (Density Functional Theory)

DFT

SCF spin treatment: [✓ Automatic] Restricted Unrestricted

[✓ Excited state (TDDFT)] Full linear response

Excited state type: Singlet

Number of excited states: 1

Maximum TDDFT iterations: 32

Energy convergence threshold: 5e-05

Residual convergence threshold: 0.01

Grid density: Medium

Functional:

[✓ Recommended]: Hybrid

M06-2X M06 PBE0 [✓ B3LYP] B3PW91

Other: B3P86

Hamiltonian: Nonrelativistic

Job name: jag_spe_B3LYP_6-31Gss

Run

Jaguar: B3LYP / 6-31G**, Single Point Energy, Host=localhost, Incorporate=Append new entries

Figure 3.2. The Theory tab showing DFT controls.

You can run calculations for excited states using time-dependent density functional theory (TDDFT)[9], both in the gas phase and with the PBF solvation model. TDDFT has been implemented in Jaguar [240] with both the full linear response[10] and the Tamm-Dancoff approximation[11] [30]. To run a TDDFT calculation, select Excited state (TDDFT), choose the approximation from the option menu, and enter the number of excited states in the Number of excited states text box. You should select more excited states than you are actually interested in, for two reasons. The first is that the initial guess might not accurately reflect the final states, and the second is to ensure that near-degeneracies are accounted for. You can set the number of TDDFT iterations[12] and some convergence thresholds[13,14]. For a closed-shell molecule, you

9. Keyword **itddft** = 1 in the **gen** section.
10. Keyword **itda** = 0 in the **gen** section.
11. Keyword **itda** = 1 in the **gen** section.
12. Keyword **nroot** in the **gen** section.

can use a spin-restricted version of TDDFT and generate the singlet excited states,[15] the triplet excited states,[16] or both, by choosing from the Excited state type option menu. For open-shell molecules, all spin states generated by single excitations from the reference are represented, but are not classified by spin, as the method is spin-unrestricted for these molecules.

When excited states are calculated, a spectrum file is written that contains the UV/visible spectral data, *jobname*`_uvv.spm`. This file can be read into the Spectrum Plot panel in Maestro to generate a simulated UV/visible spectrum. The Spectrum Plot panel is described in Section 3.12 on page 80.

The Grid density menu determines the grid for DFT calculations. By default, DFT calculations use grids with a medium point density, but finer density grids are available. You can choose between Medium,[17] Fine,[18] or Maximum[19] grid density. If you read an input file that contained some other grid density, the grid density is set to Other, otherwise this option is unavailable. If you choose a grid density from this menu, the previous grid density specification is replaced.

The functionals that are recommended for most uses can be selected under the Recommended option. There are five classes of functionals available: hybrid functionals, which include a Hartree-Fock exchange contribution as well as local and nonlocal functionals; gradient-corrected pure density functionals; local density functionals; a posteriori corrected functionals, which apply an energy correction after the SCF (for example, for dispersion); and long-range-corrected functionals, which partition the exchange functional into short-range and long-range, and use different functionals in the two ranges. To use other functionals, select Other and choose the functional from the list.

Many of the hybrid methods employ either the parameters developed for Becke's three-parameter method [32, 33] (Becke 3) or the parameters developed for Becke's Half & Half method [31]. A number of recently developed hybrid and non-hybrid functionals are also included. The zero-damping, two-body only D3 correction of Grimme et al. [50, 51] is available for many of the functionals. Some functionals with the D3 correction are included in the recommended A posteriori corrected functionals list; the rest are in the Other list. These functionals have -D3 appended to the uncorrected functional name.

The available functionals are listed in the sections below. Functionals for which the D3 correction is available are indicated in these lists. The names of the functionals are values of the keyword **dftname**, which you can use in the **gen** section of the input file to specify the functional. The names are case-insensitive.

13. Keyword **econvci** in the **gen** section.
14. Keyword **dconvci** in the **gen** section.
15. Keyword **rsinglet** in the **gen** section.
16. Keyword **rtriplet** in the **gen** section.
17. Keywords **gdftmed** = –10, **gdftfine** = –11, and **gdftgrad** = –12 in the **gen** section.
18. Keywords **gdftmed**, **gdftfine**, and **gdftgrad** = –13 in the **gen** section.
19. Keywords **gdftmed = gdftfine** = **gdftgrad** = –14 in the **gen** section.

Following the lists is a description of the B3LYP-MM and the B3LYP-LOC a posteriori-corrected functionals.

3.3.1 LDA (Local Density Functionals)

Functionals with local exchange only:

- HFS[20]: Slater local exchange functional [34]
- Xalpha[21]: Xα local exchange functional [34]

Functionals with local exchange and local correlation:

- SVWN[22]: Slater local exchange functional [34], Vosko-Wilk-Nusair (VWN) local correlation functional [35]
- SVWN5[23]: Slater local exchange functional [34], Vosko-Wilk-Nusair 5 (VWN5) local correlation functional [35]

3.3.2 Gradient-Corrected Functionals

- BLYP[24]: Exchange: Slater local functional [34], Becke 1988 nonlocal gradient correction [37]; correlation: Lee-Yang-Parr local and nonlocal functionals [38]; D3 correction available
- BPW91[25]: Exchange: Slater local functional [34], Becke 1988 nonlocal gradient correction [37]; correlation: Perdew-Wang 1991 GGA-II local and nonlocal functionals [36]
- BP86[26]: Exchange: Slater local functional [34], Becke 1988 non-local gradient correction [37]; correlation: Perdew-Zunger 1981 local functional [39], Perdew 1986 gradient correction functional [40]; D3 correction available.
- BP86-VWN5[27]: Exchange: Slater local functional [34], Becke 1988 nonlocal gradient correction [37]; correlation: Vosko-Wilk-Nusair (VWN) local functional [35], Perdew 1986 gradient correction functional [40]
- PWPW91[28]: Exchange: Slater local functional [34], Perdew-Wang 1991 gradient correction functional [36]; correlation: Perdew-Wang 1991 GGA-II local and nonlocal functionals [36]

20. Keyword **dftname** = hfs in the **gen** section.
21. Keyword **dftname** = xalpha in the **gen** section.
22. Keyword **dftname** = svwn in the **gen** section.
23. Keyword **dftname** = svwn5 in the **gen** section.
24. Keyword **dftname** = blyp in the **gen** section.
25. Keyword **dftname** = bpw91 in the **gen** section.
26. Keyword **dftname** = bp86 in the **gen** section.
27. Keyword **dftname** = bp86-vwn5 in the **gen** section.
28. Keyword **dftname** = pwpw91 in the **gen** section.

- PBE[29]: Perdew-Burke-Ernzerhof local and nonlocal exchange and correlation functional [47]; D3 correction available.
- HCTH407[30]: Hamprecht-Cohen-Tozer-Handy functional [45] including local and nonlocal exchange and correlation, reparametrized with a training set of 407 molecules by Boese and Handy [46]
- M06-L[31]: Parameterization by Zhao and Truhlar of various functionals that include gradient and kinetic energy spin density functionals [60]
- OLYP[32]: Exchange: Slater local functional [34], OPTX nonlocal exchange of Handy and Cohen [48]; correlation: Lee-Yang-Parr local and nonlocal functionals [38]; D3 correction available.
- B97-D[33]: Modified Becke 1997 functional [42] with Grimme's long-range dispersion correction, useful for noncovalent interactions [49]. D3 correction also available.

3.3.3 Hybrid Functionals

- B3LYP[34]: Exchange: exact HF, Slater local functional [34], Becke 1988 nonlocal gradient correction [37]; correlation: Vosko-Wilk-Nusair (VWN) local functional [35], Lee-Yang-Parr local and nonlocal functional [38]. See refs. 32 and 33. D3 correction available.
- B3PW91[35]: Exchange: exact HF, Slater local functional [34], Becke 1988 nonlocal gradient correction [37]; correlation: Perdew-Wang 1991 local and GGA-II nonlocal functional [36]; D3 correction available.
- B3P86[36]: Exchange: exact HF, Slater local exchange functional [34], Becke 1988 nonlocal gradient correction [37]; correlation: Vosko-Wilk-Nusair (VWN) local functional [35], Perdew 1986 nonlocal gradient correction [40]
- B1B95[37]: Becke's hybrid functional from 1996 containing 28% HF exact exchange [41]. D3 correction available.
- BB1K[38]: Reparametrization of Becke's 1996 hybrid functional for reaction barriers, containing 42% HF exact exchange [56]

29. Keyword **dftname** = pbe in the **gen** section.
30. Keyword **dftname** = hcth407 in the **gen** section.
31. Keyword **dftname** = m06-l in the **gen** section.
32. Keyword **dftname** = olyp in the **gen** section.
33. Keyword **dftname** = b97-d in the **gen** section.
34. Keyword **dftname** = b3lyp in the **gen** section. The functional follows Gaussian's definition.
35. Keyword **dftname** = b3pw91 in the **gen** section.
36. Keyword **dftname** = b3p86 in the **gen** section.
37. Keyword **dftname** = b1b95 in the **gen** section.
38. Keyword **dftname** = bb1k in the **gen** section.

- B97-1[39]: Reparametrization of Becke's 1997 hybrid functional [42] by Hamprecht, Cohen, Tozer, and Handy [45]
- B98[40]: Becke's 1998 hybrid functional including the Laplacian of the density and kinetic energy density terms as well as gradient terms [43]
- SB98[41]: Schmider and Becke reparametrization of Becke's 1998 functional [44]
- BHandH[42]: 50% exact HF exchange, 50% Slater local exchange functional [34]
- BHandHLYP[43]: Exchange: 50% exact HF exchange, 50% Slater local exchange functional [34]; correlation: Lee-Yang-Parr local and nonlocal functionals [38]
- X3LYP[44]: Extension of B3LYP by Xu and Goddard to include Perdew-Wang 1991 gradient correction exchange functional [36], with exchange parametrized to fit Gaussian exchange density [55]
- MPW1PW91[45]: Hybrid functional including modification of Perdew-Wang gradient correction exchange functional, by Adamo and Barone [52]. Exchange: Exact HF exchange, Slater local functional [34], Perdew-Wang 1991 gradient correction functional [36]; correlation: Perdew-Wang 1991 GGA-II local and nonlocal functionals [36]
- MPW1K[46]: Reoptimization of *m*PW1PW91 functional parameter for prediction of reaction barrier heights, by Lynch, Fast, Harris, and Truhlar [54]
- PWB6K[47]: Reoptimization of MPWB1K functionals for simultaneous accuracy of bond energies, barrier heights, and nonbonded interactions, by Zhao and Truhlar [56].
- PW6B95[48]: Reoptimization of MPW1B95 functionals for simultaneous accuracy of bond energies, barrier heights, and nonbonded interactions, by Zhao and Truhlar [56]. D3 correction available.
- PBE0[49]: Functional due to Adamo and Barone [53] based on PBE functional. 25% exact HF exchange, 75% PBE non-local exchange. Correlation: Perdew-Burke-Ernzerhof [47] local and nonlocal correlation. Also known as PBE1PBE. D3 correction available.

39. Keyword **dftname** = b97-1 in the **gen** section.
40. Keyword **dftname** = b98 in the **gen** section.
41. Keyword **dftname** = sb98 in the **gen** section.
42. Keyword **dftname** = bhandh in the **gen** section.
43. Keyword **dftname** = bhandhlyp in the **gen** section.
44. Keyword **dftname** = x3lyp in the **gen** section.
45. Keyword **dftname** = mpw1pw91 in the **gen** section.
46. Keyword **dftname** = mpw1k in the **gen** section.
47. Keyword **dftname** = pwb6k in the **gen** section.
48. Keyword **dftname** = pw6b95 in the **gen** section.
49. Keyword **dftname** = pbe0 in the **gen** section.

- M05[50]: Hybrid functional parametrized for broad accuracy, including noncovalent interactions, kinetics, and interactions with metals, by Zhao, Schultz, and Truhlar [58, 59]. D3 correction available.
- M05-2X[51]: Hybrid functional with larger HF exchange component, similar to M05 but parametrized for nonmetals, by Zhao, Schultz, and Truhlar [58, 59]. D3 correction available.
- M06-HF[52]: Zhao and Truhlar functional with full HF exchange and M06 local functionals that eliminates long-range self-interaction. [61]. D3 correction available.
- M06[53]: Zhao and Truhlar functional, parametrized with metallic systems, for organometallic and inorganic chemistry and noncovalent interactions [62]. D3 correction available.
- M06-2X[54]: Zhao and Truhlar functional, parametrized for nonmetals, for main-group thermochemistry, kinetics, noncovalent interactions, and electronic excitation energies to valence and Rydberg states [62]. D3 correction available.
- O3LYP[55]: Exchange: exact HF, Slater local functional [34], OPTX nonlocal functional of Handy and Cohen [48]; correlation: Lee-Yang-Parr local and nonlocal functionals [38]
- M08-HX[56]: Zhao and Truhlar functional, constraining the reduced density gradient to exact exchange and correlation functional forms through second order. Parametrized on a broad range of properties [63].
- M08-SO[57]: Zhao and Truhlar functional, constraining the reduced density gradient to exact exchange and correlation functional forms through second order. Parametrized on a broad range of properties [63].

3.3.4 Long-Range-Corrected Functionals

These functionals include range separation using a partitioning of the Coulomb operator (see Section 8.3.1 on page 189) to partition the HF exchange into short-range and long-range parts. The units of the range separation parameter ω are bohr^{-1}. Short-range and long-range exchange functionals are obtained from the full functional either by the method of Iikura et al. [64] or that of Henderson et al. [66]. The modified versions are prefixed with SR- or LR-.

50. Keyword **dftname** = m05 in the **gen** section.
51. Keyword **dftname** = m05-2x in the **gen** section.
52. Keyword **dftname** = m06-hf in the **gen** section.
53. Keyword **dftname** = m06 in the **gen** section.
54. Keyword **dftname** = m06-2x in the **gen** section.
55. Keyword **dftname** = o3lyp in the **gen** section.
56. Keyword **dftname** = m08-hx in the **gen** section.
57. Keyword **dftname** = m08-hx in the **gen** section.

- LRC-BLYP[58]: Yanai, Tew, and Handy functional [65]. Short-range exchange: SR-Becke 88 [37]; long-range exchange: LR-HF; ω=0.33; correlation: LYP [38].
- CAM-B3LYP[59]: Yanai, Tew, and Handy functional [65]. Short-range exchange: 19% SR-HF, 81% SR-Becke 88 [37]; long-range exchange: 65% LR-HF, 35% LR-Becke 88 [37]; ω=0.33; correlation: LYP [38]; D3 correction available.
- uPBE[60]: Long-range corrected PBE functional. Short range exchange: SR-PBE; long range exchange: LR-HF; ω=0.30; correlation: PBE [47].
- uPBE0[61]: Long-range corrected PBE hybrid. Short range exchange: 25% SR-HF, 75% SR-PBE; long range exchange: LR-HF; ω=0.30; correlation: PBE [47].
- wPBE[62]: Vydrov and Scuseria functional [67]. Short range exchange: SR-PBE; long range exchange: LR-HF; ω=0.3[63]; correlation: PBE [47]; D3 correction available.
- wPBEh[64]: Rohrdanz, Martins, and Herbert functional [68]. Short range exchange: 20% SR-HF, 80% SR-PBE; long range exchange: LR-HF; ω=0.2; correlation: PBE [47].
- HSE03[65]: Heyd, Scuseria, and Ernzerhof functional [69]. Short range exchange: 25% SR-HF, 75% SR-PBE; long-range exchange: LR-PBE; ω=0.33[66]; correlation: PBE [47].
- HSE06[67]: Heyd, Scuseria, and Ernzerhof functional [71]. Short range exchange: 25% SR-HF, 75% SR-PBE; long-range exchange: LR-PBE; ω=0.11; correlation: PBE [47].
- M11[68]: Peverati and Truhlar functional [72]. Short range exchange: 42.8% SR-M11 [72] and HF; long-range exchange: HF; ω=0.25; correlation: M11 [72].
- M11-L[69]: Peverati and Truhlar range separation of non-hybrid functionals [73]. Short range exchange: SR-M11 [73]; long-range exchange: LR-M11 [73]; ω=0.25; correlation: M11-L [73].
- wB97[70]: Chai and Head-Gordon functional [73]. Short range exchange: SR-Becke 97 [42]; long-range exchange: HF; ω=0.4; correlation: Becke 97 [42].

58. Keyword **dftname** = lrc-blyp in the **gen** section.
59. Keyword **dftname** = cam-b3lyp in the **gen** section.
60. Keyword **dftname** = upbe in the **gen** section. No reference; values from NWChem.
61. Keyword **dftname** = upbe0 in the **gen** section. No reference; values from NWChem.
62. Keyword **dftname** = wpbe or lc-wpbe in the **gen** section.
63. NWChem value; the authors' value is 0.4
64. Keyword **dftname** = wpbeh in the **gen** section.
65. Keyword **dftname** = hse03 in the **gen** section.
66. This is the value used in NWChem; it is not the value reported in the paper or the values actually used in the paper: see the erratum in Ref. 70.
67. Keyword **dftname** = hse06 in the **gen** section.
68. Keyword **dftname** = m11 in the **gen** section.
69. Keyword **dftname** = m11-l in the **gen** section.
70. Keyword **dftname** = wb97 in the **gen** section.

- wB97x[71]: Chai and Head-Gordon functional [73]. Short-range exchange: 16% SR-HF, SR-Becke 97 [42]; long-range exchange: HF; ω=0.3; correlation: Becke 97 [42].
- BNL[72]: Baer, Neuhauser, and Livshits functional [75, 76]. Short-range exchange: 90% SR-Savin; long-range exchange: HF; ω=0.3; correlation: LYP [38].

3.3.5 B3LYP-MM A Posteriori-Corrected Functional

Recently, a number of functionals augmented with the so-called a posteriori corrections have been developed. These functionals respond to the need for a greater accuracy in computing non-covalent interactions with DFT (the popular functional B3LYP, as it is well-known, largely neglects van der Waals interactions). In contrast to the functionals in which non-covalent interactions are accounted for in the functional through a modified dependence on electronic density (such as M06-2X), the a posteriori-corrected functionals add empirical corrections to the energy (and possibly other properties) after it has been computed with regular DFT. These empirical corrections typically depend on the Cartesian coordinates of atoms, and the types of atoms, rather than the electronic density of the system, and are therefore reminiscent of corrections used in molecular mechanics methods. The simplicity of the implementation, a negligible computational cost associated with these corrections, and a much higher resultant accuracy compensate for the burden of having to parametrize these corrections for different atom and non-covalent interaction types.

Jaguar includes B3LYP-MM, an a posteriori-corrected functional parametrized on a gigantic data set of complexes that involve non-covalent interactions [65]. This functional has been shown to be more accurate for describing the energetics of van-der-Waals, hydrogen-bonded, and cation-pi interactions than B3LYP-D3 and M06-2X which also account for these types of interactions. The energy predicted by B3LYP-MM is made up of two components: the regular B3LYP energy and the MM-like correction. The latter comprises three parts:

- Lennard-Jones-like correction to account for dispersion and induction effects (eq. (2) in [65])
- Hydrogen bond correction to account for explicit hydrogen bonds energetics (eq. (5) in [65])
- Cation-pi correction to account for cation-pi interactions (eq. (6) in [65])

Use of B3LYP-MM has the following limitations in the current implementation:

- It is parametrized for two basis sets only, LACVP* and cc-pVDZ++. The latter basis set gives slightly more accurate interaction energies, according to the benchmarks.

71. Keyword **dftname** = wb97x in the **gen** section.
72. Keyword **dftname** = bnl in the **gen** section.

- The Lennard-Jones-like correction is available only for the elements H, C, N, O, F, S, Cl, and Br. The job will fail if the structure contains other elements. Positively-charged metal atoms, for which the Lennard-Jones-like correction is not computed, are not subject to this limitation. You can, however, set `fall_through=1` in the **gen** section to compute the corrections for only the parametrized elements and set them to zero for any other elements, rather than having the job fail.
- The cation-pi correction is computed for transition-metal ions, but was not parametrized with these ions. B3LYP-MM should be used with caution in such cases.

The B3LYP-MM functional corrections have been optimized in such a way that they already include the counterpoise (CP) correction which is needed to compensate for the basis set superposition error (BSSE). To compute the interaction energy of a complex X-Y with B3LYP-MM and correct for the BSSE, simply compute the energy of the individual fragments X and Y, and the energy of the complex X-Y, and take the difference.

If you would still like to apply the CP correction as described in Section 2.4.7 on page 18, B3LYP-MM has parameters that do not absorb the CP correction, and therefore, do not account for BSSE. To enable these parameters, set `cpoise=1` in the input file. The explicit CP corrections (when `cpoise=1`) give slightly better interaction energies than the default implicit CP corrections (`cpoise=0`).

To use the B3LYP-MM functional, select B3LYP-MM under A posteriori-corrected, or set `dftname=b3lyp-mm` in the **gen** section.

B3LYP-MM cannot be used for transition state searches.

3.3.6 B3LYP-LOC A Posteriori-Corrected Functional

DFT functionals available in Jaguar fail to accurately describe non-dynamic correlation effects to a greater or lesser degree. These failures arise from a combination of the essentially single-determinant nature of the DFT wave function and fundamental inaccuracies of the approximate density functionals. Problems with non-dynamic correlation often manifest themselves in systems with non-equilibrium geometries, systems containing transition metal atoms, organic biradicals, etc. As a result of the non-dynamic correlation errors, DFT energies computed for such systems may contain significant errors.

To address the problem of non-dynamic correlation at the DFT level, Jaguar provides the localized orbital correction (LOC) [78-86]. The basic idea of this correction is that DFT errors due to non-dynamic correlation can be localized to atoms and bonds, and, once localized, can be removed. In our tests, the B3LYP functional has showed the greatest propensity toward localizing non-dynamic correlation errors on atoms and bonds [78]. Therefore, we concentrated on developing a method that combines B3LYP and LOC (B3LYP-LOC). The corrections, trained on experimental data and having an essentially semi-empirical character, are applied to the final DFT energy in the a posteriori manner, thus correcting the total energy.

Different corrections are applied to different atom and bond types. The application of the corrections depends on the details of the valence-bond structure and not on Cartesian coordinates. Consequently, when using the B3LYP-LOC method you must ascertain that the valence bond structure of the molecule is reasonable.

Different sets of basis-set dependent LOC parameters apply to different types and parts of chemical structures. The current implementation of the B3LYP-LOC method in Jaguar includes 22 parameters for structures with equilibrium-type geometries [78], 16 parameters accounting for electron transfer to and from atoms and bonds (and essentially responsible for dynamic correlation errors in ionization potential and electron affinity calculations) [79], and 8 parameters for transition state-like structures [81]. Included also are several types of corrections for octahedral complexes of 3d metals (Sc-Zn). Of these, there are 5 parameters for spin splitting [84], 7 parameters for electron addition/subtraction to/from the metal center (essentially responsible for the description of redox potentials) [85], and 6 parameters for ligand removal [86]. Note that the Jaguar implementation differs in some aspects from that in the original papers [78-79] due to subsequently discovered improvements and corrections. The LOC corrections are applied only to those aspects of the valence-bond structure which are recognized as centers where non-dynamic correlation errors are localized, and for which the parameters are available. For parts of the structure for which the B3LYP-LOC method has no parameterization, no corrections are applied.

To use the B3LYP-LOC method, choose B3LYP-LOC from the Other option menu, or set `dftname=b3lyp-loc` in the input file. To print a detailed summary of all applied LOC corrections in the output file, set `printaposteri=1` in the input file.

The following list summarizes the current limitations of the B3LYP-LOC method:

- The LOC corrections can currently be combined with the B3LYP functional only.
- The LOC corrections are basis-set dependent. They are currently available for the basis sets 6-31G*, cc-pVDZ, cc-pVTZ++, and 6-311++G-3df-3pd. If the program recognizes a qualified transition-metal system, only the LACV3P and LACV3P* basis sets can be used. The largest basis sets are recommended for the highest-quality results.
- The method does not explicitly depend on Cartesian coordinates. Different resonance structures can have different corrections applied. It is recommended that you work with the dominant resonance structure, if there is one. Sulfonyl and sulfinyl groups should be represented with the separated charge in the S-O bonds. Thus, for the best results, the B3LYP-LOC method should be applied to dimethylsulfone $(CH3)_2SO_2$ represented as $CH3\text{-}S^{2+}(\text{-}O^{-})(\text{-}O^{-})\text{-}CH3$ in Maestro. See Ref. 78 for more recommendations of this type.

- Because the LOC corrections do not explicitly depend on atomic coordinates, their derivatives with respect to these variables are trivially zero. When the B3LYP-LOC method is used for geometry optimizations, only B3LYP gradients are used. Zero LOC gradients are consistent with the energy of the B3LYP-LOC method as long as the valence-bond structure remains the same in the course of the geometry optimization. If the valence bond structure abruptly changes (for example, a bond breaks), this might lead to kinks on the B3LYP-LOC potential energy surface and, because of the zero LOC gradient assumption, to problematic geometry optimization convergence.
- Not all bond and atom types have LOC parameters associated with them.

In order to apply the B3LYP-LOC method to the calculation of a transition state, the structures of the reactant, the product, and the transition state must be supplied (in both the Maestro and Jaguar input files). The input files applicable to a QST search with traditional DFT functionals are also applicable to a transition state search with the B3LYP-LOC method.

The B3LYP-LOC method is one of the recommended methods to predict the enthalpy of formation and atomization energies. B3LYP-LOC in combination with the 6-311++G-3df-3pd basis set can be automatically chosen in the Heat of Formation workflow panel (see Section 5.4 on page 121 for more details).

3.3.7 Constrained DFT Calculations

Jaguar has the capability of performing constrained DFT calculations, in which you can define groups of atoms for which the electron density is constrained to a particular value. This is useful for ensuring that one group has a particular charge, for example, in charge transfer calculations. These calculations can only be performed as a single-point energy calculation.

The constraints can be set up in the Charge Constraints tab, which you can display by clicking Atom-Level Settings in the Input tab. First, display the entry that you want to constrain in the Workspace, using the Project table selected entries table. Click New Constraint to add a row to the constraints table, then pick the atoms in the Workspace to define the constraint. When you have finished picking atoms, you can set the charge by editing the Charge cell of the table row. A small table is displayed, listing the atoms and their weight in the constraint. You can change the weights in this table. This information is also displayed in the tooltip when you pause the pointer over the Charge cell.You can add as many constraints as you like, and atoms can be included in multiple constraints.

The constraints are written in the **cdft** section of the input file—see Section 9.16 on page 287.

3.4 Hartree-Fock and TDHF Settings

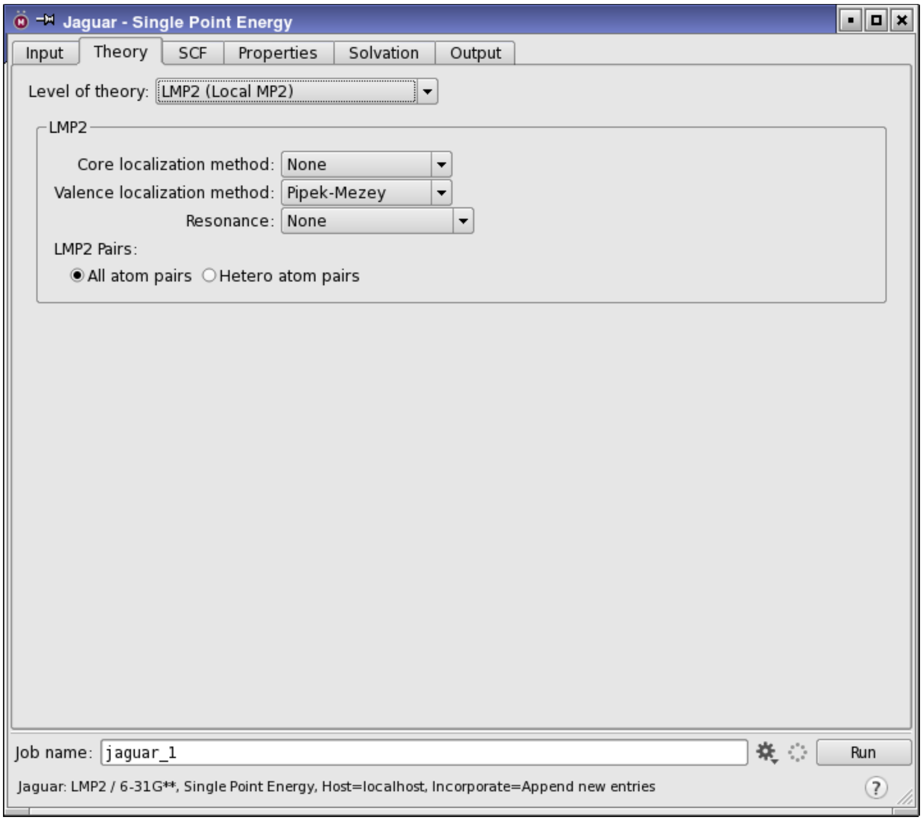
In addition to selecting unrestricted (UHF) or spin-restricted (RHF) treatment of molecules (the automatic treatment does RHF for closed-shell molecules and UHF for open-shell molecules), the controls available when you choose HF (Hartree-Fock) from the Level of theory option menu allow you to set up calculations for singly excited states. These calculations are available for a closed-shell Hartree-Fock reference wave function as spin-restricted states, or for open-shell molecules with unrestricted spin, based on a UHF reference. They can be run in the gas phase or with the PBF solvation model. To do a TDHF calculation, select Excited state (TDHF).[73] A configuration interaction singles (CIS) calculation can be run by choosing the Tamm-Dancoff approximation[74] from the option menu. For closed-shell molecules, you can generate the restricted singlet[75] or restricted triplet[76] excited states, or both, by choosing from the Excited state type option menu. You can then enter the number of states you want to generate in the Number of excited states text box.[77] You should consider including more excited states than you need, to ensure that you include the states of interest and cover any near-degeneracies. For control of the iterative procedure, you can set a limit on the number of iterations in the Maximum TDHF iterations text box,[78] and set thresholds on the energy[79] and coefficients (residual).[80]

When excited states are calculated, a spectrum file is written that contains the UV/visible spectral data, *jobname*`_uvvis.spm`. This file can be read into the Spectrum Plot panel in Maestro to generate a simulated UV/visible spectrum. The Spectrum Plot panel is described in Section 3.12 on page 80.

3.5 Local MP2 Settings

The LMP2 (Local MP2) option[81] of the Level of theory option menu allows you to set up a local Møller-Plesset second-order perturbation theory [88–91] calculation. The local MP2 (LMP2) method greatly reduces the basis set superposition errors that can arise from the canonical MP2 method [91]. The LMP2 method is much faster than canonical MP2, and typically recovers 98% of the canonical MP2 energy correction. The pseudospectral implementation of LMP2 is described in Section 8.2 on page 185.

73. Keyword **itddft** = 1 in the **gen** section.
74. Keyword **itda** = 1 in the **gen** section.
75. Keyword **rsinglet** in the **gen** section.
76. Keyword **rtriplet** in the **gen** section.
77. Keyword **nroot** in the **gen** section.
78. Keyword **maxciit** in the **gen** section.
79. Keyword **econvci** in the **gen** section.
80. Keyword **dconvci** in the **gen** section.
81. Keyword **mp2** = 3 in the **gen** section.



Level of theory: LMP2 (Local MP2)

LMP2

Core localization method: None

Valence localization method: Pipek-Mezey

Resonance: None

LMP2 Pairs:

[✓ All atom pairs] Hetero atom pairs

Job name: jaguar_1

Jaguar: LMP2 / 6-31G**, Single Point Energy, Host=localhost, Incorporate=Append new entries

Figure 3.3. The Theory tab showing LMP2 controls.

For closed-shell systems, you can perform LMP2 geometry optimizations, charge fitting, solvation calculations, and many other options available with HF wave functions. For open-shell systems, you can perform gas-phase energy and geometry optimizations but not property calculations. All local MP2 geometry optimizations employ analytic gradients.

For calculations of LMP2 dipole moments, Jaguar computes a coupled perturbed Hartree-Fock (CPHF) wave function, which can be computationally expensive. However, since CPHF methods lead to a better description of the charge density, we recommend computing LMP2 dipole moments as well for any calculation for which you need to compute accurate LMP2 electrostatic potential (ESP) fitted charges. For details, see Section 3.9.1 on page 62 and Section 3.9.3 on page 65.

Jaguar's implementation of the local MP2 method requires basis sets that allow the pseudospectral method to be used. This basis set information can be found in Section 3.2 on page 33. A warning is displayed if you choose a non-pseudospectral calculation.

The reference wave function is produced through a localization of the usual Hartree-Fock reference wave function, using a unitary transformation of the occupied canonical Hartree-Fock orbitals. (This localization procedure does *not* change the reference energy.) The default localization procedure is Pipek-Mezey [93] localization,[82] but you can choose Boys [92] localization[83] or an alternate Pipek-Mezey localization[84] (see Section 9.5.6 on page 206). These schemes are available from the Valence localization method option menu.

In LMP2, unlike in canonical MP2, the correlating virtual space for each occupied orbital is limited to those orbitals that are localized on the atoms of the local occupied Hartree-Fock orbital. The localization of the occupied orbitals makes this limitation of the virtual space a good approximation, and leads to a reduction in the basis set superposition error. In the limit that all local virtual orbitals are assigned to every occupied orbital, the local MP2 method and the canonical MP2 method are exactly equivalent.

All calculation types available for LMP2 wave functions are also available with the "local local" MP2 method, which allows you to treat only some atoms at the LMP2 level, while the remaining atoms are treated at the HF level. Local local MP2 calculations use orbitals that are localized on the specified atom pairs. After the localization of the canonical Hartree-Fock orbitals, the atomic orbital coefficients for each localized orbital are summed for each atom, and the orbital is considered localized on the two atoms whose coefficient sums are largest. If the largest coefficient sum on one atom is more than ten times as large as the coefficient sum on any other atom, the localized orbital is considered to be localized on that single atom, and that localized orbital will be included in any LMP2 calculation for which that atom is specified in any requested LMP2 atom pairs.

Jaguar includes a setting for a local LMP2 calculation that treats all atoms bonded to atoms of other elements—"heteroatom pairs"—at the LMP2 level. These heteroatom pairs do not include C atoms bonded only to C and H atoms, so hydrocarbon fragments are not correlated. We recommend this setting for solvation calculations using LMP2. To request such a calculation, select Hetero atom pairs[85] in the LMP2 pairs section. For other kinds of local LMP2 calculations, you must set up the pairs in an **lmp2** section by editing the input file. See Section 3.5 on page 50 for more information. If you add an **lmp2** section, the Level of theory is set to Other.

Correlation of orbitals on atom pairs in a virtual space that is restricted to those atoms is not the optimal choice when the orbitals are delocalized, as they are in aromatic systems such as benzene. Such delocalized orbitals are represented in valence bond theory by resonance structures. To handle aromatic systems, you can delocalize the LMP2 pairs over neighboring atoms

82. Keyword **loclmp2v** = 2 in the **gen** section.
83. Keyword **loclmp2v** = 1 in the **gen** section.
84. Keyword **loclmp2v** = 3 in the **gen** section.
85. Keywords **iheter** = 1 in the **gen** section.

in the ring (partial delocalization),[86] or over the whole ring (full delocalization).[87] These options are available from the Resonance menu.

3.6 Hamiltonian Settings

For heavy elements, it is necessary to include relativistic effects in the Hamiltonian. In Jaguar there are two ways of doing this: using ECPs, and with an explicit relativistic Hamiltonian. ECPs are handled via the basis set selection. The Hamiltonian option menu, at the bottom of the Theory tab, allows you to choose the Hamiltonian. The default is Nonrelativistic, which is the usual Schrödinger Hamiltonian. The other choices are for relativistic Hamiltonians, and include (relativistic) in the menu item, to indicate that they include relativistic effects.

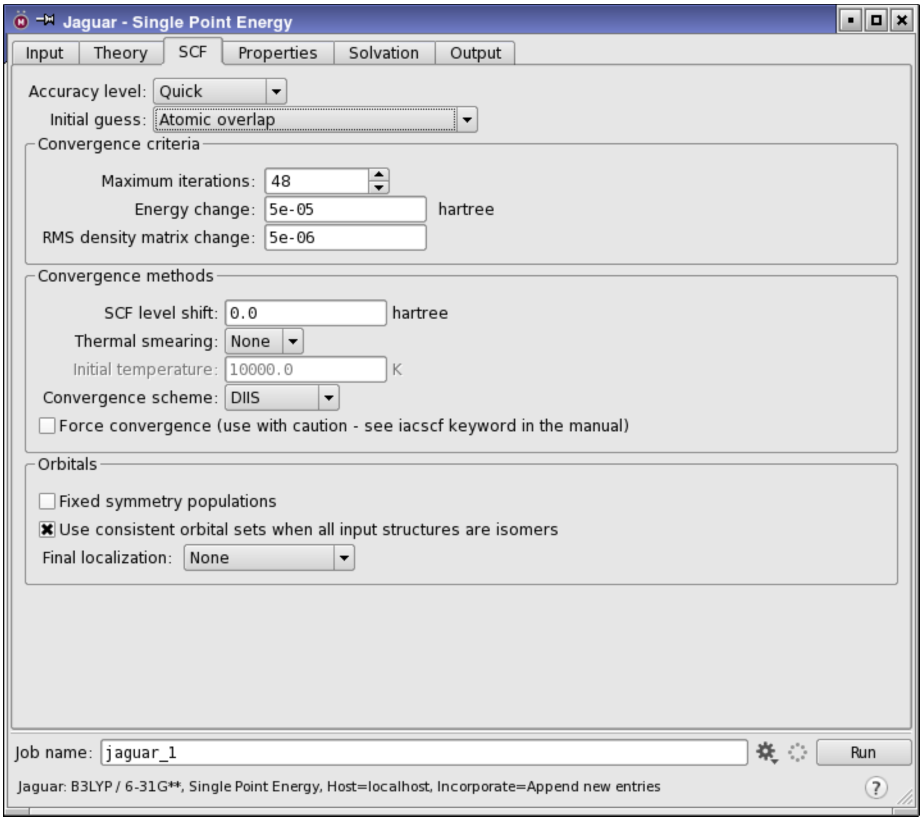
- Nonrelativistic—Use the (nonrelativistic) Schrödinger Hamiltonian. This is the default, and should be used with ECPs. If you choose LMP2 for the level of theory, this Hamiltonian is selected, as LMP2 cannot currently be used with ZORA.
- Scalar ZORA—Use the scalar ZORA Hamiltonian[88], with local evalution of the integrals[89] (see Section 8.4 on page 190 for details). This option can be used for HF and DFT calculations, but not for LMP2 calculations. If you choose this Hamiltonian for LMP2, the Level of theory is reset to DFT. You can calculate energies and properties and perform geometry optimizations. The ZORA integrals are only evaluated for elements with Z > 18; lighter atoms are treated nonrelativistically.

  It is recommended that you use one of the basis sets that is contracted for ZORA calculations for the heavy atoms (see Table 3.1). These basis sets have “zora” in the name. The dyall-v2z_zora-j-pt-gen basis set is a double-zeta polarized basis set similar to cc-pvdz; the dyall-2zcvp_zora-j-pt-gen basis set adds outer core polarization. Both basis sets are generally contracted, and cover elements up to Rn. The sarc-zora basis set is also of double-zeta quality, and is a segmented contraction, covering elements from La to Rn.

3.7 SCF Settings

The SCF tab includes various settings that control the type of calculation and how the calculation is performed, including the accuracy level, the convergence method and criteria, and the orbital symmetry and localization.

86. Keyword **ireson** = 1 in the **gen** section.
87. Keyword **ireson** = 2 in the **gen** section.
88. Keyword **relham** = zora_scalar in the **gen** section.
89. Keyword **potential** = local in the **relativity** section.


Jaguar - Single Point Energy

Input | Theory | SCF | Properties | Solvation | Output

Accuracy level: Quick

Initial guess: Atomic overlap

Convergence criteria

Maximum iterations: 48

Energy change: 5e-05 hartree

RMS density matrix change: 5e-06

Convergence methods

SCF level shift: 0.0 hartree

Thermal smearing: None

Initial temperature: 10000.0 K

Convergence scheme: DIIS

☐ Force convergence (use with caution - see iacscf keyword in the manual)

Orbitals

☐ Fixed symmetry populations

☒ Use consistent orbital sets when all input structures are isomers

Final localization: None

Job name: jaguar_1

Run

Jaguar: B3LYP / 6-31G**, Single Point Energy, Host=localhost, Incorporate=Append new entries

Figure 3.4. The SCF tab.

3.7.1 Accuracy Level

Jaguar can use integrals evaluated on a grid with the pseudospectral method and fully analytic integrals. The grids used for various SCF iterations and the accuracy with which parts of the calculation are done greatly affect the timing, and sometimes the accuracy, of the entire calculation. You can adjust the grids and the set of cutoff values determining these factors using the Accuracy level option menu. For more information on grids and cutoffs, see Section 10.4 on page 305 and Section 10.5 on page 308.

The default Quick setting[90] allows fast calculations to be performed, using several different pseudospectral grid types, and cutoffs that should generally produce well-converged energies. The Accurate setting,[91] which corresponds to tighter cutoffs (and therefore somewhat slower calculations), also uses a variety of pseudospectral grids. If you choose the Ultrafine setting,[92]

90. Keyword **iacc** = 3 in the **gen** section.
91. Keyword **iacc** = 2 in the **gen** section.

the cutoffs are even tighter (very accurate), and only the ultrafine pseudospectral grid type is used. The Ultrafine setting may be helpful for cases with convergence or accuracy problems, but increases the computational cost by a factor of two to three. The remaining choice is Fully analytic,[93] which turns off the pseudospectral method and uses the analytic methods for all integrals [173,174]. This choice is significantly slower than the pseudospectral method. However, it can be run in parallel with OpenMP threads—see Section 11.4.4 on page 337.

3.7.2 Initial Guess

Jaguar has several ways of providing an initial guess of the molecular orbitals. If you read in a Jaguar restart file (`.01.in`), the initial guess is taken from the **guess** section in this file. When starting a new calculation, you can choose a method for generating an initial guess from the Initial guess option menu, whose items are listed below.

- Atomic overlap[94]—Construct a guess for the molecular orbitals from atomic orbitals. The core molecular orbitals are taken as the atomic core orbitals. For the valence molecular orbitals, the overlap matrix is diagonalized in the space of the valence atomic orbitals, and the eigenvectors with the largest eigenvalues are taken for the valence molecular orbitals. The entire set is orthogonalized, core first, then valence. This is the default.
- Atomic density[95]—Construct an approximation to the molecular density from a superposition of atomic densities. The density is projected onto the AO basis. The resulting matrix is diagonalized to give natural orbitals, which are used as the initial guess orbitals.
- Core Hamiltonian[96]—Generate an initial guess to the molecular orbitals by diagonalizing the one-electron Hamiltonian matrix. This is rarely a good guess, as the valence orbitals are usually too tight.
- Ligand field theory[97]—For transition metal complexes, use simple ligand field theory to construct an initial guess for the metal d orbitals. An effective Hamiltonian is diagonalized, taking into account the assigned formal charges on the metal and the ligand and the occupation of the ligand orbitals, to determine the d-orbitals and the orbital ordering. The core and ligand orbitals are determined using the atomic overlap guess.
- Ligand field theory with d-d repulsion[98]—For transition metal complexes, use ligand field theory including d-d repulsion to construct an initial guess. The procedure is the same as above, except that repulsion between the d electrons on the metals is included to determine the orbital ordering and hence the occupation of the d orbitals and the spin state.

92. Keyword **iacc** = 1 in the **gen** section.
93. Keyword **nops** = 1 in the **gen** section.
94. Keyword **iguess** = 10 in the **gen** section.
95. Keyword **iguess** = 11 in the **gen** section.
96. Keyword **iguess** = 0 in the **gen** section.
97. Keyword **iguess** = 25in the **gen** section.
98. Keyword **iguess** = 30 in the **gen** section.

Further information on the initial guess is given in Section 7.1 on page 167.

3.7.3 Convergence Criteria

SCF calculations finish when the calculations converge, the maximum number of iterations is exceeded, or Jaguar determines that the calculation cannot converge. The maximum number of iterations and the convergence criteria are set in the Convergence criteria section.

You can set the maximum number of SCF iterations in the Maximum iterations text box.[99] Generally, Hartree-Fock calculations for simple organic molecules converge in fewer than 10 iterations, while complex calculations using higher-level methods or involving open shells can take a few extra iterations. Molecules that include transition metals can converge more slowly. The absolute maximum number of SCF iterations is 5000.

The default energy convergence criterion,[100] which can be set in the Energy change text box, is $5.0{\times}10^{-5}$ Hartrees. When the change in total energy on consecutive iterations is less than this value, the energy is converged. For polarizability calculations the default is $1.0{\times}10^{-6}$ Hartrees. If the energy difference is less than 1% of the previous energy difference, however, this convergence criterion is overridden for that iteration and the calculation continues.

The default density convergence criterion,[101] which can be set in the RMS density matrix change text box, is $5.0{\times}10^{-6}$. When the RMS difference in density matrix elements between two iterations is less than this value, the density is converged. For polarizability and hyperpolarizability calculations, if the energy convergence criterion described in the previous paragraph is satisfied first, the calculation ends even if the RMS density matrix element change criterion has not been met, and vice versa.

3.7.4 Convergence Methods

Jaguar provides a range of options for controlling the convergence of SCF calculations. These options are available in the Convergence methods section.

Level shifting adds a constant to the diagonal of the Fock matrix for the virtual orbitals before diagonalization. This reduces the mixing of the occupied and virtual orbitals, and damps the changes in the orbitals. As a result, the convergence is slower, but this method often helps otherwise intractable cases to converge. Level shifting is normally varied during the course of an SCF calculation without having to explicitly set a value. Useful SCF level shift values are generally in the range 0.3–1.0. You can set the constant in the SCF Level shift text box.[102]

99. Keyword **maxit** in the **gen** section.
100. Keyword **econv** in the **gen** section.
101. Keyword **dconv** in the **gen** section.
102. Keyword **vshift** in the **gen** section.

Thermal smearing populates the virtual orbitals using a Fermi-Dirac distribution with a given temperature, allowing orbitals other than the default orbitals to be populated. The temperature is progressively reduced so that the true occupied orbitals are the only ones populated at the end of the calculation. This procedure makes it easier for orbitals to switch if they are not in the default initial guess but are in the converged wave function. To use thermal smearing, choose a method from the Thermal smearing option menu[103] and enter a value in the Initial temperature text box.[104] More information on this method can be found on page 250.

Finally, you can attempt to force convergence by selecting the Force convergence option.[105] This option starts with a large level shift and adjusts the value during the calculation, and runs at ultrafine accuracy level. Use this option with caution.

To speed up convergence, Jaguar offers some convergence acceleration schemes, which are available from the Convergence scheme option menu. We recommend using the default Direct Inversion in the Iterative Subspace (DIIS)[106] or GVB-DIIS[107] SCF convergence schemes [11, 172] whenever possible. The DIIS method generally performs better, but for jobs with SCF convergence problems, GVB-DIIS may give improved convergence. The DIIS method can be used with any wave function, including those with multiple open shells. You can also use the OCBSE convergence scheme[108] [22], although it is generally much slower than DIIS.

3.7.5 Orbital Treatment

Convergence difficulties can be encountered when orbitals of different symmetries swap. You can fix the population in each symmetry by selecting Fixed symmetry populations.[109] This option is useful when you are running calculations on different occupations of degenerate d orbitals, or if you want to converge on a state that is not the ground state but can be distinguished by the occupations of orbitals of different symmetries.

When running calculations on a set of isomeric structures, the results may depend on the number of canonical orbitals used for each isomer. If they are different, the results are not strictly comparable. You can enforce the same number of canonical orbitals for each isomer by selecting Use consistent orbital sets when all input structures are isomers. An initial one-electron integral calculation is run on each isomer to determine the number of canonical orbitals, and the minimum number is selected and enforced for all subsequent steps of the job.

103. Keyword **ifdtherm** in the **gen** section.
104. Keyword **fdtemp** in the **gen** section.
105. Keyword **iacscf**=1 in the **gen** section.
106. Keyword **iconv** = 1 in the **gen** section.
107. Keyword **iconv** = 4 in the **gen** section.
108. Keyword **iconv** = 3 in the **gen** section.
109. Keyword **ipopsym**=1 in the **gen** section.

By default, the final wave function is not localized.[110] You can localize the valence orbitals after the wave function is computed with either the Boys[111] procedure [92] or the Pipek-Mezey[112] procedure [93], by choosing from the Final localization option menu. The Boys procedure localizes the doubly-occupied orbitals by maximizing the term $\Sigma_{ij}|\langle\varphi_i|r|\varphi_i\rangle - \langle\varphi_j|r|\varphi_j\rangle|^2$. Pipek-Mezey localization is performed by maximizing the sum of the squares of the atomic Mulliken populations for each atom and occupied orbital. See Section 3.13.2 on page 86 to find out how to print the localized orbitals.

Both of the available localization methods scale as N^3 with basis set size. However, the use of molecular symmetry is turned off for the entire job whenever you perform a final localization, so for faster results you might want to run a job without localization, then restart the job after turning on localization in the new input file. See Section 7.5 on page 174 for information on restart files and restarting jobs.

3.8 Solvation Settings

Jaguar can treat solvated molecular systems with a self-consistent reaction field method, using either a standard Poisson-Boltzmann solver [15, 211] or the Solvation Model 8 (SM8) [212] or Solvation Model 6 (SM6) approach [220]. These models are treated in the subsections below.

3.8.1 Poisson-Boltzmann Solvation Model

With the standard Poisson-Boltzmann solver[113], you can compute solvation free energies and minimum-energy solvated structures or solvated transition states. The solvation free energy from a geometry optimization is computed as the difference between the energy of the optimized gas phase structure and the energy of the optimized solvated structure.Jaguar first calculates the usual gas phase wave function and from that the electrostatic potential, and fits that potential to a set of atomic charges, as described in Section 3.9.1 on page 62.

These charges are passed to the Poisson-Boltzmann solver, which then determines the reaction field by numerical solution of the Poisson-Boltzmann equations and represents the solvent as a layer of charges at the molecular surface (which serves as a dielectric continuum boundary). These solvent point charges are returned to Jaguar's SCF program, which performs another quantum mechanical wave function calculation, incorporating the solvent charges. This process is repeated until self-consistency is obtained. The cost is roughly twice that of a gas phase calculation.

110. Keyword **locpostv** = 0 in the **gen** section.
111. Keyword **locpostv** = 1 in the **gen** section.
112. Keyword **locpostv** = 2 in the **gen** section.
113. Keyword **isolv** = 2 in the **gen** section.

Solvation free energies can be computed with HF, DFT, LMP2, CIS, and TDDFT wave functions. For local LMP2 solvation free energy calculations, we recommend using heteroatom pairs for the most efficient results, particularly since solvation free energy calculations often use radii optimized for calculations that use heteroatom pairs. See Section 10.6 on page 311 for more details; see Section 3.5 on page 50 for information on setting LMP2 pairs.

In addition to calculating solvated free energies, you can also calculate properties analytically for solvated species. The CPHF equations are solved in the reaction field to produced solvated perturbed wave functions, which are then used to calculate the properties.

Solvent parameters are set in the Solvation tab. To select the standard Poisson-Boltzmann model, choose PBF from the Solvent model option menu. You can choose the solvent from the Solvent option menu, and Jaguar performs a solvation calculation, setting the appropriate dielectric constant[114] and probe radius.[115] The dielectric constant [99] and probe radius [100] values set by Jaguar for various solvents are shown in Table 3.3.

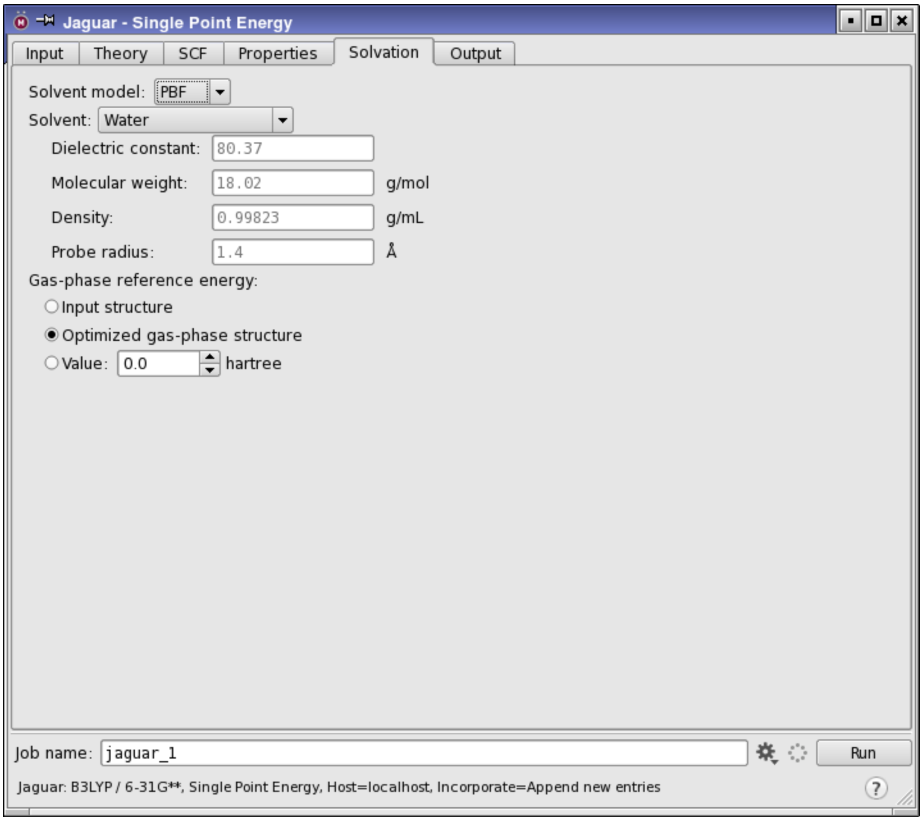
Table 3.3. Parameters for various solvents

Solvent	Dielectric Constant	Probe Radius
1,2-dichloroethane	10.65	2.51
acetonitrile	37.5	2.19
benzene	2.284	2.60
carbon tetrachloride	2.238	2.67
chlorobenzene	5.708	2.72
chloroform	4.806	2.52
cyclohexane	2.023	2.78
dichloromethane	8.93	2.33
dimethyl sulfoxide (DMSO)	47.24	2.41
dimethylformamide	36.7	2.49
ethanol	24.85	2.27
methanol	33.62	2.00
nitrobenzene	35.74	2.73
tetrahydrofuran	7.6	2.52
water	80.37	1.40

114. Keyword **epsout** in the **gen** section.
115. Keyword **radprb** in the **gen** section.

To use a solvent that is not on this option menu, you can define it by choosing Other and changing the entries for Dielectric constant, Molecular weight, and Density. The latter two values are used to calculate the probe radius (in angstroms; see reference 100).

If you compute the solvation free energy of a minimum-energy or transition-state structure optimized in solution, your calculation should compare the energy of the optimized solvated structure to the energy of the optimized gas-phase structure. Therefore, by default, geometry optimizations in solution are performed only after an optimized gas-phase structure is computed.[116] However, if you want only an optimized *structure* in solution and do not care about the computed solvation free energy, you can skip the gas phase geometry optimization by selecting Input structure[117] or Value[118] under Gas phase reference energy. These options allow you to use a reference energy other than that of the optimized gas phase structure.



Solvent model: PBF
Solvent: Water
Dielectric constant: 80.37
Molecular weight: 18.02 g/mol
Density: 0.99823 g/mL
Probe radius: 1.4 Å
Gas-phase reference energy:
○ Input structure
● Optimized gas-phase structure
○ Value: 0.0 hartree

Job name: jaguar_1

Jaguar: B3LYP / 6-31G**, Single Point Energy, Host=localhost, Incorporate=Append new entries

Figure 3.5. The Solvation tab.

116. Keyword **nogas** = 0 in the **gen** section.
117. Keyword **nogas** = 2 in the **gen** section.
118. Keyword **nogas** = 1 in the **gen** section.

3.8.2 Solvation Models 8 and 6 (SM8 and SM6)

The SM8 model[119] is the latest in the series of SM*x* continuum (or implicit) solvation models developed at the University of Minnesota. This model is a universal model that is applicable to any charged or uncharged solute in any solvent or liquid medium for which a few key descriptors are known, in particular the dielectric constant, refractive index, macroscopic surface tension, and acidity and basicity parameters. The details of SM8 are described in a recent publication [212]. The earlier SM6 model [220][120] is restricted to water as the solvent, and is deprecated in favor of SM8. Both models are available in Jaguar.

The SM8 and SM6 models can be used for computing the standard-state free energy of solvation at the gas-phase geometry. Solution-phase geometry optimizations with analytical gradients cannot be carried out using the current implementation in Jaguar. When using these models, Jaguar performs a gas-phase self-consistent field calculation followed by a solution-phase SCRF calculation.The cost of carrying out a solution-phase SCRF calculation is roughly the same as that of a gas-phase SCF calculation.

The SM8 and SM6 models separate the standard-state free energy of solvation into two components:

- the bulk electrostatic component arising from a self-consistent reaction field (SCRF) treatment using the generalized Born approximation for electrostatics [213–216] based on self-consistently polarized partial atomic charges. This component is long-range.
- the cavity-dispersion-solvent-structure (CDS) contribution [217–219] arising from short-range interactions between the solute and nearby solvent molecules. This component is restricted to the first solvation shell.

All SM8 and SM6 calculations in Jaguar correspond to a temperature of 298 K and a standard-state concentration of 1 M in both the gas and solution phases. To convert the computed solvation free energy to correspond to a gas-phase standard state of 1 atm, add 1.89 kcal/mol.

The SM8 and SM6 models can be used only with the Hartree-Fock approximation or density functional theory. The implementation of the SM8 model in Jaguar uses self-consistently polarized CM4M class IV charges when used with any member of the M06 suite of density functionals, and it uses CM4 class IV charges when used with any other density functional or with the Hartree-Fock approximation. The supported basis sets for which the charge parameters have been incorporated in Jaguar are:

- MIDIX
- 6-31G*
- 6-31G**

119. Keyword **isolv** = 6 in the **gen** section.
120. Keyword **isolv** = 5 in the **gen** section.

- 6-31+G*
- 6-31+G**
- cc-pVDZ (SM8 only)

The use of basis sets not in this list is not recommended because the SM8 model will employ Löwdin charges, which typically produce less accurate solvation free energies than do class IV charges. The charge parameters for the basis sets supported by the SM8 model are given in published references [220, 221].

To run an SM8 or SM6 solvation calculation, choose SM8 or SM6 from the Solvent model option menu in the Solvation tab. For SM8, choose a solvent from the Solvent option menu. For SM6, the solvent is automatically set to Water (the only choice available with SM6). If you use the MIDIX basis set, you must also select 6D for the Number of D functions in the Molecule tab, or set `numd=6` in the **gen** section.

If you want to use other solvents with SM8, you can do so by setting the solvent keyword in the input file to one of the accepted solvents, or you can add your own solvent parameters. The list of solvents and keywords for custom solvents are given in Section 9.5.13 on page 230.

3.9 Properties

For each kind of wave function, Jaguar can calculate various molecular properties. The range of available properties depends on the wave function. These calculations are normally performed using the converged SCF wave function. By default, none of the properties are computed, but you can compute them by changing the settings in the Properties tab. The exception is that SM6 solvation calculations produce some population and charge properties by default. These are described in Section 3.9.9 on page 68. Properties can be computed for molecules that are solvated with the Poisson-Boltzmann solvation model.

The Properties tab has a table of available properties, and an area below the table where controls for a property are displayed when you select the table row for the property. To include the calculation of a property in a job, select the check box in the Calculate column.

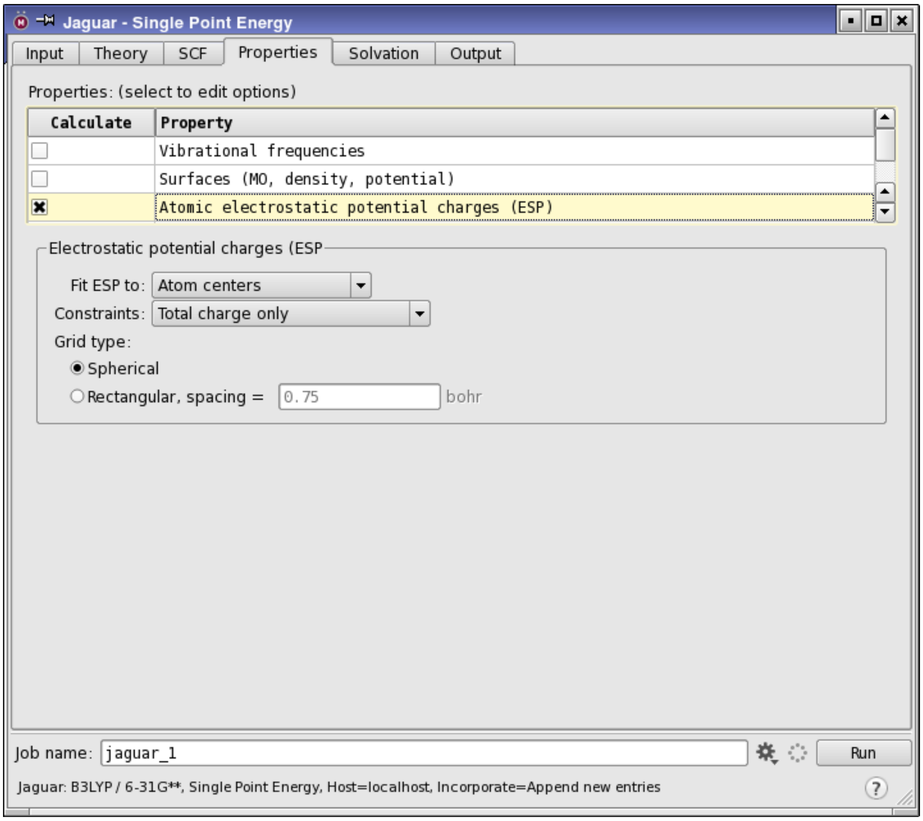
Vibrational frequencies (and related properties) and surfaces are discussed in later sections. This section focuses on the remaining properties.

3.9.1 Charges from Electrostatic Potential Fitting

Jaguar can fit a set of point charges to best reproduce the molecular electrostatic potential (ESP) as represented on a grid [102, 103]. These monopoles can be located either at the atomic centers[121] or at the atomic centers and the bond midpoints,[122] depending on the selection from

121. Keyword **icfit** = 1 in the **gen** section.
122. Keyword **icfit** = 2 in the **gen** section.

the Fit ESP to option menu. The atomic charges are written to the output Maestro (`.mae`) structure file and are available in Maestro as the partial charge. These charges can then be used in other applications, such as MacroModel or QikProp.



Properties: (select to edit options)

Calculate	Property
☐	Vibrational frequencies
☐	Surfaces (MO, density, potential)
☒	Atomic electrostatic potential charges (ESP)

Electrostatic potential charges (ESP

Fit ESP to: Atom centers

Constraints: Total charge only

Grid type:

● Spherical

○ Rectangular, spacing = 0.75 bohr

Job name: jaguar_1

Jaguar: B3LYP / 6-31G**, Single Point Energy, Host=localhost, Incorporate=Append new entries

Figure 3.6. The Properties tab showing controls for ESP charges.

For electrostatic potential fitting of an LMP2 wave function, you should also compute a dipole moment for more accurate results, since the charge fitting will then include a coupled perturbed Hartree-Fock (CPHF) term as well. You might also want to constrain the charge fitting to reproduce the dipole moment, as described below. Because the CPHF term is computationally expensive, it is not included in LMP2 charge fitting by default.

The fit can be constrained to reproduce the dipole moment (and other higher moments, if specified) exactly, by choosing the combination of moments from the Constraints option menu.[123] (For LMP2 wave functions, only dipole moments are available.) Keep in mind that the more constraints you apply to electrostatic potential fitting, the less accurately the charge fitting will

123.Keyword **incdip** in the **gen** section.

describe the Coulomb field around the molecule. The dipole moment from fitting charges only is generally very close to the quantum mechanical dipole moment as calculated from the wave function. Constraining the charge fitting to reproduce the dipole moment is generally not a problem, but you might obtain poor results if you constrain the fitting to reproduce higher multipole moments. However, this option is useful for cases such as molecules with no net charge or dipole moment.

If both electrostatic potential fitting and multipole moment calculations are performed, the moments are also computed from the fitted charges for purposes of comparison.

The electrostatic potential is itself computed on a grid. By default, this grid has the same form as the other pseudospectral grids: it is formed by merging sets of spherical shells, whose grid points are centered on each nucleus. The Spherical option is selected by default in the Grid type section.[124] An alternative is to use a regular lattice of grid points [103], by choosing Rectangular in the Grid type section.[125] You can then set the spacing in bohr between points in this lattice in the text box.[126] For either grid type, points within the molecular van der Waals surface are discarded. The van der Waals surface used for this purpose is constructed using DREIDING [104] van der Waals radii for hydrogen and for carbon through argon, and universal force field [101] van der Waals radii for all other elements. These radii are listed in Table 9.45. The radius settings can be altered by making **vdw** settings in the **atomic** section of an input file, as described in Section 9.8 on page 273.

You can also print out the values of the electrostatic potential at grid points whose locations you specify—see Section 9.5.15 on page 236. By default, the values at the nuclei are added as atom properties to the output Maestro file, and printed in the output. Values at ECP centers should be disregarded, as they do not include contributions from the core electrons (or the corresponding "core" part of the nuclear potential). The values of the electrostatic potential at the nuclei can be correlated with various molecular properties [245,246].

3.9.2 Mulliken Population Analysis

Mulliken populations [105] can be computed for each atom, giving a representation of the molecule as a set of nuclear-centered point charges.[127] For open shell cases, Mulliken spin populations are also computed when Mulliken populations are requested. If you choose to calculate both Mulliken populations and multipole moments, the multipole moments are computed from the atomic Mulliken populations as well as from the wave function.

124. Keyword **gcharge** = –1 in the **gen** section.
125. Keyword **gcharge** = –2 in the **gen** section.
126. Keyword **wispc** in the **gen** section.
127. Keyword **mulken** = 1 in the **gen** section.

Mulliken populations can be computed for each basis function as well as for each atom,[128] or for each bond between neighboring atoms, as well as by atom and basis function,[129] by choosing the appropriate option in the Method section.

3.9.3 Multipole Moments

Jaguar can compute multipole moments through hexadecapole for HF or DFT wave functions, and can compute dipole moments for LMP2 wave functions. Moments are computed with respect to the center of mass of the molecule. Note that LMP2 dipole moments can be computationally expensive, since computing them accurately requires coupled perturbed Hartree-Fock calculations.

By default, all moments are calculated. You can restrict the order of the moments with the keyword **ldips** in the **gen** section. If you select one of the higher-order moments, all moments of lower order are also calculated. If atomic charges are computed either by fitting of the electrostatic potential [102, 103] or by Mulliken population analysis [105], the multipole moments are also calculated from these point charges for comparison.

The dipole moment and its cartesian components are written to the output structure file (`.mae`) and incorporated as properties in the Project Table. It can be displayed in the Workspace by choosing Workspace → Dipole Moment. The dipole moment is displayed as an arrow, with the base of the arrow at the coordinate origin.

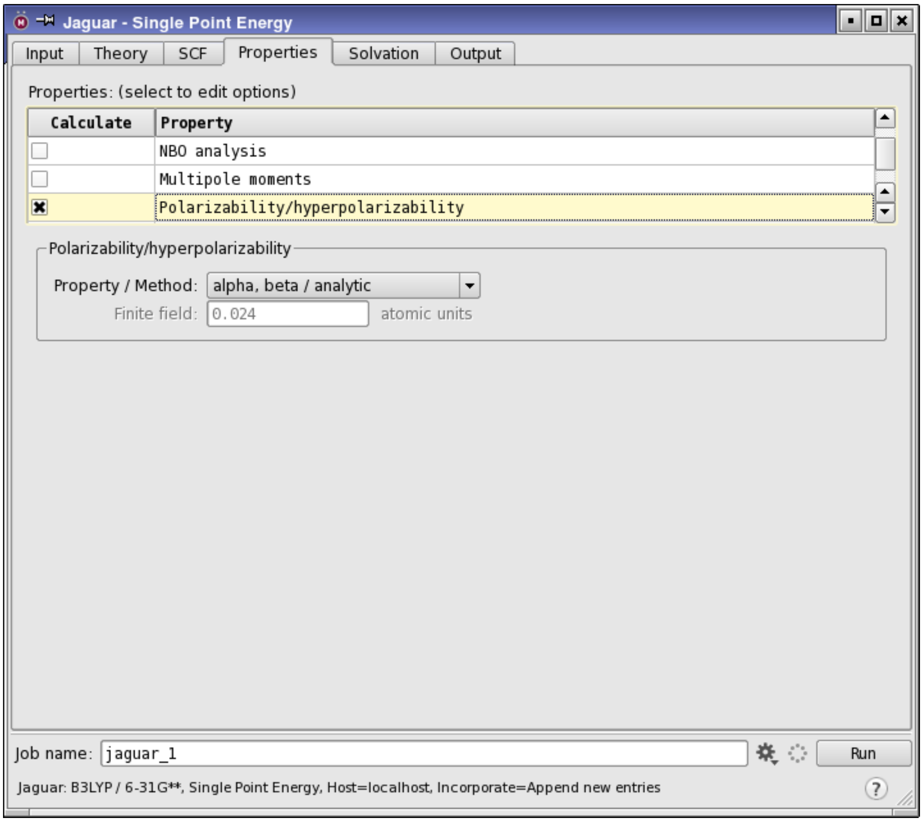
3.9.4 Natural Bond Orbital (NBO) Analysis

When you select NBO analysis[130] a Natural Bond Orbital (NBO) analysis is performed at the end of the Jaguar job. The output from the NBO analysis is included in the Jaguar output file. By default, the NBO analysis is run with the `NRT` and `PRINT=3` NBO settings, so that the output includes both NBO and Wiberg bond orders, the natural localized molecular orbitals (NLMO), analysis of the molecular dipole in terms of both NBOs and NLMOs, and natural population analysis (NPA). You can control the NBO program by adding an **nbo** section to your Jaguar input file (either via the Edit panel in Maestro or with a text editor), and including NBO directives within it. You can also add **core**, **choose**, and **nrtstr** sections to the Jaguar input file for additional control over the NBO program. See the *NBO 6.0 Manual*, or visit the NBO web site, http://nbo6.chem.wisc.edu/. If present, the **nbo** section's content overrides that implied by the `nbo` keyword. Thus, if you set `nbo=1` and add an empty **nbo** section, you will get just the basic NBO analysis without the `NRT` or `PRINT=3` output. It is not possible to run NEDA (Natural Energy Decomposition Analysis) calculations from Jaguar.

128. Keyword **mulken** = 2 in the **gen** section.
129. Keyword **mulken** = 3 in the **gen** section.
130. Keyword `nbo` = 1 in the **gen** section.

3.9.5 Polarizability and Hyperpolarizability

You can calculate polarizabilities and first and second hyperpolarizabilities by selecting Polarizability/Hyperpolarizability in the Properties table and making the appropriate choice from the Property/Method option menu. The calculations can be done analytically or with a finite field.



Properties: (select to edit options)

Calculate	Property
☐	NBO analysis
☐	Multipole moments
☒	Polarizability/hyperpolarizability

Polarizability/hyperpolarizability

Property / Method: alpha, beta / analytic

Finite field: 0.024 atomic units

Job name: jaguar_1

Jaguar: B3LYP / 6-31G**, Single Point Energy, Host=localhost, Incorporate=Append new entries

Figure 3.7. The Properties tab showing controls for polarizabilities.

The analytic method [18] calculates the polarizability and hyperpolarizabilities by solving the coupled perturbed Hartree-Fock (CPHF) equations.[131] In general, this method is superior to the finite field method, but the CPHF option can be used only with closed-shell and unrestricted open-shell wave functions. The analytic method is supported for molecules in the gas phase and in solution.

131. Keyword **ipolar** = –1 and –2 in the **gen** section.

The options for the finite field method [17] can use a 3-point,[132] 5-point,[133] or 7-point[134] finite difference method, which uses the results from a number of SCF calculations: one with no field and several with electric fields that are multiples of +E and –E in the *x*, *y*, and *z* directions. E is 0.024 au by default. You can use a different value by entering it in the Finite field text box.

Both hyperpolarizability methods are run without using molecular symmetry. Also, for any polarizability calculation, the energy convergence criterion, which is set in the SCF tab, is set by default to 1.0×10^{-6}.

For more information see Section 6.3.7 on page 149 and Section 9.5.15 on page 236.

3.9.6 NMR Shielding Constants

Gas-phase and solution-phase NMR shielding constants are available for closed-shell and unrestricted open-shell wave functions [19].[135] Shielding constants are returned as atom properties (in ppm) in the Maestro output file. You can use these values for atom selection, for example, or you can display them in labels.

Shieldings are calculated for all atoms, including those with ECPs. Shielding constants for atoms whose core is represented by an ECP should be ignored as erroneous, because the main contributions come from the core tail of the valence orbitals, which is largely absent at ECP centers. Chemical shifts derived from these shielding constants might display the correct trends, but are likely to have the wrong magnitude.

Chemical shifts are calculated for C and H atoms from the shieldings, using a linear fit to experimental data, following refs. 251, 252. The shifts for C are corrected if Cl atoms are directly attached to the C atom. No corrections are available for other heavy atoms, so shifts for C bonded to atoms heavier than Cl should be treated with caution. The shifts are added to the structures as an atom property (NMR Chemical Shifts) in the output structure file.

The chemical shift calculations are available for the 6-31G, 6-311G, cc-pVDZ, MIDIX, LACVP, and LAV2P basis sets with or without polarization or diffuse functions, in combination with the following density functionals: M06-2X, M06-2X-D3, PBE, PBE0, BLYP, B3LYP, B3LYP-D3, O3LYP, B3PW91, PW6B95-D3, and B97-D3.

If you want to calculate chemical shifts directly, you will need to calculate NMR shielding constants for the reference molecules for each element of interest, in the same basis set and with the same method as for the molecule of interest.

132. Keyword **ipolar** = 1 (for alpha) or 2 (for alpha and beta) in the **gen** section.
133. Keyword **ipolar** = 5 in the **gen** section.
134. Keyword **ipolar** = 7 in the **gen** section.
135. Keyword **nmr** = 1 in the **gen** section.

3.9.7 Atomic Fukui Indices

Atomic Fukui indices, derived from Mulliken populations for the HOMO and LUMO orbitals, can be computed for both the electron density and the spin density. They are available for closed-shell and open-shell SCF (HF or DFT) wave functions. The indices are returned as atom properties in the Maestro output file. You can use these values to select atoms, or you can display them in labels, for example. For more information, see Section 6.3.7.5 on page 153.

These indices are not the same as Fukui functions (see Section 5.2 on page 117), nor are they derived from the explicit calculation of Fukui functions. As these indices are averaged properties, they might not show trends that are observable in the Fukui functions themselves.

3.9.8 Stockholder Charges

Stockholder charges are computed by a Hirshfeld partitioning of the electron density [237]. In this method, the atomic densities are evaluated on a grid, and are scaled at each point so that their sum is equal to the molecular density. The scaled densities are then integrated for each atom to produce an atomic charge. This method has less sensitivity to the basis set than Mulliken charges. The charges are returned as atom properties in the Maestro output file.

3.9.9 Molecular Properties from SM6 and SM8 Calculations

When an SM6 calculation is carried out in Jaguar, several additional molecular properties are automatically calculated because they are necessary for the SM6 calculation.

- Löwdin Population Analysis— Löwdin populations [222, 223] are computed for each atom in the gas phase and in the solution phase.
- Redistributed Löwdin Population Analysis—When basis sets containing diffuse functions (that is, 6-31+G* or 6-31+G**) are specified with SM6, redistributed Löwdin populations [224] are computed for each atom in the gas phase and in the solution phase.
- Mayer Bond Orders—Mayer bond orders [225–227] are computed between each pair of atoms in the gas phase and in the solution phase. To print the Mayer bond order matrix, you must specify **ip378**=2 in the **gen** section.
- CM4 Partial Atomic Charges—CM4 partial atomic charges [220] are computed for each atom in the gas phase and in the solution phase, according to

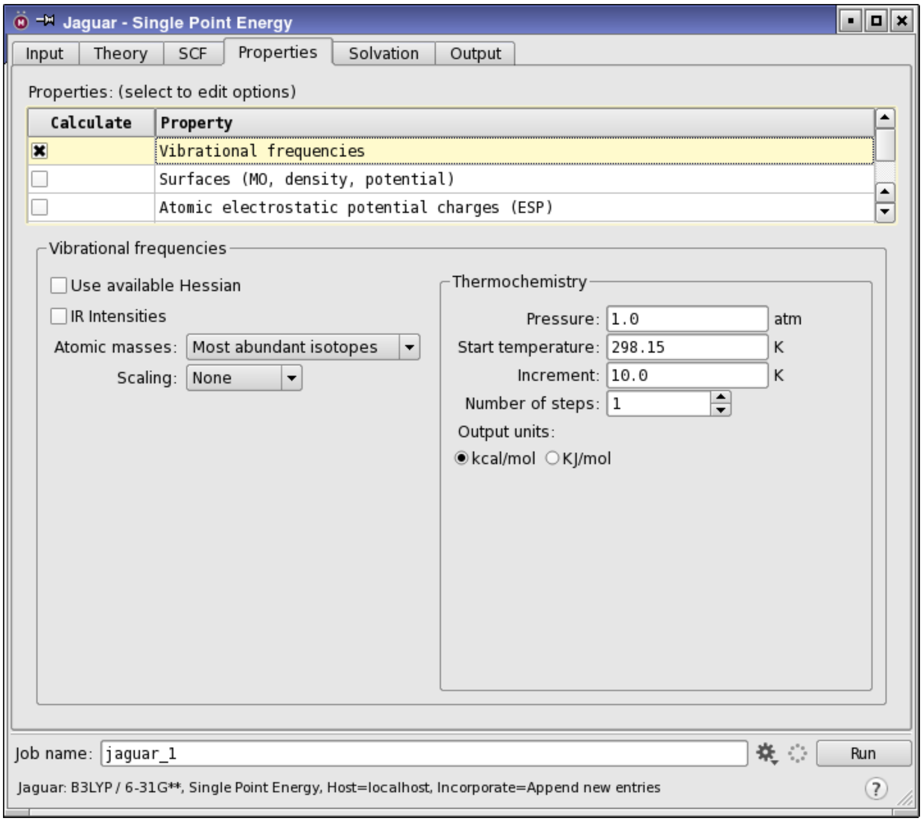
$$q_k^{\mathrm{CM4}} = q_0^k + \sum_{m \neq k} (D_{k,m} + C_{k,m} B_{k,m}) B_{k,m}$$

  where the summation goes over atoms m in the molecule, q_k^0 is the partial atomic charge from either a Löwdin population analysis (nondiffuse basis sets) or a redistributed

Löwdin population analysis (diffuse basis sets), $B_{k,m}$ is the Mayer bond order between k and m, and C and D are empirical parameters that depend on the basis set and on the atomic number of k and m.

3.10 Frequencies and Related Properties

Selecting Vibrational frequencies[136] in the Properties tab requests calculations of frequencies, infrared (ir) intensities, and thermochemical properties: heat capacity, entropy, enthalpy, and Gibbs free energy. The results of frequency calculations can be animated in Maestro.



Properties: (select to edit options)

Calculate	Property
✓	Vibrational frequencies
	Surfaces (MO, density, potential)
	Atomic electrostatic potential charges (ESP)

Vibrational frequencies

- Use available Hessian: ☐
- IR Intensities: ☐
- Atomic masses: Most abundant isotopes
- Scaling: None

Thermochemistry

- Pressure: 1.0 atm
- Start temperature: 298.15 K
- Increment: 10.0 K
- Number of steps: 1
- Output units: [✓ kcal/mol] KJ/mol

Job name: jaguar_1

Jaguar: B3LYP / 6-31G**, Single Point Energy, Host=localhost, Incorporate=Append new entries

Figure 3.8. The Properties tab showing controls for vibrational frequencies.

136. Keyword **ifreq** = 1 in the **gen** section.

3.10.1 Frequencies

Vibrational frequencies are calculated by Jaguar using analytic second derivatives by default for restricted and unrestricted HF and DFT wave functions in the gas phase and in solution, for basis sets that do not include f functions. (See Section 3.2 on page 33 for more information on basis sets.) All other frequency calculations are done with numerical derivatives of analytically computed forces. Frequencies can be calculated numerically for HF, LMP2, and DFT wave functions in gas phase or in solution. Frequencies are always calculated numerically for RODFT wave functions when an effective core potential is used.

Analytic frequency calculations are much faster than numerical frequency calculations. When frequencies are calculated analytically, molecular symmetry is turned off for the frequency calculation itself, but is used for the rest of the job. To calculate frequencies numerically, set **nmder**=2 in the **gen** section of the input file, as described in Section 9.5 on page 204.

To compute frequencies and any frequency-related properties from the Hessian available at the end of a job (either an initial Hessian, if it was never updated, or the updated Hessian), select Use available Hessian[137].

3.10.2 Atomic Masses

For frequency calculations the atomic mass used for each element is that of its most abundant isotope by default.[138] However, you can choose to use an average of the isotopic masses, weighted by the abundance of the isotopes, by selecting Average isotopic masses[139] from the Atomic masses option menu.

3.10.3 Scaling of Frequencies

Because the errors in quantum mechanical calculations of frequencies are often fairly predictable, it is sometimes desirable to scale frequencies by one or more factors. Scaling methods can also improve calculations of thermochemical properties, which use the scaled frequencies. In Jaguar, two methods are available for frequency scaling: the Pulay *et al.* Modified Scaled Quantum Mechanical Force Fields (SQM) method [107] for B3LYP calculations using the 6-31G* basis set, and standard frequency scaling, in which all frequencies are simply multiplied by a single parameter.

The SQM method alters the frequencies by scaling the Hessian elements themselves (in internal coordinates), using 11 different scale factors, which depend on the type of stretch, bend, or torsion. This method was parametrized using B3LYP calculations for 30 molecules

137. Keyword **ifreq** = –1 in the **gen** section.
138. Keyword **massav** = 0 in the **gen** section.
139. Keyword **massav** = 1 in the **gen** section.

containing C, H, N, O, and Cl, using the 6-31G* basis set. Jaguar permits the SQM scaling method to be used only for B3LYP/6-31G* frequency jobs. You can turn on SQM scaling by selecting Pulay SQM[140] from the Scaling option menu. The method is off[141] by default.

Alternatively, for any type of frequency job, you can multiply all frequencies by the same scale factor. Scale factors have been optimized for many combinations of basis set and method. You can choose one of these factors by selecting Automatic[142] from the Scaling option menu, or you can set a factor by selecting Custom from the option menu and entering a factor in the text box[143] that is displayed. Table 3.4 lists the recommended scale factors for various methods and basis sets. The factors in the table are from Ref. 108 and Ref. 109 and are optimized for the best agreement with experiment for the frequencies themselves. Ref. 108 also includes scale factors suitable for low-frequency vibrations, for zero-point vibrational energies, and for prediction of enthalpy and entropy. Other scale factors may be available in the literature.

Table 3.4. Recommended frequency scale factors for various combinations of SCF method and basis set

SCF Method	Basis Set	Factor	SCF Method	Basis Set	Factor
HF	3-21G	0.9085[a]	M06-L	6-31G*	0.9580
HF	6-31G*	0.8953[a]	PBE0	6-31G*	0.9500
HF	6-31+G*	0.8970[a]	PBE0	6-31G**	0.9515
HF	6-31G**	0.8992[a]	PBE	6-31G*	0.9869
HF	6-311G**	0.9051[a]	OLYP	6-31G*	0.9766
MP2	6-31G*	0.9434[a]	SVWN5	6-31G*	0.9860
MP2	6-31G**	0.9370[a]	PW6B95-D3	6-31G*	0.9481
MP2	6-311G**	0.9496[a]	B97-D3	6-31G*	0.9852
BLYP	6-31G*	0.9945[a]	M06-2X	6-31+G**	0.940[b]
BP86	6-31G*	0.9914[a]	M06	6-31+G**	0.950[b]
B3P86	6-31G*	0.9558[a]	M06-L	6-31+G**	0.953[b]
B3PW91	6-31G*	0.9573[a]	B1B95	6-31+G**	0.946[b]
M06-2X	6-31G*	0.9433	BB1K	6-31+G**	0.929[b]
M06-2X	cc-pvdz	0.9510	M05-2X	6-31+G**	0.936[b]
M06-2X	cc-pvtz(-f)	0.9503	M06-HF	6-31+G**	0.931[b]

140. Keyword **isqm** = 1 in the **gen** section.
141. Keyword **isqm** = 0 in the **gen** section.
142. Keyword **auto_scale** = 1 in the **gen** section.
143. Keyword **scalfr** in the **gen** section.

Table 3.4. Recommended frequency scale factors for various combinations of SCF method and basis set (Continued)

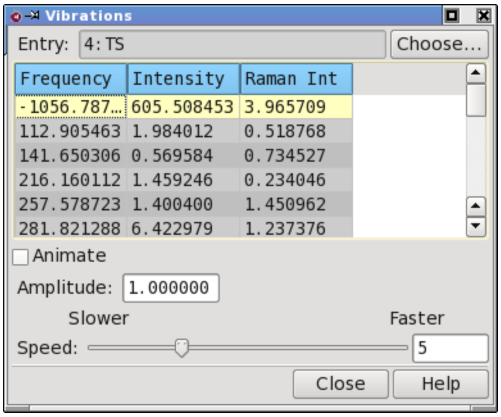
SCF Method	Basis Set	Factor	SCF Method	Basis Set	Factor
B3LYP	6-31G*	0.9605	MPW1K	6-31+G**	0.924[b]
B3LYP	6-31G**	0.9618	MPW1K	MIDIX	0.928[b]
B3LYP-D3	6-31G*	0.9612	MPWB1K	6-31+G**	0.926[b]
M06	6-31G*	0.9585	PW6B95	6-31+G**	0.945[b]

a. Ref. 108
b. Ref. 109

3.10.4 Animation of Frequencies

Maestro can display vibrational animations based on Jaguar frequency data. This data is written in a file with a `.vib` extension when you perform a frequency calculation. The vibrational data is incorporated when the job finishes, and a V button is added in the Title column for each entry that has vibrational data—much like the S button for surface data. Clicking the button opens the Vibration panel, in which you can select the frequency to be animated and control the amplitude and speed of the animation. You can switch modes and change entries during the animation.

If the frequency job was not incorporated for any reason, you can read the restart file into the Jaguar panel, then import the vibrational data by choosing Import Vibrational Data from the Selection menu in the Project Table panel.



Entry: 4: TS

Frequency	Intensity	Raman Int
-1056.787...	605.508453	3.965709
112.905463	1.984012	0.518768
141.650306	0.569584	0.734527
216.160112	1.459246	0.234046
257.578723	1.400400	1.450962
281.821288	6.422979	1.237376

Amplitude: 1.000000

Speed: 5

Figure 3.9. The Vibrations panel.

To view vibrational animations from calculations run with older versions of Jaguar, you can quickly generate the `.vib` file using the Jaguar restart file from a frequency calculation using the following procedure:

1. Read the restart file into the Jaguar panel.

   The structure is displayed in the Workspace and an entry is created in the Project Table.

2. Choose Applications → Jaguar → Initial Guess Only.
3. In the Properties tab, select Vibrational frequencies in the table, then select Use available Hessian in the Vibrational frequencies section.
4. Run the job.

An alternative to steps 2 and 3 is to edit the input file, and in the **gen** section, change **ifreq**=1 to **ifreq**=–1, and add **igonly**=1. The former setting means "use available Hessian for calculating frequencies" and the latter setting means "skip the SCF."

The job should take only a few seconds, even for a large molecule. When the job finishes, a new entry is added to the Project Table that includes a V button in the Vib column, with which you can open the Vibrations panel.

You can also generate a Molden input file after a frequency calculation, enabling you to visualize the frequencies with this program. See Section 9.5.24 on page 259 for more information on writing a Molden input file.

3.10.5 Infrared and Raman Intensities

Infrared intensities can be computed analytically for all HF and DFT wave functions, except for RODFT calculations with an effective core potential. They can also be computed numerically for HF, or DFT wave functions. Infrared intensities for each frequency in km/mol are calculated by default. You do not need to select IR Intensities.[144] For analytic calculations of IR intensities, molecular symmetry is not used. For calculations for which frequencies are computed numerically, the numerical derivative of the dipole moment is used for IR intensity calculations. If you want IR intensities for molecules for which the frequencies would normally be calculated analytically and the IR intensities are not available analytically, the frequencies must be computed numerically, by setting **nmder**=2 in the **gen** section.

When frequencies and IR intensities are calculated, a spectrum file is written that contains the spectral data, *jobname*`_vib.spm`. This file can be read into the Spectrum Plot panel in Maestro to generate a simulated ir spectrum. The Spectrum Plot panel is described in Section 3.12 on page 80.

144. Keyword **irder** = 1 in the **gen** section.

Raman intensities can be computed for any method that produces vibrational frequencies. To calculate Raman intensities, select Raman spectroscopy in the Properties table of the Properties tab. The intensities are reported in the output file (see Section 6.3.8 on page 157).

Both IR and Raman intensities are written to the `.vib` file, which is read by Maestro and used to display frequencies and intensities in the Vibrations panel (see Section 3.10.4 on page 72).

3.10.6 Thermochemical Properties

Thermochemical calculations of the constant volume heat capacity (C_v), internal energy (U), entropy (S), enthalpy (H), and Gibbs free energy (G) at standard temperature and pressure are performed by default whenever vibrational frequencies are calculated. Rotational symmetry numbers, which identify the number of orientations of a molecule which can be obtained from each other by rotation, and zero point energies are also computed. You can calculate these properties only if you are also computing vibrational frequencies.

With the settings in the Thermochemistry section, you can set the temperatures and the pressure used for calculations of these quantities. The pressure[145] (in atm) used for thermochemical calculations is 1.0 by default, and the initial temperature[146] (in K) is 298.15 by default. To compute thermochemical properties at more than one temperature, specify the differences between temperatures using the Increment[147] text box and the number of temperatures at which to compute thermochemical properties in the Number of steps[148] text box. The defaults are 10.00 K and 1 step (meaning only one temperature is used).

By default, thermochemical output is in units of kcal/mol (for H and G) and cal/mol K (for C_v and S). For output in units of kJ/mol and J/mol K instead, select kJ/mol under Output.[149]

In addition to the properties as a function of temperature, the total internal energy, total enthalpy and total free energy at the given (or initial) temperature are written to the output file and stored as Maestro properties. The units for these total values are hartrees.

3.10.7 Vibrational Circular Dichroism

Vibrational circular dichroism (VCD) spectra are used to determine the chirality of molecules. The VCD spectrum of a molecule is the mirror image of the VCD spectrum of its enantiomer, and the two spectra can therefore be easily distinguished. The assignment of chirality can be done by comparing the calculated spectrum [247, 248], for which the chirality is known, with the experimental spectrum.

145. Keyword **press** in the **gen** section.
146. Keyword **tmpini** in the **gen** section.
147. Keyword **tmpstp** in the **gen** section.
148. Keyword **ntemp** in the **gen** section.
149. Keyword **ip28** = 2 in the **gen** section.

To calculate the spectral data, select Vibrational circular dichroism in the Calculate column of the Properties table[150]. As well as information in the output file, the frequencies and intensities of the transitions are written to a spectrum file, called *jobname*`_vcd.spm`. This file can be read by the Spectrum Plot panel in Maestro, which generates a simulated spectrum. As VCD calculations involve both the frequencies and a magnetic field, the thermochemical properties and the NMR shieldings are computed as a byproduct. It also follows that any changes to the settings for the frequencies affect the VCD spectrum.

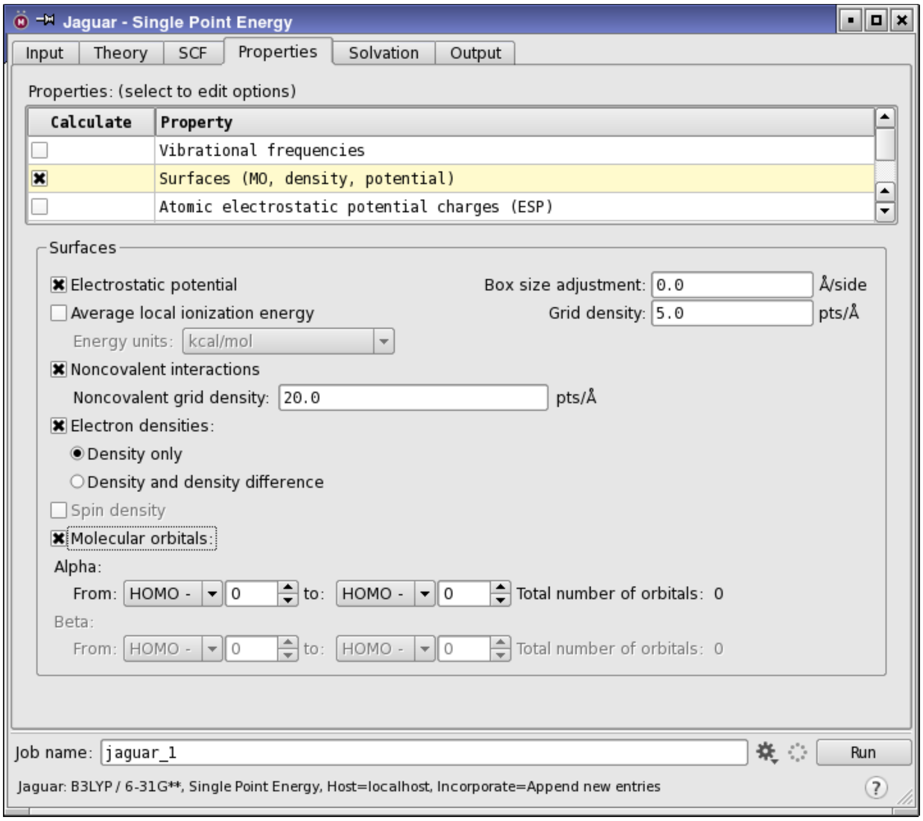
Because conformers of a molecule have significantly different VCD spectra, it is important to compute a conformationally averaged spectrum when the molecule of interest has low-lying conformers. You can use MacroModel to perform the conformational search, then refine the geometries of each conformer with Jaguar, and eliminate high-lying or redundant conformers with MacroModel. This procedure is automated in the VCD Workflow—see Section 5.3 on page 119. Then you can calculate their VCD spectra, and use the Spectrum Plot panel to generate an averaged spectrum using Boltzmann weights based on Jaguar's conformational energies. See Section 6.1 of the *Jaguar Quick Start Guide* for an example of this procedure. The Spectrum Plot panel is described in Section 3.12 on page 80.

3.11 Surfaces

Surfaces provide 3D spatial information about a molecule that cannot be obtained by merely calculating property values. Jaguar can generate electrostatic potential, average local ionization energy, noncovalent interaction, electron density, electron density difference, electron spin density, Fukui function, and orbital surface data, which can be visualized in Maestro. When you generate the electrostatic potential, the values at the nuclei are also added to the Maestro file as an atom property. Values at ECP centers should be ignored as erroneous, due to the missing core contributions.

Surfaces such as the electrostatic potential (ESP) and average local ionization energy (ALIE) are helpful for determining charge buildup and reactivity at various locations in a molecule. The ESP describes the potential energy experience by a positive test charge at a point in space; the ALIE is a sum of orbital energies weighted by the orbital densities [110,111], which provides an energetic measure of how easy or difficult it is to remove electrons from regions of the molecule. The ALIE is better than the ESP at describing how a molecule would respond to electrophilic attack. See Section 3.3 of the *Jaguar Quick Start Guide* for a tutorial example on use of the ESP surface. In addition to the surfaces, an analysis of the ALIE and ESP on an isodensity surface is performed, recorded as a set of atomic and molecular properties in the output file and the Maestro file (see Section 9.5.15 on page 236).

150. Keyword **ivcd** = 1 in the **gen** section.



Properties: (select to edit options)

Calculate	Property
☐	Vibrational frequencies
☒	Surfaces (MO, density, potential)
☐	Atomic electrostatic potential charges (ESP)

Surfaces

☒ Electrostatic potential — Box size adjustment: 0.0 Å/side

☐ Average local ionization energy — Grid density: 5.0 pts/Å

Energy units: kcal/mol

☒ Noncovalent interactions

Noncovalent grid density: 20.0 pts/Å

☒ Electron densities:

● Density only

○ Density and density difference

☐ Spin density

☒ Molecular orbitals:

Alpha:

From: HOMO - 0 to: HOMO - 0 Total number of orbitals: 0

Beta:

From: HOMO - 0 to: HOMO - 0 Total number of orbitals: 0

Job name: jaguar_1 Run

Jaguar: B3LYP / 6-31G**, Single Point Energy, Host=localhost, Incorporate=Append new entries

Figure 3.10. The Properties tab showing controls for surfaces.

Noncovalent interactions (NCI) can be visualized with the help of the reduced density gradient and interaction strength, following the methods of Johnson et al [249]. NCI plots take the form of surface maps, somewhat like color-coded ESP maps. The isosurfaces show where there are interactions between noncovalently-bound atoms, and the map coloring indicates the strength and favorability or unfavorability of the interactions. Note that the “interaction strength” is not in terms of energy, but in terms of electron density. However, there is some correlation between the magnitude of the interaction strength and the intermolecular binding energy. Since the electron density is on an absolute scale, it is possible to compare the NCI maps of different molecular systems. The method is only valid for molecules at equilibrium (or near-equilibrium) geometry, so it is advisable to perform a geometry optimization first. For dispersively-bound systems, it is not necessary to use explicitly correlated theoretical models like LMP2. This is because the NCI surface plots are based on the electron density, not the total energy. Density difference plots for correlated and uncorrelated levels of theory typically show small differences, while the differences in total energies, or intermolecular binding energies, can be large.

In addition to the reduced density gradient and interaction strength, the Laplacian of the density is calculated. The Laplacian surface can be used to identify places of electron excess and deficiency, and in turn, sites of reactivity. Holes in the isosurface identify potential sites for nucleophilic attack, as well as the preferred angle of attack.

See Section 3.4 of the *Jaguar Quick Start Guide* for a tutorial example on noncovalent interactions and the density Laplacian.

The electron density difference surface is the difference between the initial guess density and the density from the final converged SCF wave function. The interpretation of this density difference therefore depends on the initial guess. For example, if the initial guess density is the superposition of atomic densities, then the difference density is the density change on molecule formation. If it is derived from another geometry, it represents the relaxation of the density due to the geometry change.

3.11.1 Generating Surfaces

By selecting Surfaces (MO, density, potential) in the Properties tab, you can choose to generate electrostatic potential,[151] average local ionization energy,[152] electron density,[153] electron density difference,[154] electron spin density,[155] and orbital data[156] that can be visualized in Maestro. You can generate data for any combination of surfaces in the same run, by selecting the option for that surface type and making any other relevant settings. The data for each surface is written to a separate `.vis` file. You can generate surfaces for each point in a geometry scan.

As well as displaying isosurfaces, Maestro can map data for the surface property onto another surface. Electrostatic potentials and average local ionization energies are often mapped to the electron density surface, so you should also select the electron density if you want to perform this mapping.

The data for the property ("volume" data) is tabulated on a three-dimensional rectangular grid. The box containing the grid encompasses the van der Waals radii of all atoms in the molecule. You can adjust the box size within the range −1 to +25 Å, and you can change the density of grid points within the range 1–25 points/Å, using the Box size adjustment[157] and Grid density[158] text boxes in the Surfaces section.

151. Keyword **iplotesp** = 1 in the **gen** section.
152. Keyword **iplotalie** = 1 in the **gen** section.
153. Keyword **iplotden** = 1 in the **gen** section.
154. Keyword **iplotden** = 2 in the **gen** section.
155. Keyword **iplotspn** = 1 in the **gen** section.
156. Keywords **iorb1a**, **iorb2a**, **iorb1b**, **iorb2b** in the **gen** section.
157. Keywords **xadj**, **yadj**, **zadj** in the **gen** section.
158. Keyword **plotres** = 1 in the **gen** section.

Orbital surfaces can be generated only for SCF wave functions: MP2 calculations do not generate natural orbitals that could be used for surfaces. If you localize the orbitals, the surfaces are generated for the localized orbitals.

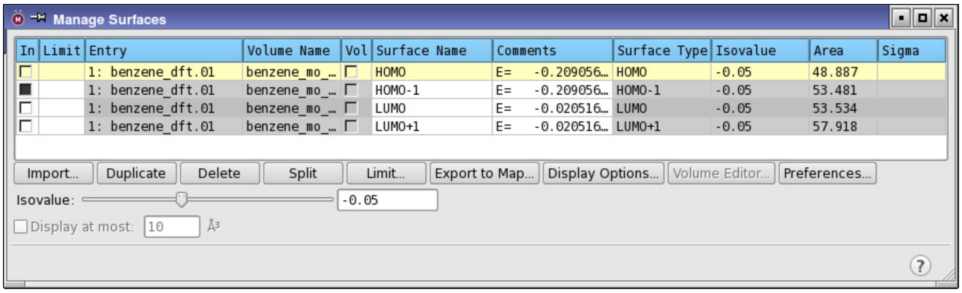
The orbital selection controls depend on whether you have an initial guess or not. If you do not have an initial guess, you can select the orbitals to plot in a range relative to the HOMO and LUMO. If you do have an initial guess, a list is displayed with the orbital index, energy, and occupation. You can select a range of orbitals from this list with the SHIFT key. If the wave function is spin-unrestricted, you can choose alpha and beta orbitals independently.

Fukui functions are generated in a separate panel—see Section 5.2 on page 117.

3.11.2 Displaying Surfaces in Maestro

When the job to generate the data finishes, the surfaces are imported into Maestro and the first surface is displayed. For geometry scans, each point in the scan is imported as a separate entry, and the surfaces are associated with the correct point in the scan.

If your molecule is not already an entry in the Project Table, the surfaces are not automatically incorporated. To display the surfaces, import your molecule into project and select it in the Project Table, then choose Table → Import → Surface. The Import Surface / Volume Data dialog box is displayed, and you can navigate to the `.vis` files for the surfaces, select them and import them. Then choose Workspace → Surface → Manage Surfaces to open the Manage Surfaces panel, or click the S button in the Title column of the Project Table.



In	Limit	Entry	Volume Name	Vol	Surface Name	Comments	Surface Type	Isovalue	Area	Sigma
☐		1: benzene_dft.01	benzene_mo_…	☐	HOMO	E= -0.209056…	HOMO	-0.05	48.887	
■		1: benzene_dft.01	benzene_mo_…	☐	HOMO-1	E= -0.209056…	HOMO-1	-0.05	53.481	
☐		1: benzene_dft.01	benzene_mo_…	☐	LUMO	E= -0.020516…	LUMO	-0.05	53.534	
☐		1: benzene_dft.01	benzene_mo_…	☐	LUMO+1	E= -0.020516…	LUMO+1	-0.05	57.918	

Isovalue: -0.05

☐ Display at most: 10 Å³

Figure 3.11. The Manage Surfaces panel.

You can view multiple surfaces (e.g., plots for several orbitals) for the same molecule, but they are superimposed by default. To view these surfaces separately, arranged in a grid in the Workspace, choose View → Tile → Tile by Surface, or choose Tile by Surface from the Tile entries button on the View toolbar.

See Section 2.6 of the *Jaguar Quick Start Guide* for a tutorial example of generating and displaying orbitals.

In addition to displaying isosurfaces for the data generated by Jaguar, you can display the value of the property for which you generated the data as a color map on another surface that is generated by Maestro. This is often done for the electrostatic potential surface and can also be done for the average local ionization energy surface.

For example, you can display the value of the electrostatic potential on the van der Waals surface generated by Maestro. To do this, select the surface that you want to map the data to in the Manage Surfaces panel, and click Display Options. In the Surface Display Options dialog box, select Map values from volume data and choose the volume from the list below, such as the electrostatic potential. See Section 3.3 of the *Jaguar Quick Start Guide* for a tutorial example. Another example is for NCI plots, where you select the reduced density gradient surface in the Manage Surfaces panel, and map the interaction strength to the surface. See Section 3.4 of the *Jaguar Quick Start Guide* for a tutorial example.

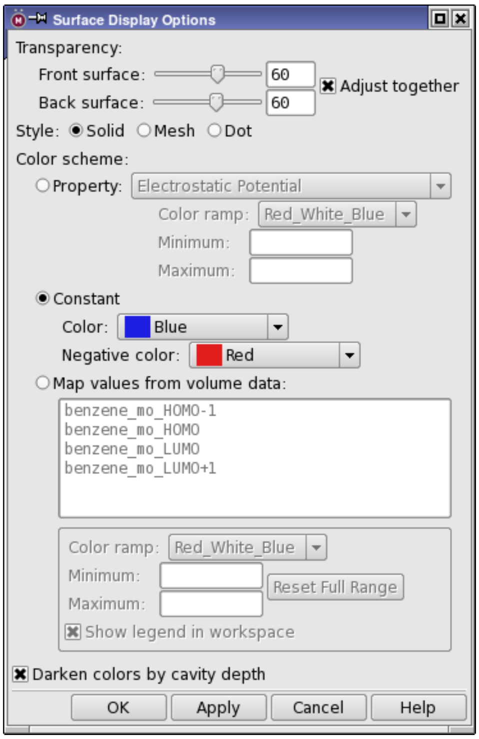


Figure 3.12. The Surface Display Options dialog box.

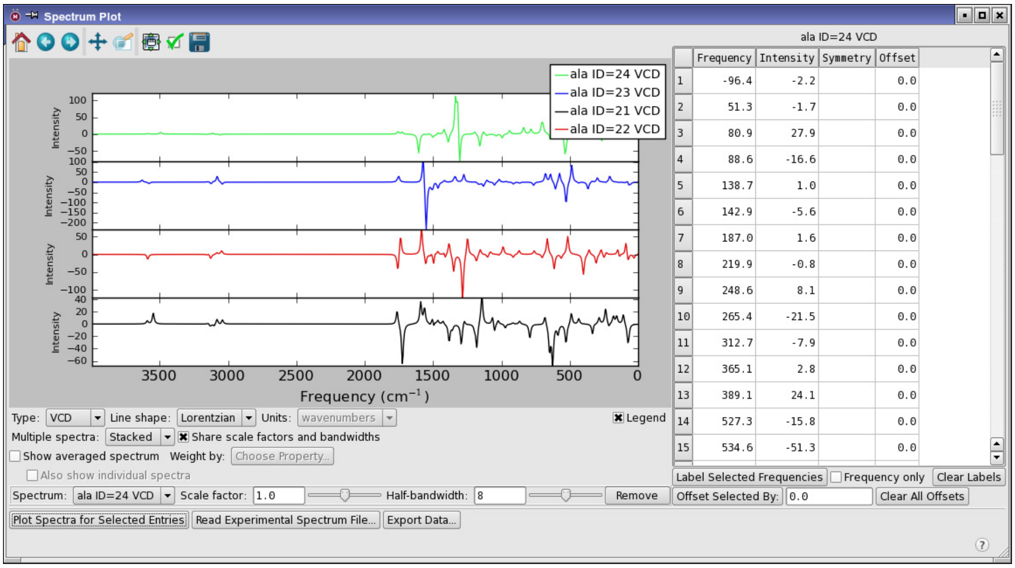
3.12 Plotting Spectra

The Spectrum Plot panel in Maestro can be used to plot UV/visible, infrared, and VCD spectra. The spectral data are stored in spectrum (`.spm`) files, which are written when you run the relevant calculation: CIS or TDDFT for UV/visible spectra (`_uvv.spm`), frequencies for IR spectra (`_vib.spm`), and vibrational circular dichroism for VCD spectra (`_vcd.spm`).

The panel allows you to plot single spectra or multiple spectra. Multiple spectra can be displayed in an overlaid format, in a stacked format, or as a weighted average. Spectra are plotted for the entries that are selected in the Project Table, so you should first select the relevant entries. To open the panel, choose Applications → Jaguar → Plot Spectra.

When the panel opens, the first task is to choose the spectrum type from the Type option menu. You must do this before importing the spectrum files, so that the file selector lists files of the correct type. You can make other settings before importing the spectra, such as the line shape and units, and whether to display the legend. If you have multiple spectra, you can choose how they are displayed from the Multiple spectra option menu.

To create the spectra, click Plot Spectra for Selected Entries. A series of file selectors opens, so that you can select the correct spectrum file for each selected entry in the Project Table. The file selector title includes the entry title and entry ID. When a spectrum file has been opened for each entry, the spectra are plotted according to the settings made in the panel.



ala ID=24 VCD

	Frequency	Intensity	Symmetry	Offset
1	-96.4	-2.2		0.0
2	51.3	-1.7		0.0
3	80.9	27.9		0.0
4	88.6	-16.6		0.0
5	138.7	1.0		0.0
6	142.9	-5.6		0.0
7	187.0	1.6		0.0
8	219.9	-0.8		0.0
9	248.6	8.1		0.0
10	265.4	-21.5		0.0
11	312.7	-7.9		0.0
12	365.1	2.8		0.0
13	389.1	24.1		0.0
14	527.3	-15.8		0.0
15	534.6	-51.3		0.0

Type: VCD Line shape: Lorentzian Units: wavenumbers ☒ Legend
Multiple spectra: Stacked ☒ Share scale factors and bandwidths
☐ Show averaged spectrum Weight by: Choose Property...
☐ Also show individual spectra
Spectrum: ala ID=24 VCD Scale factor: 1.0 Half-bandwidth: 8
Offset Selected By: 0.0

Figure 3.13. The Spectrum Plot panel.

Colors and line styles are assigned automatically, but you can change them if you want. To do this, choose the particular spectrum from the Spectrum option menu, then click the Edit axes and curve settings toolbar button, and use the Figure Options dialog box to make settings for the title, range, labeling, and scale of the axes; the color, style, and width of lines; and the color, style, and size of markers.

You can also change the scaling factor for the transition energies or frequencies and the width of the lines. If Share scale factors and bandwidths is selected, the spectra all have the same scaling factor and widths. Deselect this option to if you want to make different settings values for one or more spectra.

The list of spectral lines is given in the table on the right. If you want to label certain peaks in the spectrum with the frequency and intensity, you can select the lines in the table and click Label Selected Frequencies. To remove the labels, click Clear Labels. To label them with only the frequency (and not the intensity), select Frequency only.

You can adjust the positions of individual lines in the spectrum. To do this, edit the value in the Offset column, which is zero by default. The spectrum is recalculated when you have edited a value. If you want to apply the same offset to several lines, select them in the table, and enter a value in the text box next to the Offset Selected By button, and click the button. The offsets are added to the table, and the spectrum is recalculated.

You can remove a spectrum from the display by selecting it from the Spectrum option menu and clicking Remove. If you remove a spectrum, it cannot be restored. You would then have to click Plot Spectra for Selected Entries again to redraw the spectra.

If you change the spectrum type on the Type option menu between UV/Vis and either VCD or IR, the current spectra are cleared, and you need to click Plot Spectra for Selected Entries to import the spectrum files for the new spectrum type. VCD and IR spectra can be displayed at the same time.

To create an averaged spectrum, select Show averaged spectrum, then click Choose Property. A property selector opens, in which you can select an entry property. Suitable properties might be the concentration of the structures in solution, or Boltzmann weights calculated from the relative energies (which you can calculate with Scripts → Project Table → Calculate Boltzmann Populations). The property name is displayed to the right of the button. This is particularly useful for comparing with experimental VCD spectra, which are usually a conformational average. An example of calculating and displaying a VCD spectrum is given in Section 6.1 of the *Jaguar Quick Start Guide*. If you want to show the individual spectra as well as the averaged spectrum, select Also show individual spectra. The spectra are displayed either stacked or overlaid, depending on the choice made from the Multiple spectra option menu.

In addition to displaying predicted spectra, you can also read in experimental spectra by clicking Read Experimental Spectrum File. An experimental spectrum must be in a plain text file (`.txt`), consisting of pairs of frequency and intensity values, one per line, with the values separated by spaces. This allows you to compare experimental spectra with predicted spectra. The experimental spectra are always displayed, and are not included in any averaging. Experimental spectra have no data on the positions or intensities of lines, so the Frequency table is empty, and controls for the line shape or scaling factor are not available.

If you want to compare spectra in another program, you can export the spectral data (with the line broadening) to a text file. The file contains two columns for each spectrum, with the x (frequency) and y (intensity) data, labeled with the spectrum name as *name*`-X` and *name*`-Y`. To export the data, click Export Data, navigate to the location in the file selector, and name the file.

3.13 Options for Extra Output

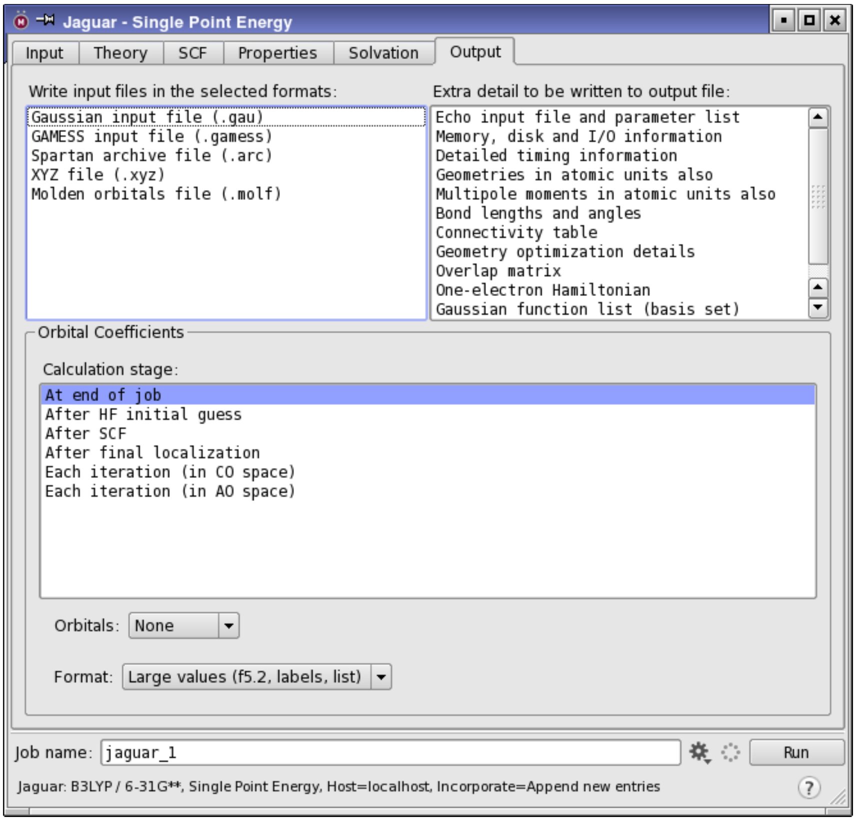
The options available in the Output tab of the Jaguar panel under Extra detail to be written to output file are described in this section. These options are presented as a list, from which you can select multiple items with the SHIFT and CTRL keys.

The output generated from these options appears in the output file for the job. If you make a non-default setting, the output from the program `pre` prints the non-default options chosen. This output appears above the molecular geometry output from `pre`, and gives the non-default values of the keywords referred to in footnotes throughout this section.

Echo input file and parameter list

If you turn this output option on, the output from the program `pre` includes an echo of the input file, a description of the path, which indicates the Jaguar programs run, and a list of keyword settings, including those made by default, and program parameters.[159] This option is likely to be useful primarily for people who have a detailed knowledge of the code itself.

159. **echo** section constructed, and keywords **mtest** = 2 and **ip24** = 2 in the **gen** section.



Write input files in the selected formats:

```
Gaussian input file (.gau)
GAMESS input file (.gamess)
Spartan archive file (.arc)
XYZ file (.xyz)
Molden orbitals file (.molf)
```

Extra detail to be written to output file:

```
Echo input file and parameter list
Memory, disk and I/O information
Detailed timing information
Geometries in atomic units also
Multipole moments in atomic units also
Bond lengths and angles
Connectivity table
Geometry optimization details
Overlap matrix
One-electron Hamiltonian
Gaussian function list (basis set)
```

Orbital Coefficients

Calculation stage:

```
At end of job
After HF initial guess
After SCF
After final localization
Each iteration (in CO space)
Each iteration (in AO space)
```

Orbitals: None

Format: Large values (f5.2, labels, list)

Job name: jaguar_1

Jaguar: B3LYP / 6-31G**, Single Point Energy, Host=localhost, Incorporate=Append new entries

Figure 3.14. The Output tab.

Memory, disk, and i/o information

The memory information provided by this option is given for most of the routines used during the run, under the heading “dynamic memory statistics.”[160] Current and maximum values for the number of arrays, their size in 8-byte words, and their size in bytes, as well as the type of variables used (e.g., real*8), are listed. The total and index i/o for the J and K matrices, in Mwords, are also provided after the energy output from the SCF iterations.

Detailed timing information

If you select this option, the CPU time in seconds spent in various Jaguar programs is listed in the output.[161]

160. Keyword **ip5** = 2 in the **gen** section.
161. Keyword **ip6** = 2 in the **gen** section.

Geometries in atomic units also

This option allows you to print the geometry output in atomic units as well as in the default units, angstroms.[162]

Multipole moments in atomic units also

If you choose to calculate multipole moments by making the appropriate setting in the Properties tab, this option allows you to list them in the output file in atomic units as well as in the default units, debye.[163]

Bond lengths and angles

When this option is turned on, the internuclear distances in angstroms are listed for all nearest neighbor atoms in the output from the program `pre`, and the bond angles in degrees are given as well.[164] The atoms are indicated with the atom labels assigned in the geometry input. When you optimize a geometry, this option is turned on automatically. For geometry optimizations, bond lengths and angles are also listed with the output from the program `geopt`.

Connectivity table

The connectivity table provided by this option describes roughly how closely the atoms interact.[165] Connectivity partially determines whether molecular fragments exist, the content of the initial Hessian, and many other properties of a calculation. The assignment of dealiasing functions for the pseudospectral method also depends upon the connectivities shown in this table, which reflect the neighbor ranges defined in the `.daf` file. (See Section 10.3 on page 300 for more information.) All of the diagonal entries are 0, indicating that the row atom and the column atom for the matrix element are the same atom. An entry of 1 indicates that the row atom and the column atom are considered to be bonded, because they are separated by a distance less than the sum of their covalent radii times the variable **covfac**, which is 1.2 by default and is also described in Section 9.5.2 on page 205. If a connectivity table entry is 2, the corresponding row and column atoms are each bonded to some same third atom, by the definition of bonding described above. An entry of 3, 4, or more means that the atoms are within the third, fourth, or higher neighbor range of each other.

Geometry optimization details

If this option is selected,[166] additional information about the progress of a geometry optimization is printed. This output often helps reveal the cause of any problems with optimizations.

162. Keyword **ip26** = 2 in the **gen** section.
163. Keyword **ip25** = 2 in the **gen** section.
164. Keyword **ip11** = 2 in the **gen** section.
165. Keyword **ip12** = 2 in the **gen** section.
166. Keyword **ip192** = 2 in the **gen** section.

Overlap matrix

The overlap matrix **S** for the basis functions is printed in five-column blocks if this option is selected.[167] Since the matrix is symmetric, the upper triangle is not printed.

One-electron Hamiltonian

The one-electron matrices representing kinetic energy and the sum of kinetic energy, nuclear attraction, and point charge-electron interactions is printed in atomic orbital space in five-column blocks if this option is selected.[168] Since the matrices are symmetric, the upper triangles are not printed.

Gaussian function list (basis set)

By selecting this option, you can print out information about the Gaussian functions that make up the basis set.[169]

Gaussian function list (derivatives)

By selecting this option, you can print out information about the derivatives of the basis set functions in terms of primitive Gaussians.[170] The format and information is the same as that discussed for the Gaussian function list (basis set) option immediately above.

3.13.1 File Output Options

The options available in the Output tab of the Jaguar panel under Write input files in the selected formats are described in this section. These options are presented as a list, from which you can select multiple items with the SHIFT and CTRL keys. These output options generate additional files. For each of the options described below, the relevant file appears in the same directory as the output file. Each file name is in the form *jobname*.*suffix*, where the different suffixes for each kind of file are described below.

If you make a setting in the Output tab, the output from the program `pre` lists the non-default options chosen. This output appears above the molecular geometry output from `pre`, and gives the non-default values of the keywords referred to in footnotes in this section.

Gaussian input file (.gau)

When this option is selected, a file in the format of a GAUSSIAN input file is created, with the suffix `.gau`.[171] The file information includes the molecular geometry, the basis set name, and

167. Keyword **ip18** = 2 in the **gen** section.
168. Keyword **ip19** = 2 in the **gen** section.
169. Keyword **ip1** = 2 in the **gen** section.
170. Keyword **ip8** = 2 in the **gen** section.
171. Keyword **ip160** = 2 in the **gen** section.

the type of calculation to be performed, as well as the molecular charge and the spin multiplicity of the molecule and any relevant effective core potential information. If symmetry is turned off, that setting will be entered into the `.gau` file.

For more information on setting up GAUSSIAN input files, see Section 7.9 on page 180.

GAMESS input file (.gamess)

To write out an input file for the program GAMESS, you can select this option.[172] The resultant file's suffix will be `.gamess`. The file will include the molecular geometry, the basis set, and some information on the type of calculation to be performed, as well as the molecular charge and the spin multiplicity of the molecule and any relevant effective core potential information.

SPARTAN archive file (.arc)

You can use this option to generate a SPARTAN 4.0 archive file with the suffix `.arc`.[173]

XYZ file (.xyz)

If you set this option, Jaguar creates a file in XYZ format with the suffix `.xyz`.[174] The file contains all geometries generated during the course of the job, except that for solvated geometry optimizations, the file only contains the solvated structures.

Molden orbitals file (.molf)

You can use this option to produce a file with the final orbitals in a format suitable for the program Molden [176].[175] If you have run a frequency calculation, the normal modes are written to the `.molf` file.

3.13.2 Output Options for Orbitals

Orbital information can be printed to the output file as well. Several possible choices are available in the Orbital Coefficients section of the Output tab. If you choose to print out orbital information, the output from the program `pre` lists the non-default options chosen above the molecular geometry output from `pre`, and indicates the keywords referred to in footnotes throughout this section.

You can select options that determine the point at which orbitals are printed out from the Calculation Stage list. To select multiple items, you can use the SHIFT and CTRL keys in combination with clicks. The available items are:

172. Keyword **ip168** = 2 in the **gen** section.
173. Keyword **ip165** = 3 in the **gen** section.
174. Keyword **ip175** = 2 in the **gen** section.
175. Keyword **ip90** = 2 in **gen** section of input file.

- At end of job—Print the orbitals at the end of the job.[176]
- After HF initial guess—Print orbitals used for the HF initial guess.[177]
- After SCF—Print orbitals in atomic orbital space after the SCF converges.[178]
- After final localization—Print orbitals after the localization procedure, if Boys or Pipek-Mezey localization of the wave function has been requested.[179]
- Each iteration (in CO space)—Print orbitals after each SCF iteration in canonical orbital space.[180] (Canonical orbital eigenvectors with very small eigenvalues are removed from the calculation before the SCF process.) The number of orbitals printed depends on whether five or six d functions are specified for the basis set, as described in Section 3.2 on page 33.
- Each iteration (in AO space)—Print orbitals after each SCF iteration in atomic orbital space.[181]

By default, no orbitals are printed in the output file, so None is selected by default in the Orbitals option menu.[182] If you select Occupied orbitals, all occupied orbitals are printed.[183] If the All orbitals option is selected, all occupied orbitals and ten virtual orbitals are printed.[184] To change the default of ten virtual orbitals, see the keyword **ipvirt** in Section 9.5.26 on page 261. The virtual orbitals are obtained by diagonalizing $H_0 + \sum f(2J - K)$, where f is the fractional occupation of each orbital (1 for a closed shell).

The format for printing the selected orbitals can be chosen from the Format option menu. The choices available are:

- Large elements as f5.2, labels, in list[185]
- All elements as f10.5, labels, in table[186]
- All elements as f19.15, in list[187]
- All elements as f8.5, in list[188]
- All elements as e15.6, in table[189]

176. Keyword **ip102** in the **gen** section.
177. Keyword **ip105** in the **gen** section.
178. Keyword **ip104** in the **gen** section.
179. Keyword **ip107** in the **gen** section.
180. Keyword **ip101** in the **gen** section.
181. Keyword **ip103** in the **gen** section.
182. This setting corresponds to having all of the orbital output keywords set to 1.
183. Relevant orbital output keyword set to 2, 3, 4, 5, or 6 in the **gen** section, depending on the format setting chosen.
184. Relevant orbital output keyword set to 7, 8, 9, 10, or 11 in the **gen** section, depending on the format setting chosen.
185. Relevant orbital output keyword set to 2, 7, or 12 in the **gen** section, depending on which orbitals are printed.
186. Relevant orbital output keyword set to 3, 8, or 13 in the **gen** section, depending on which orbitals are printed.
187. Relevant orbital output keyword set to 4, 9, or 14 in the **gen** section, depending on which orbitals are printed.
188. Relevant orbital output keyword set to 5, 10, or 15 in the **gen** section, depending on which orbitals are printed.
189. Relevant orbital output keyword set to 6, 11, or 16 in the **gen** section, depending on which orbitals are printed.

Examples of each of these style options appear below.

In the first option listed, the phrase “large elements” indicates that only coefficients larger than a particular value (generally .05) are listed. The notations “f5.2” and the like refer to standard Fortran formats. The word “labels” indicates that the atom identifiers (for instance, “h2”) and the basis function types (for instance, S for s, Z for p_z, or XX for d_{xx}) are shown. Note that in canonical orbital space, the labels indicating atom identifiers and basis function types are meaningless.

The output for each style is shown in either table form or list form. When the orbital output is in table form, each function’s coefficient for each orbital is shown, with the functions shown in numbered rows and the orbitals in numbered columns. When it is in list form, each orbital is listed in turn, with the basis function coefficients listed in order. For the third and fourth options, those with f19.15 and f8.5 formatting, all coefficients are listed, in order but without numbering. The three styles presented in list form also include information on the occupation and energy of each orbital.

If you generate output in the f19.15 or f8.5 formats, you can use it for input in the **guess** section of an input file, which is described in detail in Section 9.10 on page 283, or for input to GAUSSIAN (guess=cards).

Here are some examples of output for each of these style options. The output shown is from output files generated from a calculation of water with a 6-31G** basis set, for occupied orbitals after the SCF iterations. Only the first two occupied orbitals are shown in each case, and not all functions are shown; these gaps are indicated by [...].

For the Format option Large values (f5.2, labels, list):

```
    1 Orbital Energy   -20.555133 Occupation  1.000000 Symmetry A1
       S
O    0.99
    2 Orbital Energy    -1.345597 Occupation  1.000000 Symmetry A1
       S    S    Z    S
O   -0.21 0.47 0.09 0.42
       S
H1   0.15
       S
H2   0.15
[...]
```

For the Format option All values (f10.5, labels, table):

```
                         1         2         3
  eigenvalues-        -20.55513  -1.34560   [...]
     1 O       S        0.99466  -0.21055
     2 O       S        0.02122   0.47102
    [...]
     5 O       Z        0.00155   0.08586
     6 O       S        0.00430   0.41777
    [...]
    16 H1      S        0.00000   0.14851
    [...]
```

```
   21 H2      S        0.00000   0.14851
   [...]
   25 H2      Z        0.00025  -0.01342
```

For the Format option All values (f19.15, list):

```
    1 Orbital Energy   -20.555133 Occupation  1.000000 Symmetry A1
  0.994661070265476  0.021223773328496  0.000000000000000  0.000000000000000
  0.001550431863529  0.004301782758377  0.000000000000000  0.000000000000000
 -0.000190485390547 -0.003952404680376 -0.003763985866478 -0.003807504316264
  0.000000000000000  0.000000000000000  0.000000000000000 -0.000004988565650
 -0.000343482092802  0.000000000000000  0.000372571507087  0.000252040203901
 -0.000004988565650 -0.000343482092802  0.000000000000000 -0.000372571507087
  0.000252040203901
    2 Orbital Energy    -1.345597 Occupation  1.000000 Symmetry A1
 -0.210549363265932  0.471018758398392  0.000000000000000  0.000000000000000
  0.085862488931510  0.417774726334513  0.000000000000000  0.000000000000000
  0.031498167188452  0.001405346737926  0.006172871870042  0.008194082815896
  0.000000000000000  0.000000000000000  0.000000000000000  0.148513692384474
  0.013067257872503  0.000000000000000 -0.022047889711935 -0.013419565122871
  0.148513692384474  0.013067257872503  0.000000000000000  0.022047889711935
 -0.013419565122871
  [...]
```

For the Format option All values (f8.5, list):

```
    1 Orbital Energy   -20.555133 Occupation  1.000000 Symmetry A1
 0.99466 0.02122 0.00000 0.00000 0.00155 0.00430 0.00000 0.00000-0.00019
-0.00395-0.00376-0.00381 0.00000 0.00000 0.00000 0.00000-0.00034 0.00000
 0.00037 0.00025 0.00000-0.00034 0.00000-0.00037 0.00025
    2 Orbital Energy    -1.345597 Occupation  1.000000 Symmetry A1
-0.21055 0.47102 0.00000 0.00000 0.08586 0.41777 0.00000 0.00000 0.03150
 0.00141 0.00617 0.00819 0.00000 0.00000 0.00000 0.14851 0.01307 0.00000
-0.02205-0.01342 0.14851 0.01307 0.00000 0.02205-0.01342
 [...]
```

For the Format option All values (e15.6, table):

```
             1              2              3
   1    9.946611E-01  -2.105494E-01      [...]
   2    2.122377E-02   4.710188E-01
   [...]
   5    1.550432E-03   8.586249E-02
   6    4.301783E-03   4.177747E-01
   [...]
  16   -4.988566E-06   1.485137E-01
   [...]
  21   -4.988566E-06   1.485137E-01
   [...]
  25    2.520402E-04  -1.341957E-02
```

Optimizations and Scans

For Hartree-Fock, LMP2, and DFT calculations in gas phase or in solution, Jaguar can use calculated analytic gradients to optimize the molecular geometry to a minimum-energy structure or a transition state. In addition to locating stationary points, Jaguar can calculate points along one or more coordinates, with or without optimizing the other coordinates. These scans include intrinsic reaction coordinate (IRC) and minimum energy path (MEP) scans as well as geometry scans of specified coordinates.

There are five tasks on the Jaguar submenu of the Applications menu that support optimizations and scans:

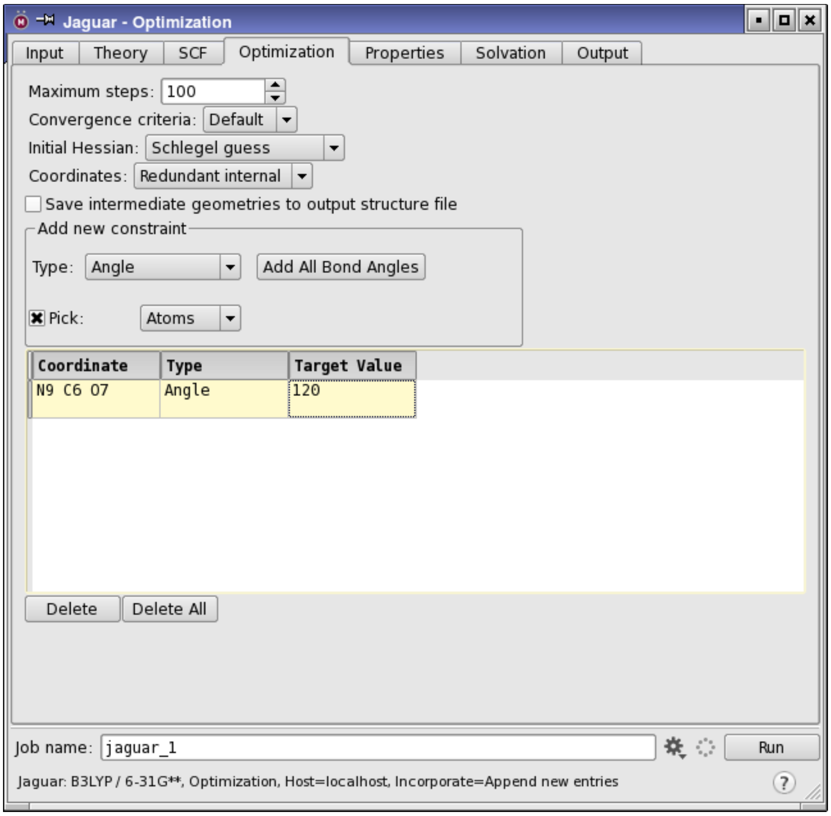
- Optimization[1]
- Relaxed Coordinate Scan
- Rigid Coordinate Scan
- Transition State Search[2]
- Reaction Coordinate[3]

Throughout this chapter, footnotes indicate the Jaguar input file keywords and sections that correspond to particular GUI settings. If you are working from the GUI, you can ignore these footnotes, but you may find them helpful if you decide to use input files to submit jobs without using the GUI or if you want to edit keywords directly by using the Edit Job window described in Section 2.3 on page 11.

4.1 Geometry Optimization: The Basics

To perform a geometry optimization, you need a guess at the geometry and the direction in which to search, a set of coordinates to optimize, and some criteria for when the optimization is complete. The search direction is obtained from the gradient of the energy and the initial Hessian. The general settings for geometry optimizations are in the Optimization tab of the Jaguar panel (Figure 4.1). This tab is present for all of the tasks listed above except Rigid Coordinate Scan, for which no optimization takes place. Four of the tasks have tabs in which settings are made that are specific to the task. These tabs—Transition State, Scan, and IRC—are described in later sections.

1. Keyword **igeopt** = 1 in the **gen** section.
2. Keyword **igeopt** = 2 in the **gen** section.
3. Keyword **irc** = 1 in the **gen** section.



Maximum steps: 100
Convergence criteria: Default
Initial Hessian: Schlegel guess
Coordinates: Redundant internal
☐ Save intermediate geometries to output structure file
Type: Angle
☒ Pick: Atoms

Coordinate	Type	Target Value
N9 C6 O7	Angle	120

Job name: jaguar_1
Jaguar: B3LYP / 6-31G**, Optimization, Host=localhost, Incorporate=Append new entries

Figure 4.1. The Optimization tab.

4.1.1 SCF and Geometry Convergence

During geometry optimization, Jaguar adjusts the convergence criteria for the SCF calculations at each geometry step for efficiency. For the initial iterations of an optimization, the SCF calculations are performed at the Quick accuracy level described in Section 3.7.1 on page 54 (unless the input contains a transition metal, in which case the accuracy level is Accurate). However, for the last few geometry iterations, the accuracy level for the SCF calculations is reset to the Accurate level, which uses tighter cutoffs and denser pseudospectral grids than the Quick level.

For optimizations to minimum-energy structures or transition states, the convergence criterion for SCF calculations is chosen to assure accurate analytic gradients. For these jobs, a wave function is considered converged when the root mean squared (RMS) change in density matrix elements is less than the RMS density matrix element change criterion,[4] whose default value is

4. Keyword **dconv** in the **gen** section.

5.0 x 10^{-6}. (The SCF calculations during an optimization to a minimum-energy structure or transition state do not use the energy convergence criterion used by other SCF calculations.) The RMS density matrix element criterion may be set in the SCF tab.

The geometry is considered to have converged when the energy of successive geometries and the elements of the analytic gradient of the energy and the displacement have met the convergence criteria. Convergence is tested by calculating a score, in which each of the five criteria (on the energy difference, maximum gradient, rms gradient, maximum displacement, rms displacement) adds one point to the score if it is met. The optimization is considered converged if the score is 5 or more (the “standard” criteria). By default, however, an extra point is added if the energy change is less than 0.05 of the threshold and another if it ls less than 0.005 of the threshold, or if the other criteria are met to 0.2 times the threshold (the “flexible” criteria). You can enforce the standard criteria by setting `geoconv_mode=standard` in the **gen** section of the input file. Using the standard criteria is recommended for transition state searches.

The convergence criteria are all affected by the choices in the Convergence criteria section, Default[5], Loose[6], or Tight[7]; the loose criteria are all five times larger than the default criteria, and the tight convergence criteria are a tenth of the default criteria. For optimizations in solution, the default criteria are multiplied by a factor of three, and a higher priority is given to the energy convergence criterion. Thus, if the energy change criterion is met before the gradient and displacement criteria have been met, the geometry is considered converged. See Section 9.5.10 on page 217 for details on the geometry optimization convergence criteria or information on how to edit the input file to set them directly.

In case the optimization process does not converge, you can set an upper limit on the number of steps taken, by entering a value in the Maximum steps[8] text box The default is 100. Many cases will meet the convergence criteria after ten or fewer geometries are computed. However, input containing very floppy molecules, transition metal complexes, poor initial geometries, or poor initial Hessians may require many more cycles, and in particularly bad cases may also require you to stop the calculation and restart it with a change in one or more of the other default settings described below. If you want to save the geometry at each step, you can select Save intermediate geometries in output structure file.[9] You can then select one of these geometries in Maestro to restart your optimization.

At the end of a geometry optimization Jaguar performs a simple analysis of the geometry optimization convergence. The output file contains a verdict that indicates the dynamics of the convergence process—if the convergence was monotonic or erratic—and the quality of the converged structure—if the final geometry corresponds to the lowest energy or not.

5. Keyword **iaccg** = 2 in the **gen** section.
6. Keyword **iaccg** = 3 in the **gen** section.
7. Keyword **iaccg** = 5in the **gen** section.
8. Keyword **maxitg** in the **gen** section.
9. Keyword **ip472** in the **gen** section.

The analysis places the convergence in one of the following five categories:

0: Monotonic convergence.

1: Non-monotonic convergence. However, no erratic convergence is detected. The structure converged to an optimal structure.

2: Erratic convergence but the optimization converged to an optimal structure.

3: The optimization converged to a non-optimal structure. The difference between the converged energy and the minimal energy is less than 0.1 kcal/mole.

4: Converged to a non-optimal structure. The difference between the converged energy and the minimal energy is more than 0.1 kcal/mole.

Categories 0-2 are considered successful convergence. Categories 3-4 might require additional attention. Category 3 convergence is considered successful for optimizations in solution but borderline otherwise. Category 4 should be considered unsuccessful convergence. No category is assigned if the geometry optimization ran out of iterations.

In most cases only category 4 convergence should be scrutinized for potential problems. You might consider starting from a different initial guess or using different geometry optimization settings. If you choose not to rerun your calculation it might be safer to consider the geometry corresponding to the optimization step with the lowest energy as the converged result.

The geometry optimization analysis is controlled by the input file keyword `optverdict`. Set it to 0 if you want to disable the geometry optimization analysis. Set it to 2 if you want to perform the analysis after each optimization step.

If you want to test that the converged structure is a minimum, you can set the **check_min** keyword. There are several options for checking the convergence and finding the minimum if the structure is not actually at a minimum: calculating an analytic Hessian and following its negative eigenvalues, or perturbing various torsions and optimizing the perturbed geometries. Note that this feature is not available with dynamic constraints or for scans or IRC calculations.

4.1.2 The Initial Hessian

To perform an optimization, Jaguar first needs to read or generate an initial Hessian (second derivative matrix or force constant matrix). The Hessian and the gradient are used to define a search direction that should result in a lowering of the energy.

You can provide Jaguar with a Hessian in the **hess** section of an input file (see Section 9.9 on page 282). For instance, if you restart a geometry optimization from a previous job (see Section 7.5 on page 174), Jaguar uses the Hessian provided in your input file. If you want to provide an initial Hessian in the input file, choose Other[10] from the Initial Hessian option menu.

If your input file does not contain a Hessian, you can use the Initial Hessian option menu in the Optimization tab to specify what kind of initial Hessian Jaguar should generate. You can select from among several internal guesses: the Fischer-Almlöf Hessian[11] [96], the Schlegel Hessian[12] [97], or the unit matrix.[13] For most cases, the Schlegel or Fischer-Almlöf options are the best choices. The Schlegel guess is the default.

The final option, Quantum-mechanical,[14] is to have Jaguar compute the initial Hessian. This calculation is the most time-consuming of the initial Hessian options. Theoretically, it is the best option for cases where the other Hessian choices are inadequate, although in practical terms, other steps taken to improve optimizations are likely to be more cost-effective.

4.1.3 Coordinate Systems

The coordinate system you choose for optimization can have a substantial impact on the convergence of the optimization. The ideal set of coordinates is one in which the energy change along each coordinate is maximized, and the coupling between coordinates is minimized. The default coordinate system used by Jaguar is redundant internal coordinates.[15] In most cases, this set of coordinates proves to be the most efficient.

There are cases where geometry optimizations with this set can fail. One example is when a group of atoms becomes collinear, and the internal coordinates become ill-defined. When this happens, Jaguar chooses a new set of redundant internal coordinates. If this process fails, you can restart the optimization with a different choice of coordinates.

Two other coordinate systems are available from the Coordinates option menu in the Optimization tab: Cartesian[16] and Z-matrix.[17] Cartesian coordinates avoid the problems of collinear coordinate sets, but an optimization in Cartesian coordinates is likely to take longer than one in redundant internal coordinates. Using the Z-matrix coordinates is the other alternative. Choosing a set of Z-matrix coordinates that produces an efficient optimization is not a trivial task, and requires an understanding of the coupling between simple internal coordinates. You can mix Cartesian and Z-matrix coordinates in the geometry definition (**zmat** section), but if you do, you cannot use either of these two options for geometry optimizations.

10. Keyword **inhess** = 2 in the **gen** section.
11. Keyword **inhess** = –1 in the **gen** section.
12. Keyword **inhess** = 0 in the **gen** section.
13. Keyword **inhess** = 1 in the **gen** section.
14. Keyword **inhess** = 4 in the **gen** section.
15. Keyword **intopt** = 1 in the **gen** section.
16. Keyword **intopt** = 0 in the **gen** section.
17. Keyword **intopt** = 2 in the **gen** section.

4.2 Constraining Coordinates

When you optimize the geometry of a molecule, you might want to freeze certain coordinates, or constrain them to be equal to each other. Freezing coordinates reduces the number of free parameters in the optimization, and may reduce the number of steps to convergence, or allow you to converge a difficult optimization in stages. Constraining coordinates can be used to enforce symmetry, in either Z-matrix coordinates or Cartesian coordinates.

4.2.1 Freezing Specific Coordinates

To constrain specific coordinates to their original values during an optimization, you can use the controls in the Add New Constraint section of the Optimization tab, or you can edit the input file. (The Edit Job dialog box provides a convenient way of editing the input file.)

Using the Add New Constraint section of the Optimization tab (see Figure 4.1 on page 92), you can pick atoms in the Workspace to define the coordinates to freeze. First choose the coordinate type—Cartesian X, Cartesian Y, Cartesian Z, Cartesian XYZ, Dihedral, Angle, or Distance—from the Type option menu, choose Atoms or Bonds from the Pick option menu, then pick the required number of atoms or bonds in the Workspace to define the coordinate. The coordinate and its type are listed in the Constraints table. For distances you can constrain all atom pairs; for angles you can constrain all bond angles (defined as angles between two contiguous bonds); while for dihedrals you can constrain all torsions (defined as angles between three contiguous bonds).

When you define constraints in this way, internal coordinates are added to a **coord** section, with a # sign to indicate that the coordinate is frozen, and Cartesian coordinate constraints are added to the **zmat** section, by adding a # sign after the constrained coordinates. This means that the **zmat** section must be in the right format for the constraint.

If you edit the input file to add constraints, you can freeze a specific coordinate by adding a "#" sign at the end of its value in your geometry input, in either the **zmat** or the **zvar** section. For example, to fix the HOH bond angle of water at 106.0°, you could use the following Z-matrix:

```
O
H1  O   0.9428
H2  O   0.9428   H1   106.0#
```

If you performed a geometry optimization on this input geometry, the bond angle would remain frozen at 106° while the bond lengths varied.

To freeze a variable during an optimization, add a "#" sign to the end of the variable setting. In this example, the C–H bond is frozen at 1.09 Å:

```
chbond=1.09#  HCHang=109.47
```

You can also freeze a variable by adding a “#” sign to the variable in the Z-matrix or the Cartesian coordinate list. For example, in the following input for optimization of a water molecule, the H atoms are only allowed to move within the xy plane in which they started.

```
O   0.000000     0.000000  -0.113502
H1  0.000000     ycoor      zcoor#
H2  0.000000    -ycoor      zcoor#
ycoor=0.753108   zcoor=0.454006
```

If frozen Cartesian coordinates are included in the input for an optimization, Jaguar uses Cartesian coordinates for the optimization rather than generating redundant internal coordinates, and the optimization does not make use of molecular symmetry.

You can also freeze the average of a set of all possible dihedral angles about a given bond. This average is called a “natural torsional coordinate” [98]. To set up a natural torsional coordinate, you specify the bond in a **coord** section and mark it with `#nt`, as in the example below. Jaguar automatically determines all of the torsional angles about this bond, averages them, and constrains the averaged coordinate to its initial value in an optimization.

```
&coord
C1 C2 #nt
&
```

4.2.2 Applying Harmonic Constraints

Sometimes you don’t want to entirely freeze the value of a coordinate, but allow it to vary within defined limits. You can then optimize the geometry while allowing for small variations of coordinates that you know should remain essentially the same. You can achieve this end by applying harmonic constraints. Harmonic constraints are additional potential energy terms in the form of a harmonic potential with a given force constant *k*, centered at an atom or on a particular internal coordinate value. The constraint can include a region where the potential is zero inside the constrained area. The form of the potential is as follows:

$$\begin{aligned} V &= (k/2)(d-a)^2 && d > a \\ &= (k/2)(d+a)^2 && d < -a \\ &= 0 && -a < d < a \end{aligned}$$

For atomic (Cartesian) harmonic constraints, the value of *d* is the distance from the atom. For internal coordinate harmonic constraints (bond length, bond angle, or dihedral angle), the value of *d* is the difference between the coordinate value and its desired value: $d = r - c$, where *r* is the coordinate and *c* is the center or target value. The constraint is then more like a dynamic constraint, in which the coordinate is at the target value at the end of the optimization. These extra potential energy terms are listed in the output and used to determine geometry convergence, but they are not included in the final energy.

Harmonic constraints must be set in the **coord** section of the input file, and can be set on the Cartesian position of an atom or on any bond length, angle, or dihedral angle. To set a harmonic constraint, adding `#hc` or `#HC` after the coordinate, followed by the force constant. If you want to specify a region of zero potential, add the half-width of this region, *a*, after the force constant. If you want to specify a target value *c* for an internal coordinate, it must follow the radius *a*. A target value cannot be specified for a Cartesian harmonic constraint. The units of the force constant, the half-width, and the target coordinate values are specified by the gen section keywords **iunit** and **eunit**.

The Cartesian position is specified by a single atom label, as in the following example.

```
&coord
C1 #hc 10.0
&
```

The following example specifies a harmonic constraint on a bond length, with a force constant of 10.0 kcal mol^{-1} Å^{-2}, a width of 0.1 Å, and a target bond length of 1.5 Å:

```
&coord
C1 C2 #hc 10.0 0.1 1.5
&
```

In this constraining potential, the bond length can vary freely between 1.4 Å and 1.6 Å, but the energy rises if it goes outside this region.

Assuming units of kcal/mol for energy and degrees for angles, the following ranges are recommended for harmonic constraints of *nonbonded* atoms:

- bond stretch: 10 - 1000
- angle bend or torsion: 0.0001 to 0.1

Values for *bonded* atoms can be somewhat larger, but much higher values can begin to freeze the coordinate, while values as large as 10 for a torsional constraint can actually introduce errors into the interconversion of the internal coordinate hessian and the Cartesian hessian, and therefore should be avoided.

4.2.3 Applying Constraints by Using Variables

When you define a set of coordinates, bond lengths or bond angles in terms of a variable, these coordinates, bond lengths or bond angles are constrained to be the same during a geometry optimization. The variable becomes the optimization parameter, and the coordinates, bond lengths or bond angles are set to the value of the variable at each optimization step.

The effect of using variables depends upon the format of your input structure. If your input structure is in Z-matrix format, you can set several bond length or angle coordinates to the same variable. For input in Cartesian format, you can use variables to keep several atoms within the same plane during an optimization by setting their coordinates along one axis to the same variable.

To use variables to set coordinate values from the Edit Job dialog box, first type the variable name (`zcoor`, for instance) where you would normally type the corresponding numerical value for each relevant coordinate. You can put a + or – sign immediately before any variable, and you may use several variables if you want. When you have entered the full geometry, add one or more lines setting the variables.

For instance, in a geometry optimization using the following Cartesian input

```
O   0.000000     0.000000  -0.113502
H1  0.000000     ycoor      zcoor
H2  0.000000    -ycoor      zcoor
ycoor=0.753108   zcoor=0.454006
```

the H atoms remain in the same *xy* plane and the same *xz* plane: the molecular symmetry is preserved.

Optimizations are run without symmetry when Cartesian input with variables is used.

4.2.4 Applying Dynamic Constraints

Dynamic constraints, also called “soft” or “harmonic” constraints, are implemented by means of Lagrange multipliers. A dynamic constraint on a geometric coordinate is met gradually during the course of an optimization. One advantage of using a dynamic constraint on a variable is that you can choose a value that is different from its current value. For example, if you have a structure whose conformation you want to change, and you know that changing a particular torsional angle would cause parts of the molecule to collide if the torsional angle’s value were suddenly imposed, you can instead specify the desired value for the torsion as a dynamic constraint. The optimizer changes the torsion gradually during the optimization, so that the final torsional angle is as close as possible to the desired torsional angle.

Defining dynamic constraints is handled in the **coord** section, which is described in Section 9.4 on page 201. You can make a constraint dynamic by editing the input file or by using the controls in the Optimization tab.

To make a constraint in the Constraints table dynamic, enter the value that you want the constraint to converge on in the Target Value column for that constraint. This value is added to the variable definition in the **coord** section.

4.3 Transition-State Optimizations

To perform transition-state searches with Jaguar, you can use either a simple quasi-Newton method that searches for the transition state nearest to the initial geometry, or quadratic synchronous transit (QST) methods, also known as synchronous transit quasi-Newton (STQN) searches. We generally recommend using QST methods any time you can provide both reactant and product geometries and a transition state guess. For recommendations, see Section 7.4 on page 173.

It is also recommended that you change the geometry convergence criteria to standard (the default is flexible—see Section 4.1.1 on page 92 for more information), particularly if you plan to perform an IRC calculation after the transition state search. You can do this by setting `geoconv_mode=standard` in the **gen** section of the input file. Using the standard criteria ensures that all criteria are met.

To set up a transition-state search, choose Applications → Jaguar → Transition State Search[18] in the main window. You can set optimization parameters that are not unique to transition-state searches in the Optimization tab. The Transition state tab contains controls for settings that are specific to transition-state searches, including selection of the reactant and product geometries. It also contains controls for the basis set, charge, and multiplicity of the system.

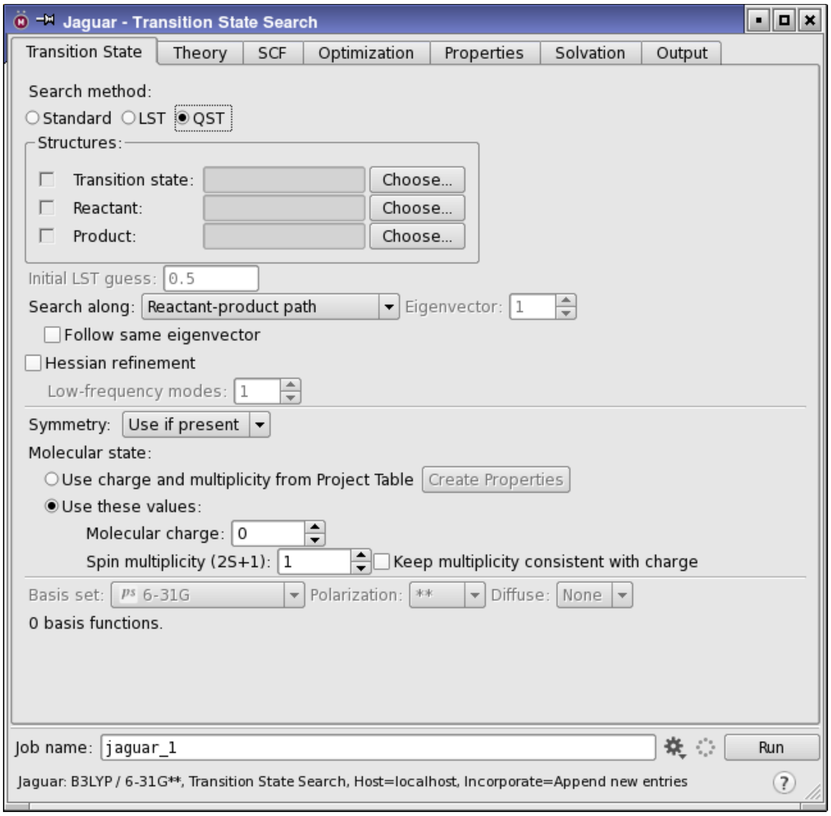
This section describes various transition-state search options. For information on general settings that are useful for all types of geometry optimizations, see Section 4.1 on page 91.

4.3.1 Transition-State Search Method

The first choice listed in the Transition State tab is the Search method, which can be set to Standard,[19] LST,[20] or QST.[20] The default choice is Standard because it does not require more than one input geometry, but if you can provide product and reactant geometries, we recommend selecting LST. If you also have a good guess for the transition state, select QST.

Both the LST option and the QST option set up a QST-guided search. If you select LST, Jaguar generates a transition-state guess by interpolating between the reactant and product geometries. By default, this linear synchronous transit (LST) transition-state guess is midway between the reactant and product geometries. This choice is indicated by the default value of 0.5 for the Initial LST guess[21] setting. To pick a transition-state guess closer to the reactant geometry, change this setting to a number between 0 and 0.5; to pick a guess closer to the product geometry, set the Initial LST guess value to a number between 0.5 and 1.0.

18. Keyword **igeopt** = 2 in the **gen** section.
19. Keyword **iqst** = 0 in the **gen** section.
20. Keyword **iqst** = 1 in the **gen** section.
21. Keyword **qstinit** in the **gen** section.



Jaguar - Transition State Search

Transition State | Theory | SCF | Optimization | Properties | Solvation | Output

Search method: Standard LST [✓ QST]

Structures:

- ☐ Transition state: Choose...
- ☐ Reactant: Choose...
- ☐ Product: Choose...

Initial LST guess: 0.5

Search along: Reactant-product path Eigenvector: 1

☐ Follow same eigenvector

☐ Hessian refinement

Low-frequency modes: 1

Symmetry: Use if present

Molecular state:

Use charge and multiplicity from Project Table Create Properties

[✓ Use these values:]

Molecular charge: 0

Spin multiplicity (2S+1): 1 ☐ Keep multiplicity consistent with charge

Basis set: 6-31G Polarization: ** Diffuse: None

0 basis functions.

Job name: jaguar_1 Run

Jaguar: B3LYP / 6-31G**, Transition State Search, Host=localhost, Incorporate=Append new entries

Figure 4.2. The Transition State tab.

For the first few steps of a QST-guided search, the optimizer is restricted to searching along the circular curve connecting the reactant, transition-state guess, and product structures. This restriction prevents the optimizer from being led far astray by the inaccuracies of the guess Hessian, and prevents it from exploring transition states that do not correspond to the reaction of interest. During these steps, the optimizer approaches the maximum-energy structure along the reactant-to-product curve, and also greatly improves the Hessian.

Once it has obtained the improved Hessian and transition-state guess, the optimizer removes the requirement that the search must be along the curve between the structures. For all subsequent steps in the search, the optimizer follows the Hessian eigenvector that is most similar to the tangent of the circular curve. (If no Hessian eigenvector is sufficiently similar to the tangent to the curve, the optimizer follows the lowest eigenvector.)

If you have a fairly good transition-state guess but cannot provide reactant or product structures, you can still use the standard, non-QST method. This optimizer attempts, at each step, to maximize the energy along the lowest-frequency eigenvector of the Hessian and to minimize along all other coordinates. This process is well-defined and straightforward when the Hessian has exactly one negative frequency, indicating that the structure is near a saddle point. The negative-eigenvalue mode, which is sometimes known as the *reaction coordinate*, is referred to as the *transition vector* in this chapter.

4.3.2 Specifying Structures for the Reaction

Structure selection for transition state searches operates differently from structure selection for other tasks. You select structures for the reaction from the entries in the Project Table using the controls in the Structures section of the Transition State tab. When you choose a search method, the controls for selecting the structures that you must provide become available. Click Choose to open an entry selector and select the entry. To display any of these structures in the Workspace, click the inclusion button to the left of the row. Click the button again to undisplay the structure.

The structures that you specify for the reaction must have the atoms listed in the same order. One way of ensuring the correct ordering is to build one structure in Maestro, then modify it to generate the other one or two. If the atoms are not in the correct order, you must edit the geometries to place them in the correct order.

For best results, the reactant and product structures should not be radically different from the transition state. For instance, to find the transition state in a bond-breaking reaction, it would be better to provide a product structure in which the breaking bond was fairly long and weak than a true minimum-energy structure in which the bond had completely dissociated.

The reactant and product structures are added to the **zmat2** and **zmat3** sections of the input file for QST searches, and to the **zmat** and **zmat2** sections for LST searches. If you want to edit the structures, you can do so in the Edit Job dialog box. When you select Structure in this dialog box, the three structures are available in separate tabs, labeled Reactant, Product, and Transition State. If you want to use the same Z-matrix for all geometries, choose Standardize Z-matrix format from the Structures menu. You can then set any variables to the desired values.

If you want to apply constraints in the search, you must add them to the “main” geometry. The main geometry for the job is the topmost in the Structures section: the transition state for standard and QST searches, and the reactant for LST searches. To add constraints, click the inclusion button for the main geometry, then add the constraints in the Optimization tab (see Section 4.2 on page 96).

It is highly recommended that you add a **connect** section to the input file, in which all bonds that are broken or formed are specified. This ensures that the bonds that are broken or formed are included in the internal coordinates used to drive the search, and usually results in faster convergence and more reliable location of the transition state. For an example, see Section 5.1

of the *Jaguar Quick Start Guide*. See Section 9.4 on page 201 for information on the **connect** section.

4.3.3 Specifying the Symmetry, State, and Basis Set

The lower part of the Transition State tab provides controls for the molecular charge, the spin multiplicity, the application of molecular symmetry, and the basis set. You can enter the charge[22] and spin multiplicity[23] in text boxes, or you can use values that are listed in the Project Table. If you choose the latter, you can create the corresponding properties and enter values for them if they don't exist. You can choose the basis set[24] from the Basis set option menu, and specify polarization and diffuse functions. See Section 3.2 on page 33 for information on basis sets.

By default, Jaguar takes advantage of molecular symmetry in order to obtain CPU savings. Both Abelian and non-Abelian point groups are recognized. You can select whether to use the full symmetry,[25] Abelian symmetry (D2h and subgroups),[26] or turn the use of symmetry off[27] in the Symmetry option menu. For information on how to make sure the symmetry of your input structure is treated as you expect, see Section 2.7.2 on page 23.

4.3.4 Searching Along a Particular Hessian Eigenvector

If you are using the standard (non-QST-guided) transition-state optimization method, you can specify a Hessian eigenvector for the optimizer to follow. The choices of eigenvector are available from the Search along option menu: Lowest Hessian eigenvector[28] (the default), the Lowest non-torsional mode,[29] the Lowest bond-stretch mode,[30] or a User selected eigenvector. You also have access to these search directions if you are doing an LST search, in addition to the Reactant-product path.[31]

Under certain circumstances, you might want to direct your transition-state search using these options, rather than having the optimizer simply minimize along the lowest Hessian eigenvector found for each iteration. The Lowest non-torsional mode and Lowest bond-stretch mode options can be useful for steering the optimizer to a particular type of transition state—for instance, for a study of a bond-breaking reaction, you can avoid converging to a torsional transition state by choosing Lowest bond-stretch mode.

22. Keyword **molchg** in the **gen** section.
23. Keyword **multip** in the **gen** section.
24. Keyword **basis** in the **gen** section.
25. Keyword **isymm** = 8 in the **gen** section.
26. Keyword **isymm** = 8 and **idoabe**=1 in the **gen** section.
27. Keyword **isymm** = 0 in the **gen** section.
28. Keyword **itrvec** = 0 in the **gen** section.
29. Keyword **itrvec** = –1 in the **gen** section.
30. Keyword **itrvec** = –2 in the **gen** section.
31. Keyword **itrvec** = –5 in the **gen** section.

If you know the index number of the eigenvector along which you would like to minimize (a particular bond stretch, for instance), you can make the optimizer follow that eigenvector by choosing User selected eigenvector and specifying the eigenvector number in the Eigenvector text box.[32] You can identify the index number by running one geometry optimization iteration (see Section 4.1.1 on page 92 for more information) and examining the output summary of the Hessian eigenvectors, which indicates the dominant internal coordinates and their coefficients for each eigenvector.

You can also determine the eigenvector to follow from a vibrational frequency calculation. However, because different mass-weighting schemes are used in vibrational frequency calculations and transition state searches, it is possible for the two calculations to produce a different number of negative Hessian eigenvalues for a non-stationary point. If you select the mode to follow based on the results of a vibrational frequency calculation, you might discover that the mode actually followed is different, and that the number of negative Hessian eigenvalues has changed. There are two ways around this problem. The preferred way is to use the QST-guided algorithm, which automatically selects the mode that best connects the reactant and product geometries with the transition state guess geometry. In this case there is no need to select an eigenvector (or to set the **itrvec** keyword). If you do not wish to use the QST method, then you should check the output file from your transition-state search job while it is running to ensure that the desired mode was selected. For each negative Hessian eigenvalue, Jaguar prints out the internal coordinates that dominate the corresponding eigenvector. If you find that the wrong mode is being followed, kill the job and select the desired eigenvector index, either in the GUI or by setting **itrvec** in the **gen** section of the input file.

The order and the character of the eigenvectors can change during an optimization. To ensure that the optimization follows the eigenvector that most closely correlates with the one chosen in the previous iteration, select Follow same eigenvector.[33] Otherwise, the optimization follows the eigenvector of the same index number as the previous iteration.[34]

4.3.5 Refinement of the Initial Hessian

The quality of the Hessian in the initial steps of a transition-state optimization can have a marked effect on the speed of the job, since the Hessian controls the direction the optimization takes on a potential energy surface in the search for an appropriate saddle point. The QST-guided transition search method effectively refines the Hessian automatically in the first steps along the circular curve connecting the reactant, transition-state guess, and product.

With the standard, non-QST-guided optimization method, if a coordinate with a negative force constant (Hessian eigenvalue) exists, it is critical for this transition vector to be properly identi-

32. Keyword **itrvec** > 0 in the **gen** section, where **itrvec** is the eigenvector number for the selected eigenvector.
33. Keyword **ifollow** = 1 in the **gen** section.
34. Keyword **ifollow** = 0 in the **gen** section.

fied as efficiently as possible, since it leads to the transition state. Consequently, for transition-state searches with the standard optimizer, when the initial Hessian chosen is a guess Hessian (one not calculated numerically or read from a restart file), it can be helpful to refine the Hessian during the calculation before using it to compute any new geometries.

Hessian refinement is especially likely to improve transition-state optimizations that employ eigenvector following (described in Section 4.3.4 on page 103), because any eigenvector selected for following should be accurate enough to be a reasonable representation of the final transition vector.

To refine an initial Hessian, select Hessian refinement in the Transition State tab, then enter the number of low-frequency Hessian eigenvectors to be used in the refinement in the Low-frequency modes[35] text box. By default, no eigenvectors are used—that is, no refinement is performed unless the input specifies particular coordinates for refinement. Hessians can be refined using any number of the lowest-frequency Hessian eigenvectors. Refinements involve SCF and gradient calculations for displacements along these modes, which allow more accurate information about the most important modes to be included in the Hessian.

You can also specify particular coordinates for Hessian refinement. If you put an asterisk (*) after a coordinate value, Jaguar computes the gradient of the energy both at the original geometry and at a geometry for which the asterisk-marked coordinate has been changed slightly, and uses the results to refine the initial Hessian to be used for the optimization. To request refinement of a coordinate whose value is set using a variable, add an asterisk to the end of the variable setting in the line at the end of the geometry input that defines the variables. For instance, either of the following two input geometries would use both O–H bonds and the H–O–H angle in a Hessian refinement:

```
O1
H2   O1   1.1*
H3   O1   1.1*   H2   108.0*
```

or

```
O1
H2   O1   ohbond
H3   O1   ohbond   H2   108.0*
ohbond = 1.1*
```

Molecular symmetry or the use of variables, either of which may constrain several coordinate values to be equal to each other, can reduce the number of coordinates actually used for refinement. For example, for the second water input example shown above, only two coordinates will actually be refined: the O–H bond distance, which is the same for both bonds, and the H–O–H angle. The same would be true for the first example if molecular symmetry were used.

35. Keyword **nhesref** = 3 in the **gen** section.

4.4 Geometry Scans

Geometry scans are a series of jobs run with input files that vary only in the value of one or more variables used to define an internal or Cartesian coordinate in the input structure. You can perform a “relaxed scan,” finding minimum-energy geometries while holding one or more coordinates fixed to various values, or a “rigid scan”, varying only the scan coordinates. To scan over a coordinate, you must first define the coordinate as a variable, then define the values that the coordinate must take. The maximum number of points per coordinate is 360. You can set up scans from Maestro or you can create an input file with the appropriate sections.

When an internal coordinate is scanned, the last atom in the coordinate definition is the atom that is moved, along with any other atoms that are attached to it. Jaguar will not allow you to scan the distance between two atoms in a ring. This is because it is not possible to change the distance coordinate without changing some other coordinate. Bond angles can be scanned through 0° and 180°, though the scan may not start at these values.

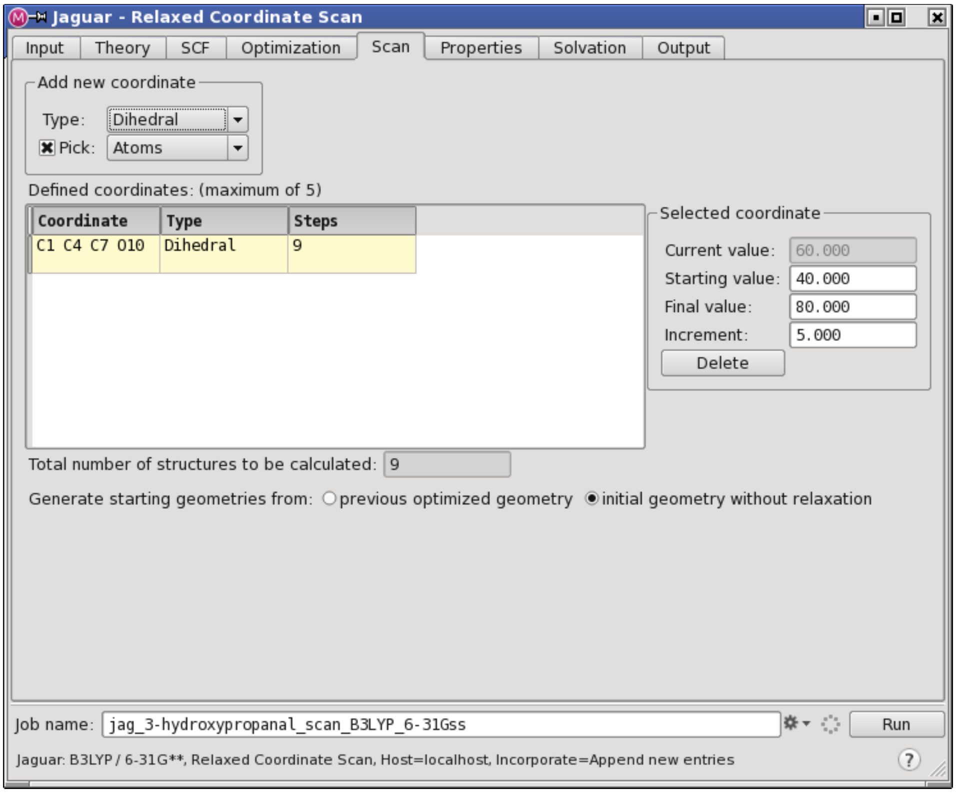
4.4.1 Setting up Scans in Maestro

To set up a scan from Maestro, choose Relaxed coordinate scan or Rigid coordinate scan from the Jaguar submenu of the Applications menu. The Jaguar panel opens with the tabs that are relevant to the chosen coordinate scan type. After you have made settings in the other tabs, you can set up the coordinates and their values in the Scan tab.

Coordinates can be added to the scan using the Add New Coordinate section and picking atoms or bonds in the Workspace. To set up a coordinate, first choose a coordinate type from the Type option menu. The allowed coordinate types and the number of atoms to pick for each are:

Cartesian-X	X coordinate of an atom. Pick one atom.
Cartesian-Y	Y coordinate of an atom. Pick one atom.
Cartesian-Z	Z coordinate of an atom. Pick one atom.
Distance	Distance between two atoms. Pick two atoms or one bond.
Angle	Angle between three atoms. Pick three atoms or two bonds.
Dihedral	Dihedral angle between four atoms. Pick four atoms or three bonds.

When you have chosen a coordinate type, select Pick and choose an object from the Pick option menu. You can pick Atom for Cartesian coordinates, and Atom and Bond for all other types of coordinates. If you pick atoms for a distance, angle, or dihedral, they need not be bonded to each other. The atoms are marked in the Workspace as you pick them, and each coordinate is marked in the Workspace and entered in the Defined coordinates table as it is completed. The last atom picked is the atom that is moved during the scan, with its attachments.



Add new coordinate

Type: Dihedral

[✓ Pick:] Atoms

Defined coordinates: (maximum of 5)

Coordinate	Type	Steps
C1 C4 C7 O10	Dihedral	9

Selected coordinate

Current value: 60.000

Starting value: 40.000

Final value: 80.000

Increment: 5.000

Total number of structures to be calculated: 9

Generate starting geometries from: ○ previous optimized geometry [● initial geometry without relaxation]

Job name: jag_3-hydroxypropanal_scan_B3LYP_6-31Gss

Jaguar: B3LYP / 6-31G**, Relaxed Coordinate Scan, Host=localhost, Incorporate=Append new entries

Figure 4.3. The Scan tab.

The Defined coordinates table displays information on the scan coordinates: the definition, the type, and the number of steps. The total number of structures to be calculated is reported below the table, and is the product of the numbers in the Steps column.

To set the values for the coordinate, select the row for the coordinate in the table, then enter the starting value, the final value, and the increment in the Selected Coordinate section. The Current value and Number of steps text boxes are noneditable. The number of steps is calculated from the values you provide.

The starting geometry for each scan point in a relaxed coordinate scan can be generated from the initial geometry or from the previous scan point, by selecting one of the Generate starting geometry from options. If you select previous optimized geometry, the scan points must be run sequentially, but the starting geometry for the optimization of the coordinates that are not being scanned is likely to be a lot better than if you select initial geometry without relaxation. The starting geometry for rigid scans is always generated from the initial geometry, as there is no relaxation to take into account.

Once you have defined all the coordinates and their values, you can start the job, or write out the input file for the job. The scan can be distributed over multiple processors if you use the initial geometry to generate the starting geometries for the scan points.

4.4.2 Setting up Input Files for Scans

To create an input file for a coordinate scan, you set up an input file with the coordinates defined as variables and the variable values defined as described below. For a relaxed scan, the job must be a geometry optimization; for a rigid scan, it must be an energy calculation.

You can define a variable in the geometry input (as described in Section 2.4.5 on page 16), or you can define variables in the **coord** section. Using the **coord** section allows you to define variables that are not part of the Z matrix geometry input. To define a variable in the **coord** section, add the variable name after the # sign in the coordinate definition. For example, the following input file section defines a coordinate HH as the distance between H1 and H2.

```
&coord
H1 H2 # HH
&
```

The values taken by a variable are defined in a **zvar** section.

To specify the values that a variable will take in a scan, you can assign a list of values to the variable in the format `at` *number-list*, or you can assign the initial value as either *number* or `from` *number,* and use two specifications from the following list, in the order given:

- `to` *number*—specify the final value of the coordinate
- `by` *number*—specify the step size
- `in` *integer*—specify the number of steps

Here, *integer* means an appropriate integer and *number* means an appropriate real number. If you specify the initial and final values, they are always among the values set. For example, varying a coordinate from 0 to 120 by a step size of 30 takes 5 steps: 0, 30, 60, 90, and 120.

As an example, to vary the angle HCCH over the values {0, 30, 60, 90, 120, 150, 180}, you could use any one of the following lines:

```
HCCH = from 0 to 180 by 30
HCCH = 0 to 180 in 7
HCCH = from 0 by 30 in 7
```

You can also set a coordinate to a set of specific values using the word `at`. With this format, the values of the scanned coordinate do not have to be evenly spaced. For example, this line would vary the angle HCCH over the values {0, 30, 60, 70, 80, 90, 120, 150}:

```
HCCH = at 0 30 60 70 80 90 120 150
```

You can define up to five scan coordinates at once. The first scan coordinate will be in the innermost loop—that is, the scanner will run through all values of the first scan coordinate before updating the others, and so on, finally looping last over the last scan coordinate.

For each geometry in the scan, the default initial guess for the wave function and the default initial Hessian are taken from the previous geometry. You can change this behavior using the **scanguess** and **scanhess** keywords in the **gen** section of the input file.

By default, the scan increments are applied to the final geometry from the previous scan step, thus minimizing the number of geometry optimization steps in a relaxed scan required to optimize the new structure. Sometimes this can have the effect of biasing a relaxed scan such that a number of scan points optimize to the same local minimum for other parts of the structure. In these cases you can set **scangeo**`=1` so that the scan increments are applied to the input geometry. The danger here is the possibility of steric clashes, especially for torsional scans around bonds connecting bulky substituents.

4.4.3 Restarting Scans

Single-processor scan jobs store a record of their progress, by writing **scanindex**=*n* in the **gen** section of the restart file for scan jobs. If you want to restart a scan job that stopped prematurely, you can use the restart file to restart the scan.

Distributed scan jobs run the scan as a set of single-point calculations, and collate the results when each point is done. If any subjob fails, you should delete the output file for the failed subjobs, and then you can restart the distributed scan job. The completed points are skipped (on the basis of the existence of the output file), and the remaining points are run.

4.4.4 Scan Results

The results of scan jobs started from Maestro are incorporated into the Project Table. Each scan point is added as a separate entry to the table, with the energy and scan coordinates as properties. The properties are labeled with an r for distance coordinates, a for angles, and d for dihedrals. Cartesian coordinates are labeled with the atom name and x, y, or z. If you want to submit a particular scan point to Jaguar for refinement, you can select the corresponding entry as the source of job input. You can use the plot facility to display plots of the energy as a function of a particular coordinate, for example.

If the calculation for any point fails to converge, this point is skipped and the calculation proceeds to the next point. The failure is noted in the output.

For one- and two-dimensional scans, Jaguar writes `.grd` files, which can be read by Maestro and displayed in the Plot of One-Coordinate Scan and Plot of Two-Coordinate Scan panels, which you open from the Plot Coordinate Scan submenu of the Tools menu. These panels display interactive plots and interactive contour plots, respectively. For more information on these panels, see Section 11.2 of the *Maestro User Manual*. You can use any kind of scan except `at` scans, i.e. scans that supply a list of coordinate values.

If you generated surfaces in the scan job, the surfaces are incorporated along with the structures at each point.

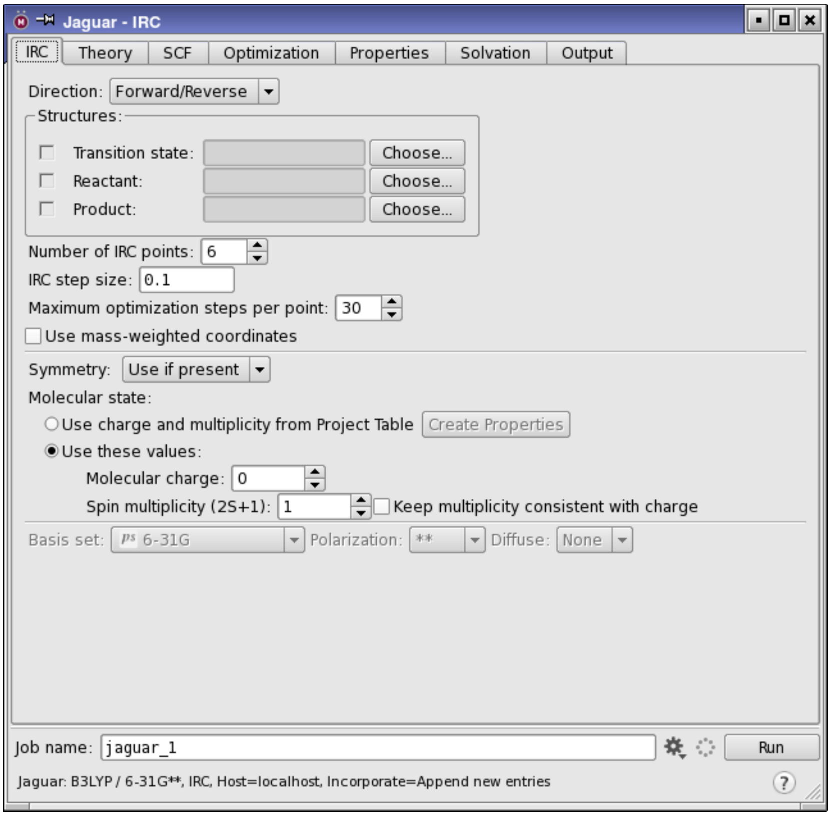
If you want to make use of the Jaguar input files for individual scan points, you can extract them from the file *jobname*`.steps.in`, which is written to the working directory whenever a scan is performed. This file contains the geometry specifications for each geometry in the scan, along with the calculated energies, keywords, and forces. Each input section in the *jobname*`.steps.in` file terminates with the string `END_OF_INPUT`., which you can use to determine where to split the file.

4.5 Intrinsic Reaction Coordinate Calculations

Intrinsic Reaction Coordinate (IRC) calculations can be used to check that the given transition state is the expected transition state for the reaction of interest. IRC calculations start at a transition state and move downhill in energy along the reaction path toward a minimum of the potential energy surface, calculating a series of points in which all geometric variables orthogonal to the path are optimized. The points should lead towards the reactant and the product structures, but as a fixed number of points is calculated, the end points might not actually be reached.

Jaguar has a variation of the IRC method that can be used for inexpensively verifying the transition state. In this variation, the first IRC point is calculated on either side of the transition state. The constraint to the reaction path is then removed and the geometry obtained from each IRC point is optimized to find a minimum. There is of course no guarantee that the minimum corresponds to the end point of the reaction, as the constraints to the path have been relaxed. However, if the minimum found does not correspond to the expected end point, it is possible that the potential energy surface is more complicated than expected, or that the transition state is not the correct one. In this case, reintroducing the constraint by computing a full IRC path is advisable. The variation can be requested by adding `three_pt_irc=1` to the **gen** section. You should also ensure that the step size in the IRC path is large enough to produce a substantial gradient at the first IRC point, otherwise the subsequent optimization may not produce the desired results.

IRC scans have been implemented in Jaguar using the methods described in Ref. 235. The implementation includes both IRC and minimum energy path (MEP) calculations. The difference between the two is that the reaction coordinate for the IRC path is mass-weighted, whereas the reaction coordinate for the minimum energy path is not.



Direction: Forward/Reverse

Structures:

- [ ] Transition state:
- [ ] Reactant:
- [ ] Product:

Number of IRC points: 6

IRC step size: 0.1

Maximum optimization steps per point: 30

- [ ] Use mass-weighted coordinates

Symmetry: Use if present

Molecular state:

○ Use charge and multiplicity from Project Table

● Use these values:

Molecular charge: 0

Spin multiplicity (2S+1): 1 [ ] Keep multiplicity consistent with charge

Basis set: 6-31G Polarization: ** Diffuse: None

Job name: jaguar_1

Jaguar: B3LYP / 6-31G**, IRC, Host=localhost, Incorporate=Append new entries

Figure 4.4. The IRC tab.

IRC calculations can be set up from Maestro or by adding keywords to an input file (see Section 9.5.12 on page 228). To set up an IRC or minimum energy path calculation from Maestro, you must first perform a transition-state calculation and read in the restart file, then choose Reaction Coordinate from the Jaguar submenu of the Applications menu.

The lower part of the IRC tab provides controls for the molecular charge, the spin multiplicity, the application of molecular symmetry, and the basis set. You can enter the charge and spin multiplicity in text boxes, or you can use values that are listed in the Project Table. If you choose the latter, you can create the corresponding properties and enter values for them if they don't exist. You can choose the basis set from the Basis set option menu, and specify polarization and diffuse functions. See Section 3.2 on page 33 for information on basis sets.

By default, Jaguar takes advantage of molecular symmetry in order to obtain CPU savings. Both Abelian and non-Abelian point groups are recognized. However, if the transition state has a higher symmetry than other points on the reaction path, you should turn symmetry off, or lower it. You can select whether to use the full symmetry,[36] Abelian symmetry (D2h and subgroups),[37] or turn the use of symmetry off[38] in the Symmetry option menu. For information on how to make sure the symmetry of your input structure is treated as you expect, see Section 2.7.2 on page 23.

The calculation requires an accurate Hessian for the transition state. You can precalculate the Hessian and read it in from the restart file, or calculate the Hessian as part of the IRC or MEP calculation. To read the Hessian, choose Other[39] from the Initial Hessian option menu in the Optimization tab; to calculate the Hessian, choose Quantum-mechanical.[40] Do not use any of the other options from this menu, because it is important to have an accurate Hessian.

If you precalculate the Hessian and read it in, the symmetry of the transition state is used for the entire calculation. If the IRC path breaks symmetry, you should turn symmetry off (in the IRC tab). If you calculate the Hessian as part of the IRC or MEP calculation, symmetry is turned off in the Hessian evaluation and remains off for the remainder of the run.

IRC and MEP calculations can be run in either direction from the transition state. You can select the direction of the scan from the Direction option menu in the IRC tab.[41] The direction of the reaction coordinate can be defined by supplying the reactant structure and the product structure in the Structures section of the IRC tab. These geometries are selected in the same way as for a transition state search—see Section 4.3.2 on page 102. The same rules for atom numbering in the reactant and product structures apply as for transition state searches. If you do not supply the reactant and product structures, Jaguar defines the forward direction using the rules described in Section 9.5.12 on page 228.

You can run an IRC scan from a point that is not a transition state, by selecting Downhill from the Direction option menu,[42] and providing the geometry as if it were a transition state. For these calculations, you do not need an initial guess of the Hessian. The scan follows the gradient from the initial point. Downhill mode is set automatically in the restart file for IRC jobs.

The default calculation generates 6 points in both forward (toward the products) and backward (toward the reactants) directions from the transition state, at an interval of 0.1 in the reaction coordinate. You can enter the number of points in the Number of IRC points text box,[43] and the step size in the IRC step size text box.[44] You can also limit the number of geometry optimiza-

36. Keyword **isymm** = 8 in the **gen** section.
37. Keyword **isymm** = 8 and **idoabe**=1 in the **gen** section.
38. Keyword **isymm** = 0 in the **gen** section.
39. Keyword **inhess** = 2 in the **gen** section.
40. Keyword **inhess** = 4 in the **gen** section.
41. Keyword **ircmode** in the **gen** section.
42. Keyword **ircmode** = downhill in the **gen** section.
43. Keyword **ircmax** in the **gen** section.
44. Keyword **ircstep** in the **gen** section.

tion steps at each point in the Maximum optimization steps per point text box.[45] This value overrides any value set in the Optimization tab.

Finally, for an IRC calculation, Use mass-weighted coordinates must be selected.[46] For an MEP calculation, this option must be deselected.[47]

When the calculation finishes, the structures at the IRC (or MEP) points are automatically incorporated as separate entries in the Project Table, and the reaction coordinate is incorporated as a property. You can then sort the entries based on this property, and display them in sequence using the ePlayer. For an example, see Section 5.3 of the *Jaguar Quick Start Guide*.

45. Keyword **ircmxcyc** in the **gen** section.
46. Keyword **irc** = 2 in the **gen** section.
47. Keyword **irc** = 1 in the **gen** section.

Jaguar Workflows

Some tasks that you may want to do involve multiple steps, such as calculating a transition state and generating an IRC path. The main Jaguar panels provide access to the basic functions, but you may have to run several calculations to complete your task.

Jaguar provides batch scripts as a means to perform a sequence of steps, which constitute a workflow. The input settings may be predefined for the particular task, and you only need to provide the input files, or you may need to specify a few settings. The mechanism of running batch scripts is described in Section 2.10 on page 26.

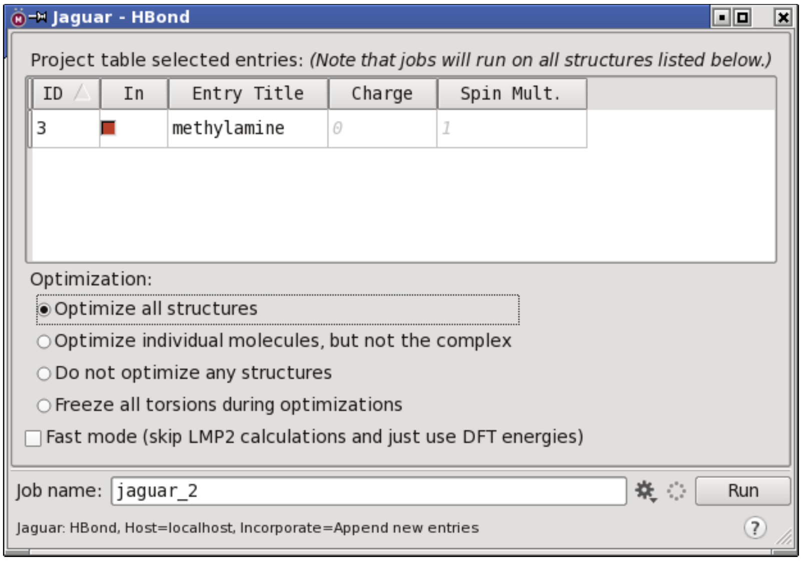
Jaguar also has panels in Maestro for some common workflows. The panels allow you to make settings and then run the batch script that encodes the workflow. This chapter describes these workflows.

5.1 Binding Energies of Hydrogen-Bonded Complexes

To calculate an accurate binding energy for an intermolecular hydrogen-bonded complex, you can use the Hydrogen Bond workflow, which is encoded in the Jaguar batch script `hydrogen_bond.py` (see Section 11.3.2 on page 330 for instructions) with a Maestro interface. In this workflow, the binding energy is calculated using two correlation-consistent basis sets (cc-pVTZ(-f) and cc-pVQZ(-g)) and LMP2 theory, including corrections for basis set superposition error. The final correction to the binding energy uses two parameters from a fit of the energies to reference energies that were obtained from CCSD(T) calculations extrapolated to the basis set limit. The binding energies are accurate to within 0.5 kcal/mol when compared to the extrapolated CCSD(T) energies.

The protocol is based on that described in Ref. 28, except that all geometry optimizations are carried out using X3LYP/6-31G** instead of LMP2/cc-pVTZ(-f). The X3LYP density functional gives very good geometries for hydrogen-bonded complexes and is much faster than LMP2. The energy calculations, however, are still carried out using LMP2 with the cc-pVTZ(-f) and cc-pVQZ(-g) basis sets.

An option is provided to use the DFT energies rather than the LMP2 energies. In a test for a series of ten hydrogen-bonded complexes, the RMS error using the DFT energies was 0.72 kcal/mol, while the RMS error using the default LMP2 method was 0.34 kcal/mol. Other options are provided to control the optimization process.



Project table selected entries: *(Note that jobs will run on all structures listed below.)*

ID	In	Entry Title	Charge	Spin Mult.
3	🟥	methylamine	0	1

Optimization:

- (●) Optimize all structures
- ( ) Optimize individual molecules, but not the complex
- ( ) Do not optimize any structures
- ( ) Freeze all torsions during optimizations

☐ Fast mode (skip LMP2 calculations and just use DFT energies)

Job name: jaguar_2

Jaguar: HBond, Host=localhost, Incorporate=Append new entries

Figure 5.1. The Jaguar panel for hydrogen bond calculations.

To run a hydrogen bond binding energy calculation, select the desired structures in the Project Table or Entry List, and choose Applications → Jaguar → Hydrogen Bond in the main window.

The panel that opens allows you to set charge and spin in the entry table, and has controls for choosing which structures to optimize:

- Optimize all structures—Optimize the complex and the two molecules that make up the complex.
- Optimize individual molecules, but not the complex—Optimize the two molecules that make up the complex, but not the complex itself, which retains its input geometry.
- Do not optimize any structures—Use the input geometry for all calculations: do not perform any optimizations.
- Freeze all torsions during optimizations—Optimize the complex and the two molecules that make up the complex, but freeze all torsions (dihedral angles) in the optimization and optimize only the bond lengths and bond angles. This helps convergence with weakly-bound systems.

You can also select Fast mode to use the DFT energies instead of the LMP2 energies.

When the job finishes, the optimized structure of the complex is incorporated into the project, with the binding energy, in kcal/mol, as a property.

5.2 Fukui Functions

Fukui functions represent the changes in the molecular electron density upon addition or removal of charge. While the atomic Fukui indices (see Section 3.9.7 on page 68) are convenient to use because they are simply scalar values for each atom, they are somewhat less well-defined than Fukui functions because they are only based on the MO coefficients for the frontier orbitals for the *N*-electron system. Thus, they do not consider relaxation of the full charge density when the molecular charge is changed. In contrast, Fukui functions account for relaxation of the density and are dependent upon actual charge perturbations. But because they are functions, rather than discrete numbers, they must be visualized as isosurfaces or as property maps in Maestro.

Fukui functions are calculated by Jaguar in a finite-difference approach as follows:

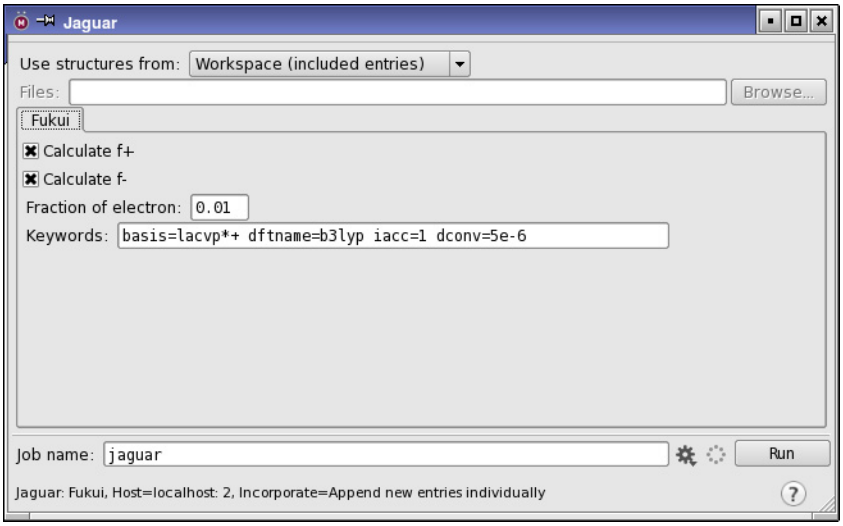
$$f^{+} = (\rho^{N+\delta}(r) - \rho^{N}(r))/\delta$$

$$f^{-} = (\rho^{N-\delta}(r) - \rho^{N}(r))/\delta$$

where *N* is the number of electrons in the reference state of the molecule, and δ is a fraction of an electron. The value of δ used by default in Jaguar is 0.01 electron. This value has been shown to give results which are comparable to those obtained with a differential approach [241] for small molecules. Furthermore, when using DFT, the finite difference method used by Jaguar retains the effect of the N-derivative of the DFT exchange-correlation potential, whereas the differential method proposed in Ref. 241 neglects this. For small molecules the neglect seems justifiable, but for larger molecules it may not be. In any case, the value of δ may be controlled using the **fukui_delta** keyword.

The calculation of both Fukui functions requires explicit calculation of three different charge states for the molecule. The Python script, `fukui.py` runs all three calculations and generates the Fukui functions, which can be visualized in Maestro as a surface.

To set up a Fukui function calculation from Maestro, choose Applications → Jaguar → Fukui Functions. The panel that opens allows you to select the input structures and make some keyword settings. Because the calculation of Fukui functions involves changes in the electron density, no other properties may be calculated except for atomic charges (the atomic Fukui indices for the *N*-electron system are calculated automatically), and you should use pre-optimized structures as input, since geometry optimization cannot be combined in the same job as Fukui functions. By default, B3LYP/LACVP*+ is used for calculating Fukui functions. This default gives broad coverage of the periodic table, and is a fast and robust method, which is important for automatic treatment of *N*+δ systems where SCF convergence can be challenging.



Use structures from: Workspace (included entries)

Files:

[✓] Calculate f+

[✓] Calculate f-

Fraction of electron: 0.01

Keywords: basis=lacvp*+ dftname=b3lyp iacc=1 dconv=5e-6

Job name: jaguar

Jaguar: Fukui, Host=localhost: 2, Incorporate=Append new entries individually

Figure 5.2. The Jaguar panel for Fukui function calculations.

The script generates five `.vis` files for each structure: three for the electron densities of the *N*, *N*+δ, and *N*-δ systems, and two for the f^- and f^+ Fukui functions, defined above. These volumes are imported into the Project Table when the job is incorporated.

There are two ways to visualize a Fukui function: either as an isosurface, or as a mapping onto an isosurface of some other property, such as the density. To view the surfaces, you can click the S button in the Title column; the Manage Surfaces panel opens, in which you can select and display the surfaces, and perform the surface mappings.

When you view a Fukui function by itself as an isosurface, the red and blue regions correspond to negative values and positive values, respectively. If you choose to map the Fukui function onto the electron density isosurface (which is readily available since it is calculated by the script), you can choose from various color schemes to represent the range of numerical values on the surface. Either way, negative values for f^- correspond to regions that lose electron density when the molecule undergoes electrophilic attack or when the molecule itself acts as a nucleophile, and positive values for f^+ correspond to regions which gain electron density when the molecule undergoes nucleophilic attack or when the molecule itself acts as an electrophile.

In addition to the surfaces, the minimum and maximum values of the Fukui functions on the density isosurface for each atom are added to the Maestro file as atom properties.

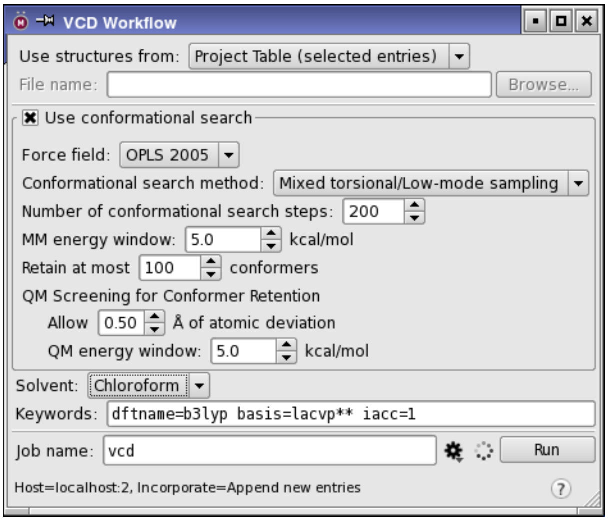
5.3 Vibrational Circular Dichroism (VCD) Spectra

VCD spectra can be used to determine the chirality of a molecule, by comparing the predicted spectrum with the experimental spectrum. The VCD spectrum of a molecule is the mirror image of the VCD spectrum of its enantiomer, so the assignment of a spectrum to a particular enantiomer is usually straightforward. Jaguar writes the VCD spectral data to a file called *jobname*_`vcd.spm`, which can be read into the Spectrum Plot panel in Maestro to generate a simulated spectrum.

Because different conformers have significantly different VCD spectra, it is important to compute a conformationally averaged spectrum when the molecule of interest has conformers with similar energies. You can use MacroModel to perform the conformational search, then refine the geometries with Jaguar, and eliminate redundant or high-lying conformers with MacroModel. This procedure is automated in the VCD Workflow (as encoded in the `vcd.py` script), which also calculates the Boltzmann weights used to average the spectrum.

To open the VCD Workflow panel, choose Applications → Jaguar → VCD Spectra or Tasks → Quantum Mechanics → VCD Spectra.

To open the Spectrum Plot panel, choose Applications → Jaguar → Plot Spectra or Tasks → Quantum Mechanics → Plot Spectra.



Use structures from: Project Table (selected entries)
File name:
[✓] Use conformational search
Force field: OPLS 2005
Conformational search method: Mixed torsional/Low-mode sampling
Number of conformational search steps: 200
MM energy window: 5.0 kcal/mol
Retain at most 100 conformers
QM Screening for Conformer Retention
Allow 0.50 Å of atomic deviation
QM energy window: 5.0 kcal/mol
Solvent: Chloroform
Keywords: `dftname=b3lyp basis=lacvp** iacc=1`
Job name: vcd
Host=localhost:2, Incorporate=Append new entries

Figure 5.3. The VCD Workflow panel.

To set up a calculation:

1. Specify the source of the structures with the Use structures from option menu.
2. If the structure source is a single molecule, select Use conformational search. If it is a set of conformers, deselect this option.
3. Choose a force field for the conformational search from the Force field option menu.

   The default is OPLS_2005.
4. Choose a conformational search method from the Conformational search method option menu.

   The methods are described in detail in Section 8.1 of the *MacroModel User Manual*.
5. Specify the number of steps performed in the search in the Number of conformational search steps box.

   The default is 200. When the number of generated trial structures matches the value in field, the conformational search is terminated.
6. Specify the threshold for comparison of trial conformers in the MM energy window box.

   New structures that are generated and minimized are kept only if their energy is less than this value above the current global minimum. Lowering this value results in fewer structures saved.
7. Specify the maximum number of conformers to keep for each input structure in the Retain at most N conformers box.
8. Specify cutoffs for eliminating redundant conformers after the set generated by MacroModel is minimized by Jaguar, in the QM screening for conformer retention boxes:
   - Allow *N* Å of atomic deviation—consider structures to be different if the maximum atom deviation for any pair of corresponding atoms exceeds the threshold given in the text box.
   - QM energy window—keep conformers only if their QM energy is less than this value above the current global minimum.
9. Choose the solvent to be used from the Solvent option menu. The default is None (a gas-phase calculation).
10. Specify any **gen** section keywords for the Jaguar calculations.

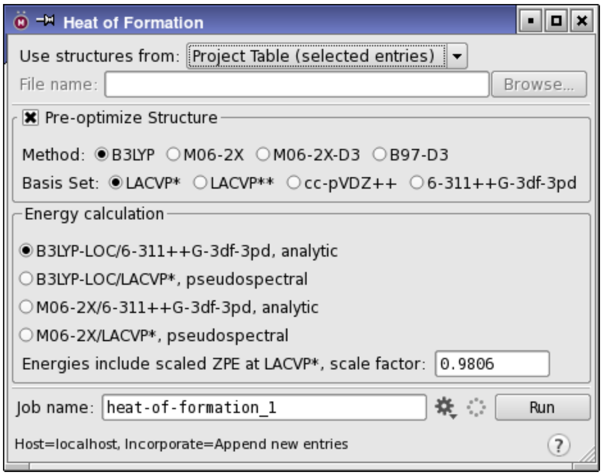
    The same keywords are used for all calculations. You might want to specify the basis set and the method in this section, for example. Some keywords are set automatically, such as the keywords for generating VCD spectra. The defaults are `dftname=b3lyp basis=lacvp** iacc=1`.

5.4 Heat of Formation

Jaguar calculates enthalpies and free energies as a standard part of a vibrational frequency calculation. Calculating the heats (enthalpies) of formation of molecules requires calculations on the standard states of the elements, which is challenging for elements whose standard state is solid or liquid.

The Heat of Formation workflow avoids this problem by calculating the atomization energies directly, then using experimental heats of formation of the gaseous atoms from the elements to arrive at the heat of formation of the gaseous molecule. The workflow is restricted to calculation of heats of formation of closed-shell organic molecules. You can run calculations on multiple molecules in the same job.

To open the Heat of Formation panel, choose Applications → Jaguar → Heat of Formation.


Heat of Formation

Use structures from: Project Table (selected entries)

File name:

[✓] Pre-optimize Structure

Method: [✓ B3LYP] M06-2X M06-2X-D3 B97-D3

Basis Set: [✓ LACVP*] LACVP** cc-pVDZ++ 6-311++G-3df-3pd

Energy calculation

[✓ B3LYP-LOC/6-311++G-3df-3pd, analytic]

B3LYP-LOC/LACVP*, pseudospectral

M06-2X/6-311++G-3df-3pd, analytic

M06-2X/LACVP*, pseudospectral

Energies include scaled ZPE at LACVP*, scale factor: 0.9806

Job name: heat-of-formation_1

Host=localhost, Incorporate=Append new entries

Figure 5.4. The Heat of Formation panel.

The first stage of the workflow is to pre-optimize the structures, to find the minimum. You can select a method and a basis set for the optimization. The basis sets available for the pre-optimization include the basis set used for the energy calculation.

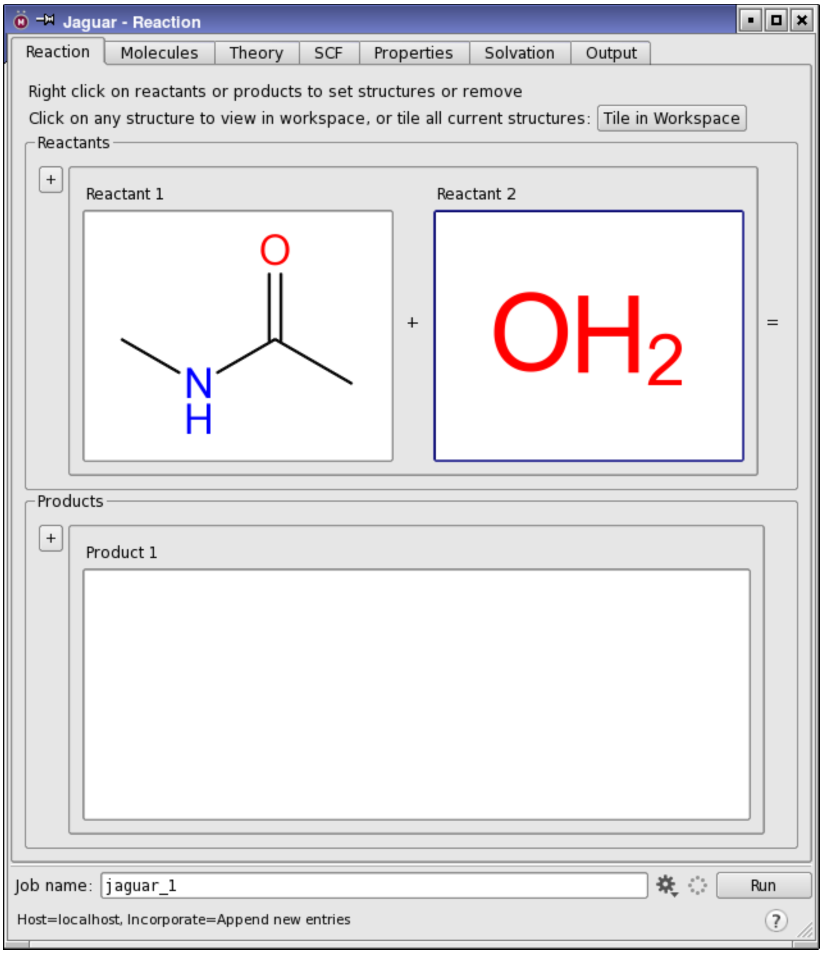
For the calculation of the energies, you can select one of two density functional methods, combined with one of two basis sets. The smaller basis set, LACVP*, is used with the pseudospectral method, so it is much faster than the larger basis set, but the results are not quite of the same quality.

You can also set the scale factor used when evaluating the zero-point energy from the frequencies, in the text box. The frequencies are calculated at the B3LYP/LACVP* level of theory with the specified scale factor.

Note: LACVP* is used with `numd=6` (Cartesian d functions), so that results for H–Ar are identical to those obtained with 6-31G*. To run with spherical d functions and this basis set, run `deltah.py` from the command line and set `-k=numd=5`.

5.5 Reaction Energies

Calculating a reaction enthalpy involves calculations on each of the individual molecules in the reaction, then taking the difference. Jaguar provides a convenient way of doing this calculation directly. In the Jaguar - Reaction panel, you can specify the reactant and product molecules, and Jaguar performs all the individual calculations and then calculates the reaction enthalpy.



Right click on reactants or products to set structures or remove

Click on any structure to view in workspace, or tile all current structures:

Job name: jaguar_1

Host=localhost, Incorporate=Append new entries

Figure 5.6. The Reaction tab.

To open the Reaction panel, choose Applications → Jaguar → Reaction. The Jaguar - Reaction panel opens with the Reaction tab displayed. This tab is used to choose the structures for the reaction.

The Reaction tab has two sections, one for reactants and one for products. Each section shows the structures, in separate boxes. Initially there is only one box in each section and they are empty. To add a reactant or a product structure to a box, click in the box and choose the source of the structure:

- Import from Project Table—Import the structure from the Project Table. A Choose Entry dialog box opens, so you can choose the entry.
- Import from File—Import the structure from a Maestro file. If the file has more than one structure, only the first is imported.

When the structure is imported, a 2D image is displayed in the box, and is labeled Reactant *N* or Product *N*. To display the structure in the Workspace, click in the box.

To add another reactant or product, click the + button to the left of the first box in either section. Another box is displayed, and you can add a structure to it. The boxes are numbered sequentially. To delete a reactant or product, right-click in its box and choose Delete from the shortcut menu. To clear the entire panel and start over, click the arrow next to the Settings button (gear icon at the lower right of the panel) and choose Reset Panel.

To display all the structures in the Workspace, click Tile in Workspace. Each structure is shown in a separate tile, labeled with the entry title and the structure label (which is stored as the Jaguar Reaction ID property in the project). See Section 4.4 of the *Maestro User Manual* for more information on tiles and their features.

When you have chosen all the structures, you should check that the stoichiometry is correct, otherwise the job will fail.

You can set the charge, spin multiplicity, symmetry use, and basis set for each structure in the Molecules tab. Choose the structure from the option menu at the top right, then make the settings. Make sure that you choose the same basis set for each molecule, and that the charge and spin multiplicity is consistent between reactants and products. The job will fail if any inconsistency is detected.

The Properties tab provides two options: one to calculate ΔG and ΔH, and one to pre-optimize the geometries of all the structures. If you choose to calculate ΔG and ΔH, a frequency calculation is performed for each structure.

The remaining tabs allow you to control the theoretical method, SCF parameters, and solvent, and choose properties to calculate for each molecule. These tabs are described in Chapter 3.

When you have made all the settings, you can run the job. The structures are incorporated as new entries in the project, with their properties. In addition, the ΔE, ΔG and ΔH values for the reaction are added as properties of the structures.

Output

The output from a Jaguar run always includes a Jaguar output file, which contains the primary output; a log file, which contains a job summary that is updated as the job is being run; and a Maestro-formatted file, which contains the geometry and properties in a form that can be read by Maestro and other Schrödinger software. If you request other output options in the Output tab of the Jaguar panel, various other files can also be generated as output.

You can view the output file from Maestro either from the Monitor panel, or the Project Table panel. In the Monitor panel, choose the file in the Details tab, and click on it to display it in the File tab. In the Project Table, there is an entry property that stores a link to the file. Click the link and choose View to open the file in a read-only text viewer (the View File panel), where you can browse the file, search for text, and copy text to the clipboard.

This chapter begins with a description of the `jaguar results` utility, which can be used to obtain summaries of Jaguar results. The chapter continues with a description of the Jaguar output file for a standard Hartree-Fock calculation, followed by a discussion of the changes in the output for various other calculation options and the output options that can be set in the Output tab. The final section explains the log file, which is the file displayed in the Monitor panel as a job runs.

In this chapter, footnotes indicate the Jaguar input file keywords and sections that correspond to particular Maestro settings. If you are working from Maestro, you can ignore these footnotes, but you may find them helpful if you want to use input files to run jobs outside Maestro.

6.1 Summarizing Jaguar Results

You can obtain summaries of Jaguar results in simple table form with the following command

`jaguar results` [*option-list*] [*output-file-list*]

Jaguar searches the output files you specify for the information you request through the command options. The order of the options determines the order in which the corresponding data is printed. The options are listed in Table 6.1, grouped into classes. You can also obtain a list of supported options by entering the command

```
jaguar results -help
```

The tables produced by `jaguar results` can describe results from one job or several jobs. The results can be restricted to final results from each job listed (the default), or can include intermediate results (SCF energies for each geometry in an optimization, for instance). By default, each line lists information that pertains to the entire input structure, but you can also request some kinds of information for each individual atom in the structure. Each of these types of results tables are described below. Data values for each output file are printed with results for each job on a separate line.

Table 6.1. Options for the `jaguar results` *command*

Option	Meaning
Job options	

`-jobname`
`-longjobname`
`-hugejobname`
	job name, with different output lengths

`-filename`	output file name (`.out`)
`-errno`	error number reported when an error was detected by the program
`-lastexe`	name of the last executable run
`-glibc`	version of glibc printed in the output file, indicates a crash
Molecular information options:	
`-stoich`	stoichiometry
`-weight`	molecular weight
`-basis`	basis set
`-nbasis`	number of basis functions
`-ncanorb`	number of canonical orbitals
`-nelectron`	number of electrons
`-npair`	number of electron pairs
`-nsigma`	number of sigma electron pairs
`-npi`	number of pi electron pairs
`-natom`	number of atoms
`-symmetry`	molecular symmetry
`-nsymm`	symmetry number
`-charge`	molecular charge
`-multip`	spin multiplicity

Table 6.1. Options for the `jaguar results` *command (Continued)*

Option	Meaning
`-s2`	spin: <S**2>
`-sz2`	spin: Sz*<Sz+1>
`-method`	SCF/post-SCF method
Energy options:	
`-energy`	final molecular energy
`-enuc`	nuclear repulsion energy
`-egas`	gas-phase energy of molecule (solute)
`-esoln`	solution-phase energy (esolv+egas)
`-esolv`	solvation free energy
`-esolute`	solution phase solute energy
`-esolvent`	solvent energy
`-ereorg`	solute reorganization energy (esolute-egas)
`-aposteri`	a posteriori energy correction from density functional calculations
`-aposteri0`	energy without a posteriori correction from density functional calculations
`-homo`	HOMO energies
`-lumo`	LUMO energies
`-gap`	HOMO-LUMO energy gap
`-zpe`	zero-point energy
`-entropy`	entropy (S) at 298.15 K
`-enthalpy`	enthalpy (H) at 298.15 K
`-gibbs`	Gibbs free energy (G) at 298.15 K
`-cv`	heat capacity (C_v) at 298.15 K
`-int_energy`	internal energy (U) at 298.15 K
`-Utot`	Total internal energy (U_{tot}) at 298.15 K, including SCF and zero-point energies
`-Htot`	Total enthalpy (H_{tot}) at 298.15 K, including SCF and zero-point energies
`-Gtot`	Total Gibbs free energy (G_{tot}) at 298.15 K, including SCF and zero-point energies
Molecular property options:	
`-dipole`	Total dipole moment as expectation of the dipole operator

Table 6.1. Options for the `jaguar results` *command (Continued)*

Option	Meaning
`-espdipole`	Total dipole moment from ESP charge-fitting with multipole constraints
`-polardipole`	Total dipole moment from a polarizability calculation
`-alpha`	Polarizability
`-beta`	First hyperpolarizability
`-gamma`	Second hyperpolarizability
`-pka`	pK_a
`-pkb`	pK_b
`-msa`	Molecular surface area from PBF solvation calculation
`-sasa`	Solvent-accessible surface area from PBF solvation calculation
Vibrational frequency options:	
`-freqs`	Vibrational frequencies
`-ir`	Infrared intensities
`-raman`	Raman intensities
`-vibsymm`	Vibrational symmetries
`-redmass`	Reduced masses
`-fc`	Force constants
`-d_str`	Dipole transition moments
`-r_str`	Optical rotation transition moments
Geometry optimization and scan options:	
`-iterg`	geopt iteration number
`-stepg`	geopt step number
`-zvar` *name*	z-variable value (must be followed by z-variable name)
`-zvar_all`	values of all z-variables
`-grms`	rms gradient
`-gmax`	maximum gradient component
`-drms`	rms displacement
`-dmax`	maximum displacement
`-echange`	energy change
`-irc`	return values from IRC points only for whatever options are given

Table 6.1. Options for the `jaguar results` *command (Continued)*

Option	Meaning
Timing options:	
`-time`	total cpu time for job
`-tscf`	total time in `scf` (cumulative)
`-trwr`	total time in `rwr` (cumulative)
`-tder1b`	total time in `der1b` (cumulative)
`-walltime`	wall clock time for job (total elapsed time)
SCF information options:	
`-iter`	number of SCF iterations
Per-atom information options:	
`-atoms`	atom names
`-atomnums`	atomic numbers
`-coords`	cartesian atomic coordinates
`-forces`	cartesian atomic forces
`-charges`	ESP atom-centered charges
`-mulliken`	Atomic Mulliken charges
`-stockholder`	Atomic stockholder charges
`-nmr`	NMR isotropic shielding
`-nmr_shifts`	NMR chemical shifts relative to TMS for C and H
`-fukui`	All Fukui indices
`-f_NN_H`	Fukui index f_NN HOMO
`-f_NS_H`	Fukui index f_NS HOMO
`-f_SN_H`	Fukui index f_SN HOMO
`-f_SS_H`	Fukui index f_SS HOMO
`-f_NN_L`	Fukui index f_NN LUMO
`-f_NS_L`	Fukui index f_NS LUMO
`-f_SN_L`	Fukui index f_SN LUMO
`-f_SS_L`	Fukui index f_SS LUMO
Options for printing and intermediate results:	
`-csv`	print in CSV format by adding a comma after each value

Table 6.1. Options for the `jaguar results` *command (Continued)*

Option	Meaning
`-title`	print column titles
`-titleonly`	print only the column titles
`-all`	report results every geometry iteration
`-allscf`	report results for each `scf` calculation
`-allder1b`	report results for each `der1b`

Note: The vibrational frequency options are incompatible with the molecular property and per-atom properties options. Property values that do not exist in the output file are listed as N/A.

6.1.1 Reporting Final Results From One or More Jobs

By default, each row of the Jaguar results table (except the title row) shows final results from a Jaguar output file that was given in the `jaguar results` command. For instance, if you entered the command

```
jaguar results -energy RuCp2.out piperidine.out
```

from a directory containing the output files `RuCp2.out` and `piperidine.out`, you would get a very simple table like this:

```
-480.726524
-250.470399
```

where the first line lists the final energy from the job `RuCp2` and the second lists the final energy from the job `piperidine`.

If you use the option `-title`, column headings are added. The columns appear in the table in the order they are listed in the `jaguar results` command. For instance, the command

```
jaguar results -title -jobname -method -energy h2o.out \
  h2o_b3lyp.out
```

(where `h2o.out` and `h2o_b3lyp.out` are output files from jobs at the Hartree-Fock and B3LYP density functional theory levels, respectively) gives the table

```
    Jobname          Method        Energy
                                  [hartree]
================  ===========  ============
h2o                   HF         -76.023641
h2o_b3lyp            B3LYP       -76.418721
```

with the job name, method, and energy listed from left to right in the same order they were in the `jaguar results` command. If you want to see ahead of time what the column headings of your table would look like without any results listed, use the `-titleonly` option.

The Jaguar results tables can list both information describing the job run (for instance, its name, the basis set and SCF method used, or the stoichiometry of the molecule) and information about the results of the job (for example, the final energy or dipole moment). Each of these types of information appears in a column in the table.

6.1.2 Reporting Intermediate Results

By default, only the final results are reported for each job; therefore, for instance, a table of results from three jobs would have three rows of information. However, you can also request that information from each geometry, SCF, or gradient calculation be reported in a different row of the results table. For instance, the command

```
jaguar results -title -all -iterg -echange -gmax -grms \   -dmax -
drms dftg.out
```

here produces a table showing the convergence of a BLYP geometry optimization of water:

```
Geopt   Energy      Gradient     Gradient     Displace.    Displace.
iter   [change]       [max]        [rms]        [max]        [rms]
==== =========== =========== =========== =========== ===========
  1               3.22E-02 .   2.65E-02 .   5.53E-02 .   4.88E-02 .
  2   -2.04E-03 .  3.85E-03 .   3.18E-03 .   2.79E-02 .   1.70E-02 .
  3   -7.04E-05 .  4.19E-04 *   3.82E-04 .   1.45E-03 *   1.01E-03 *
  4   -1.04E-06 #  3.05E-05 #   2.52E-05 #   6.13E-05 #   5.13E-05 #
```

The last section of Table 6.1 lists options for specifying when to report intermediate and final results from jobs. The `-all` option, which lets you track the progress of a geometry or transition-state optimization, is likely to be the most useful of the options. The `-allscf` option can be used for intermediate results in complex non-optimizations, such as solvation jobs.

6.1.3 Reporting Results for Each Atom

By default, each line of output from a `jaguar results` command lists information that pertains to the entire input structure, but you can also request some kinds of information for each individual atom in the structure. The options that let you print tables of coordinates, forces, or charges for individual atoms are listed in the per-atom information options section of Table 6.1. You should not use the atom-related options with any of the options that request information pertaining to the entire molecule (the `-energy` option, for instance).

6.2 Output From a Standard HF Calculation

The contents of a Jaguar output file vary according to the calculation and output selections made. This section describes the output file for a standard, default, single point, closed shell Hartree-Fock calculation. Section 6.3 on page 136 describes the variations in the output file for the calculation options described in Chapter 3.

All output files begin with a line listing the job name, the machine upon which the job ran, and the time the job was started, followed by the general copyright information for the version of Jaguar that was used for the run. The rest of this section describes output from each individual Jaguar program run for a default calculation.

The output from the program `pre` begins with a description of the calculation to be performed: the job name, the directory containing the executables used to run the job, the directory for temporary files, comments from the input file (if any), and the names and paths of any non-default data files used for the calculation (see Section 9.1 on page 195 and Chapter 10).

Next, the basis set used for the calculation, the molecule’s net charge and multiplicity, and the number of basis functions used for the calculation are specified. This information is followed by the molecular geometry input, which gives the atom label and coordinates for each atom. (If the atom labels provided in the geometry are not unique—for instance, if two hydrogens are each called “h”—this information is preceded by a list of original atom labels and new atom labels assigned by the program.)

The molecule’s symmetry is analyzed, a process which may involve translating and rotating the molecule. These procedures are noted in the output file, along with the point group used for the calculation, the nuclear repulsion energy, and the symmetrized geometry, which is used for the rest of the calculation.

One-electron integrals are calculated by the `onee` program, which prints the smallest eigenvalue of the overlap matrix **S** and the number of canonical orbitals used for the calculation. Canonical orbital eigenvectors with very small eigenvalues (less than 5.0×10^{-4}) are removed and thus are not counted. The eigenvalue cutoff can be controlled by setting the keyword **cut20** to the desired value in the **gen** section of the input file. The number of canonical orbitals can also be controlled by setting the keyword **ncanorb** in the **gen** section of the input file.

The program `hfig` constructs a starting wave function (initial guess) for a Hartree-Fock calculation. The output from the program `hfig` for a default calculation begins with the line, “initial wave function generated automatically from atomic wave functions.” Next, a table lists the number of orbitals, and of occupied orbitals in each shell, having each irreducible representation for the appropriate point group. Finally, the orbital occupation for each shell is listed; an occupation of “1.000” indicates a closed shell. An example, for a calculation of water using a 6-31G** basis set, follows:

```
  start of program hfig
  initial wave function generated automatically from atomic wave functions

  Irreducible     Total no   No of occupied orbitals
  representation  orbitals   Shell_1  Shell_2    ...
  A1                12         3
  A2                 2         0
  B1                 4         1
  B2                 7         1
  ------------------------
  Orbital occupation/shell    1.000

  end of program hfig
```

If the molecule contains a transition metal atom, there may be several ways of occupying the d orbitals. In this case, `hfig` prints a list of the possible states, and continues with the first of these. It is possible, however, that a different initial occupation of the metal d orbitals would lead to a lower energy wave function. To see whether this is the case, you should run an SCF calculation for each of the possible degenerate states, by selecting the state with the **istate** keyword. An example for the FeH_4 molecule follows.

```
Low energy states below 0.005000 hartree:

    State  Rel. Energy    MOs:    9  10  11  12  13
                                metal d occupations
       1    0.00000000          2   1   1   0   0
       2    0.00000224          1   2   1   0   0
       3    0.00062053          2   1   0   1   0
       4    0.00062276          1   2   0   1   0
       5    0.00071513          2   1   0   0   1
       6    0.00071737          1   2   0   0   1

    WARNING: The lowest energy configurations are
    degenerate. The MO numbers with occupied metal
    d orbitals are given in the table above.  Jaguar
    will use the first configuration, but you can
    select a different state configuration number
    from the table above with the istate keyword.

  Using state configuration  1:   2   1   1   0   0
```

In this example, Jaguar has found six possible occupations of the five metal d orbitals that have essentially the same energy. The table shows which MO numbers correspond to the metal d orbitals (9-13 in this example), the occupation numbers (0, 1 or 2 electrons per orbital), and the relative energy, in hartrees. The lowest energy is the reference energy and is always 0.0.

The probe program, which follows hfig and ensures orthogonalization, has no significant output.

The output for the grid generation done by the program grid lists the number of grid points for each atom, as well as the total number of grid points, for each grid used in the application of the pseudospectral method. If you would like more information about these grids, see Section 10.4 on page 305. The rwr program, which generates the **Q** operators needed for the pseudospectral method, runs next, but has no significant output.

An example of the output from the next program, scf, again for a water molecule, is given here and is explained below.

```
  start of program scf
 number of electrons..........         10
 number of alpha electrons....          5
 number of beta electrons.....          5
 number of orbitals, total....         25
 number of core orbitals......          5
 number of open shell orbs....          0
 number of occupied orbitals..          5
 number of virtual orbitals...         20
 number of hamiltonians.......          1
 number of shells.............          1
 SCF type: HF

       i  u  d  i  g
       t  p  i  c  r                                RMS    maximum
       e  d  i  u  i                      energy  density   DIIS
       r  t  s  t  d     total energy    change  change    error

etot   1  N  N  5  M   -75.61350567257           1.6E-02  3.3E-01
etot   2  Y  Y  6  M   -75.99456008691  3.8E-01  6.2E-03  6.9E-02
etot   3  Y  Y  6  M   -76.01904109359  2.4E-02  1.7E-03  2.9E-02
etot   4  N  Y  2  U   -76.02333233097  4.3E-03  7.6E-04  4.7E-03
etot   5  Y  Y  6  M   -76.02361760760  2.9E-04  1.7E-04  1.5E-03
etot   6  Y  N  6  M   -76.02364072535  2.3E-05  0.0E+00  0.0E-00


 Energy components, in hartrees:
   (A)  Nuclear repulsion............      9.33000672144
   (E)  Total one-electron terms.....  -123.34165776264
   (I)  Total two-electron terms.....    37.98801031585
   (L)  Electronic energy............   -85.35364744679  (E+I)
   (N)  Total energy.................   -76.02364072535  (A+L)

 SCFE: SCF energy: HF     -76.02364072535 hartrees   iterations:   6
```

```
HOMO energy:    -0.49745
LUMO energy:     0.21516

 Orbital energies/symmetry label:
 -20.55693 A1       -1.34635 A1       -0.71380 B2       -0.56828 A1
  -0.49745 B1        0.21516 A1        0.30862 B2        1.01720 B2
   1.09266 A1        1.13459 A1        1.16904 B1        1.29575 B2
   1.41126 A1        1.80256 A2        1.82999 A1

 end of program scf
```

The output from the program `scf` begins with a list of information detailing various numbers of electrons, orbitals, Hamiltonians used for the calculation, shells, and the calculation type.

Next, the energy output from the SCF iterations is shown in table form. Some of the text for the column headings should be read down rather than across. The number of the iteration is provided first in each row, followed by a "Y" or "N" indicating whether the Fock matrix was updated or not. When the Fock matrix is updated, the changes are made using a difference density matrix whose elements reflect the changes in the density matrix elements from the previous iteration to the current one.

The next entry indicates whether the DIIS convergence scheme was used for that iteration. As above, "Y" or "N" indicate yes or no. The DIIS method produces a new estimate of the Fock matrix as a linear combination of previous Fock matrices, including the one calculated during that iteration. DIIS, which is enabled by default, usually starts on the second iteration, and is not used on the final iteration. If the entry in this column reads "A," it indicates that DIIS was not used for that iteration, but the density matrix was averaged.

The cutoff set for each iteration is indicated under the "icut" heading. Cutoff sets are explained in the cutoff file description in Section 10.5 on page 308.

The grid column lists the grid used for that iteration, which must be one of the grid types coarse (signified by a C), medium (M), fine (F), or ultrafine (U). See Section 9.5.27 on page 263 and Section 10.4 on page 305 if you want more information on grids and grid types.

The total energy for the molecule in Hartrees appears in the next column, followed by the energy change from the previous iteration to the current one.

The RMS density change column provides the root mean square of the change in density matrix elements from the previous iteration to the current one.

In the last column, the maximum DIIS errors listed provide a measure of convergence by listing the maximum element of the DIIS error vector. For HF and DFT closed shell calculations, the DIIS error vector is given by **FDS** – **SDF** in atomic orbital space, where **F**, **D**, and **S** are the Fock, density, and overlap matrices, respectively. For open shell cases, the definition of the error vector is given in reference 11.

After the energy information for each SCF iteration, a summary of the components of the final, converged energy is given. Each of these energies is labeled with a letter (for example, "A" for the nuclear repulsion), and information to the right of some of the energies describes the relations between the components in terms of these letters. A line below the table summarizes the calculation type and energy, as well as the number of SCF iterations.

If the input system's spin multiplicity is not singlet, the summary of the SCF output also includes a breakdown of the two-electron contribution to the energy into Coulomb and exchange parts. For each of these parts, the contribution from each Hamiltonian is listed.

The highest occupied molecular orbital (HOMO) and lowest unoccupied molecular orbital (LUMO) energies are listed next. Finally, the energies for each occupied orbital and for the ten lowest-energy virtual orbitals are provided, with each orbital identified by a symmetry label. Virtual orbitals and eigenvalues are determined in the same manner as in ref. 175. The virtual orbitals are obtained by diagonalizing $H_0 + \sum f(2J - K)$, where f is the occupation of each orbital (1 for a closed shell). For closed shell Hartree-Fock calculations, this definition yields the standard orbitals and eigenvalues.

Finally, the CPU time for the job, the machine upon which the job ran, and its time of completion are noted at the end of the output file.

6.3 Output File Content for Various Calculation Types

Any time you make a non-default setting for a calculation, the output from the program `pre` notes the non-default options chosen. This output appears above the molecular geometry output from the `pre` program. This section describes the changes in output for various calculation settings described in Chapter 3.

Generally, only the *format* changes that result from these settings are discussed below. Naturally, these settings will often change the data listed. Options that have no significant impact on the output *format* are not discussed in this section.

6.3.1 DFT

If you use density functional theory for the SCF calculation, the output above the SCF table lists the functional or combination of functionals used. The energy information for DFT calculations includes the breakdown of the two-electron energy into Coulomb and exchange-correlation terms. For DFT calculations, virtual orbitals are obtained by diagonalizing $H_0 + \sum f(2J + V_{xc})$, where f is the occupation of each orbital (1 for a closed shell). For closed shell calculations, this definition yields the standard orbitals and eigenvalues.

The scf output from post-SCF DFT energy evaluations first lists the standard output for the HF or DFT SCF calculation, then lists the energy breakdown and total energy from the post-SCF DFT analysis. Since the post-SCF DFT treatment does not change the wave function, no orbital output is reported from this step.

The output from the program pre for non-default options contains the detailed description of customized functional combinations for SCF or post-SCF DFT calculations.

6.3.1.1 B3LYP-LOC

In the Jaguar output file the computation of the LOC corrections is marked with the message:

```
 Computing the LOC energy correction...
```

which follows the solution of the SCF equations. The value of the correction itself is reported in hartree in the (N0) line of the energy breakdown, for example:

```
(N0) L0 correction...............       0.000269525813
```

The LOC correction is automatically included in the total energy.

For ionized systems in which an electron is removed from an orbital to which at least two atoms contribute, a special section, **loc_ib**, must be provided in the input file, for best results. This section specifies two such atoms forming a bond in an un-ionized state. Essentially, it is convenient to think that such the electron is removed from a bond. For example, the specification

```
&loc_ib
4 8
&
```

indicates that an electron is removed from a bond connecting atoms 4 and 8.

6.3.2 ZORA Relativistic Calculations

If you perform a calculation with the ZORA relativistic Hamiltonian, the total energy reported is the energy calculated with this Hamiltonian. After the total energy, the scaled ZORA correction and the scaled ZORA total energy are reported, as follows:

```
   (N)  Total energy.................  -24547.73960260797  (A+L)
   (Zc) Scaled ZORA Correction.......    1026.51936482586
   (ZE) Scaled ZORA Energy...........  -23521.22023778210  (N+Zc)
```

See Section 8.4 on page 190 for a description of these quantities.

6.3.3 LMP2

If you perform a local MP2 calculation, the output from the programs `pre` and `hfig` is somewhat different from that of a Hartree-Fock calculation, since the use of symmetry is turned off automatically for LMP2 calculations. The output from the program `scf` includes the Coulomb and exchange contributions to the two-electron terms for these calculations, and the symmetry labels are not included in the output of orbital energies.

The program `loclmp2`, which computes localized orbitals, runs after `scf` in an LMP2 calculation, and its output notes the number of orbitals that are localized. Below that output, the output from the program `lmp2` appears.

For local MP2 calculations, the output begins by listing the localized orbitals involved in the local MP2 treatment—namely, the localized orbitals centered on one or both atoms in the pairs of atoms for which an LMP2-level treatment was requested.

All LMP2 output includes a description of the type of orbitals used in the MP2 calculation. First, it lists the total number of orbitals. Next, it lists the number of frozen core and valence MP2 orbitals. The numbers of core and valence orbitals will be affected by whether you use valence electrons only or all electrons for the atoms in the calculation. Next, the numbers of occupied and virtual orbitals for the molecule are listed. The list ends with the number of exchange Hamiltonians.

Some information on the convergence of the LMP2 energy correction appears below the list of orbital information, followed by the Hartree-Fock energy and the LMP2 energy correction, which gives the improvement to the energy over the HF value. The total LMP2 energy (the HF energy plus the correction) is given immediately afterwards. (If your job is a local MP2 calculation and you want to see the energy from each LMP2 pair, use the **gen** section keyword setting **ip170**=2, as described in Section 9.5.23 on page 257.)

6.3.4 Geometry or Transition-State Optimization

The output format description for optimizations in this subsection applies to calculations of either minimum-energy structures or transition states. Although the Hessians used during these calculations are different, the Jaguar programs run are the same, and the output format is very similar. (Exceptions are described below.)

If you calculate an optimized molecular structure, a transition state, or forces, any SCF calculations during the run use the RMS density change convergence criterion described in Section 3.7 on page 53 instead of the usual energy convergence criterion. Therefore, these SCF calculations often proceed for several more iterations than single point energy calculations.

If you select forces only for the Optimize geometry setting, the programs der1a, rwr, and der1b will run after scf does. The forces felt by each atom in the unoptimized geometry will be output from der1b, in a table listing each atom and the components of the force upon it in the x, y, and z directions. The x, y, and z components of the total force on the molecule are listed in the last line, and provide a judge of how accurate the force calculations are in most cases, since they should generally be zero. An example of this force table for a water molecule optimization follows:

```
  forces (hartrees/bohr) : total

 atom  label          x              y              z
 ----  -----   -------------  -------------  -------------
   1    O       0.000000E+00   0.000000E+00  -2.620407E-05
   2    H1      0.000000E+00  -6.462331E-05   1.291533E-04
   3    H2      0.000000E+00   6.462331E-05   1.291533E-04
 -----------   -------------  -------------  -------------
    total       0.000000E+00   0.000000E+00   2.321025E-04
```

When force calculations or optimizations of a system's minimum energy structure or transition state are performed at the LMP2 level, the program der1b never runs. Instead, forces are calculated by the programs lmp2der, lmp2gda, and lmp2gdb. The last of these programs provides a table of output listing the forces on each atom in the same format as the sample table above.

For geometry optimizations, Jaguar prints bond length and angle information in the output from the program pre. If you have constrained bond lengths or angles of the geometry so that they are frozen during the optimization, as described in Section 4.2 on page 96, the constraints are also listed in the pre output.

At the end of the first SCF calculation, the programs der1a, rwr, and der1b run, calculating the forces felt by each atom in the unoptimized geometry and writing them to the output file, as described above.

These force results are followed by the output from the program geopt, which includes a number indicating how many times it has been called, in the "start of program geopt" line. Every time geopt is called, this number is updated. However, since geopt can be called for Hessian refinement steps as well as for generating new geometries during an optimization, and since geometry optimizations occasionally revert back to a previous geometry and "restart" the calculation from there, the next line of the geopt output reports what sort of step is being performed and numbers that step accordingly.

If the program detects that the input lists separate fragments, each of which contain only atoms unbonded to the atoms in any other fragment, as for a van der Waals complex, then the number of fragments is listed near the start of the geopt output.

For transition-state optimizations, the eigenvalues of the nuclear Hessian are reported the first time `geopt` runs. If the initial Hessian is being refined, the coordinates for the refinement and their eigenvalues are listed. (If a coordinate you have specified is inappropriate because of symmetry restrictions or other constraints, the output will indicate the problem.) The `geopt` output then lists information on the current (original) geometry's gradient elements, describes the small step it will use to alter the first coordinate used in the Hessian refinement, describes the internal coordinates and optimization variables as stretches, bends, or torsions, and indicates how it generates a new geometry by altering the relevant coordinate by the amount described by the step size.

The new geometry generated for Hessian refinement is used to obtain energy and gradient information, which requires the programs `onee`, `grid`, and `rwr` to run and generate output in the usual formats. This is followed by output from the program `scf`, which now starts with the calculation type and the table showing the energy output from each SCF iteration (skipping the listed information about electrons, orbitals, and so on). The output continues with output in the usual formats from `der1a`, `rwr`, and `der1b`. The information obtained on that geometry is then used in `geopt`, which runs a second time, reporting similar information about the planned changes to the molecular structure for the next Hessian refinement step (if there is one) and reporting the change in total energy from the original geometry to the geometry for the first Hessian refinement step. This process of altering single coordinates from the original geometry and calculating energies and gradients for the changed geometry continues until all requested Hessian refinement steps have been performed, which the output indicates with a line beginning "Hessian optimization completed." At that point, `geopt` performs a geometry optimization step from the original geometry, and the optimization continues until convergence.

For transition-state optimizations, the output for iterations that follow any Hessian refinement includes information identifying the transition vector used for that iteration. This output includes the transition vector's eigenvalue and the stretches, bends, or torsions that are its most important components.

For any optimization iteration using level shifting, after any relevant lines of `geopt` output described above, some information on the computed level shift (which may then be adjusted to satisfy step-size constraints) is included in the output. For optimization steps past the first geometry change, the change in total energy from the previous geometry to the newly calculated geometry (in Hartrees) is listed next.

The `geopt` output then lists the maximum element of the analytic gradient calculated by the earlier programs; the root mean square of the gradient elements; the step size predicted for the geometry change, the trust radius for that iteration and, if it is smaller than the step size, the factor used to scale the step size so it is no larger than the trust radius; the maximum element of nuclear displacement; and the root mean square element of the nuclear displacement. The predicted energy change for the new structure generated by `geopt` is also listed.

The values for the energy change, gradient, and nuclear displacement described in the previous paragraph are important because they are each tested against the convergence criteria determined by the Convergence criteria setting in the Optimization tab, as described in Section 4.1 on page 91, or, alternatively, the criteria set by the **gconv** keywords in the input file. The criteria are described in detail in Section 9.5.10 on page 217. If the gradients are converged and the energy change is below 2.5 x 10^{-7}, the optimization stops (unless it is on the first geometry optimization iteration). Similarly, if the gradients are converged and one of the gradient criteria is 5 times lower than the convergence level, then the optimization stops if the energy change is less than 2.5 x 10^{-6}.

The symbol following each quantity used to judge convergence indicates how well converged it is. The symbol "." indicates convergence criteria that are not satisfied, "*" indicates criteria that are satisfied, "#" indicates criteria that are quite well satisfied, "!" indicates values that are essentially zero. If the convergence criteria mentioned are not met, and if the maximum number of iterations has not been exceeded, the output notes "molecular structure not yet converged..." and the optimization continues.

The output next lists the movement of the center of mass. If the output option for the bond length and angles is enabled, the output then lists this information for the new structure. Finally, the nuclear repulsion energy for the new geometry is listed.

If the molecular structure was not yet converged and the maximum number of geometry optimization iterations allowed was not reached in the previous iteration, the output from more geometry optimization iterations follow. The output from each iteration begins with `onee`, `grid`, and `rwr` output in the usual formats, and continues with output from `scf`, which now starts with the calculation type and the energy output from each SCF iteration (skipping the listed information about electrons, orbitals, and so on). The output further continues with output in the usual formats from `der1a`, `rwr`, and `der1b`, winding up with the output from `geopt`. The last such geometry optimization iteration contains, in the `geopt` output, either the line, "Geometry optimization complete," or the line, "stopping optimization: maximum number of iterations reached," depending on whether the convergence criteria were met before the maximum number of iterations was reached.

6.3.5 Solvation

6.3.5.1 Output from a PBF Calculation

Performing a solvation calculation involves several iterations in which the wave functions for the molecule in the gas phase are calculated. The program `ch` performs electrostatic potential fitting, which represents the wave function as a set of point charges on the atomic centers. The interactions between the molecule and the solvent are evaluated by Jaguar's Poisson-Boltzmann solver, which fits the field produced by the solvent dielectric continuum to another set of point charges. These charges are passed back to `scf`, which performs a new calculation of the wave function for the molecule in the field produced by the solvent point charges. Electrostatic potential fitting is performed on the new wave function, the solvent-molecule interactions are reevaluated by the Poisson-Boltzmann solver, and so on, until the solvation free energy for the molecule converges.

For solvation calculations on neutral systems in water whose atoms all have atomic numbers under 19 (H-Ar), by default, the program `pre` evaluates the Lewis dot structure for the molecule or system and assigns atomic van der Waals radii accordingly. (For more information on this process, see Section 10.6 on page 311.) These van der Waals radii are used to form the boundary between the solvent dielectric continuum and the solute molecule. The Lewis dot structure and van der Waals radii information both appear in the output from the program `pre`. The radii are listed under the heading "vdw2" in the table of atomic information below the listing of non-default options. See Section 9.8 on page 273, which describes the **atomic** section of the input file, if you want information on the other information in this table.

After the `pre` output, the usual output appears for the first, gas-phase calculation, except that the energy breakdown for the `scf` output also describes the electron-nuclear and kinetic contributions to the total one-electron terms in the energy, as well as the virial ratio –V/T, where V is the potential energy and T is the kinetic energy. This ratio should be –2 if the calculation satisfies the virial theorem.

After the first `scf` output, the output from the first run of the program `ch` appears. Since performing a solvation calculation enables electrostatic potential fitting to atomic centers, the usual output for that option, which is described in Section 6.3.7 on page 149, is included every time output from the program `ch` appears in the output file. The `post` program writes out the necessary input files for the Poisson-Boltzmann solver; this step is noted in the output file.

The next output section comes from the Poisson-Boltzmann solver. The output includes information on the area (in Å^2) of the molecular surface formed from the intersection of spheres with the van der Waals radii centered on the various atoms; the reaction field energy in kT (where T = 298 K), which is the energy of the interaction of the atom-centered charges with the solvent; the solvent-accessible surface area (in Å^2), which reflects the surface formed from the points whose closest distance from the molecular surface is equal to the probe radius of the solvent; and the cavity energy in kT, which is computed to be the solvation energy of a nonpolar solute whose size and shape are the same as those of the actual solute molecule, as described in reference [15].

The output from the program `solv` follows the Poisson-Boltzmann solver results, giving the number of point charges provided by the solver to model the solvent, the sum of the surface charges, the nuclear repulsion energy already calculated by Jaguar, the nuclear-point charge energy representing the energy of interaction between the molecule's nuclei and the solvent point charges, and the point-charge repulsion energy, which is calculated but not used by the rest of Jaguar because it is irrelevant to the desired solvation results.

Next, the output for the second solvation iteration begins. The output from `scf` comes first, giving the results for the molecule-and-solvent-point-charges system. The following is the output of the scf program for a HF/6-31g** calculation on water in cyclohexane:

```
  start of program scf

       i  u  d  i  g
       t  p  i  c  r                                 RMS    maximum
       e  d  i  u  i                      energy  density    DIIS
       r  t  s  t  d      total energy    change  change     error

etot   1  N  N  2  U     -76.03575354976          7.2E-04  6.7E-03
etot   2  Y  Y  6  M     -76.03606249325  3.1E-04  2.0E-04  1.8E-03
etot   3  Y  Y  6  M     -76.03609084305  2.8E-05  9.9E-05  8.5E-04
etot   4  Y  N  6  M     -76.03609374058  2.9E-06  0.0E+00  0.0E+00



 Energy components, in hartrees:
   (A)  Total zero-electron terms....       9.28561418426
   (B)    Nuclear-nuclear...........       9.26351790628
   (C)    Nuclear-solvent...........       0.02209627798
   (E)  Total one-electron terms.....    -123.28959841503
   (F)    Electron-nuclear..........    -199.09118131853
   (G)    Electron-solvent..........      -0.03501895602
   (H)    Kinetic...................      75.83660185952
   (I)  Total two-electron terms.....      37.96789049019
   (L)  Electronic energy...........     -85.32170792484  (E+I)
   (N)  Total quantum mech. energy...     -76.03609374058  (A+L)
   (O)  Gas phase energy............     -76.02348975693
   (P)  Solution phase energy........     -76.02963240156  (Q+R+S)
   (Q)    Total solute energy........     -76.02317106254  (N-C-G)
   (R)    Total solvent energy.......      -0.00646133902  (C/2+G/2)
   (S)    Solute cavity energy.......       0.00000000000
   (U)  Reorganization energy........       0.00031869439  (Q-O)
   (V)  Solvation energy............      -0.00614264463  (P-O)

 SCFE:  Solution phase energy: HF        -76.02963240156 hartrees
        SCF iterations:   4


 HOMO energy:    -0.49872
 LUMO energy:     0.22339

  Orbital energies/symmetry label:
  -20.55740 A1        -1.34254 A1        -0.70660 B2        -0.57166 A1
  -0.49872 B1         0.22339 A1         0.31683 B2         1.01176 B2
   1.09142 A1         1.12989 A1         1.16588 B1         1.29364 B2
   1.41759 A1         1.80536 A2         1.82107 A1

 end of program scf
```

As for any later solvation iterations, the scf output begins with the calculation type and the table of energy results for each iteration, skipping the list of information about the molecule's electrons and orbitals. The energy information below the table includes several additional terms, whose relations to each other are described with the usual alphabetic labels. First, the total of the terms with no electron contribution is listed (term (A)), followed by terms (B) and (C), the nuclear-nuclear and nuclear-solvent energies.

Next, the total one-electron energy is listed, along with its three components, the electron-nuclear, electron-solvent, and kinetic energies. The total two-electron energy, and the total of the one- and two-electron energies, the electronic energy, follow. Term (N), the total of the zero-, one-, and two-electron terms, is then listed, with the label "Total quantum mech. energy." This term corresponds to the final energy from the scf energy table for that iteration, and includes the entire energies for the molecule-solvent interactions.

The output next includes the gas phase and the solution phase energies for the molecule, since these terms are, of course, necessary for solvation energy calculations. The first solution phase energy component is the total solute energy, which includes the nuclear-nuclear, electron-nuclear, kinetic, and two-electron terms, but no terms involving the solvent directly. The second component of the solution phase energy is the total solvent energy, which is computed as half of the total of the nuclear-solvent and electron-solvent terms, since some of its effect has already changed the solute energy. Third, a solute cavity term, which computes the solvation energy of a nonpolar solute of identical size and shape to the actual solute molecule, as described in reference [15], is included. This is only done for water as solvent. The last solution phase energy component (shown only if it is nonzero) is term (T), the first shell correction factor, which depends on the functional groups in the molecule, with atoms near the surface contributing most heavily.

Finally, the list ends with the reorganization energy and the solvation energy. The reorganization energy is the difference between the total solute energy and the gas phase energy, and does not explicitly contain solvent terms. The final solvation energy is calculated as the solution phase energy described above minus the gas phase energy.

The results of the self-consistent reaction field iterations so far performed are summarized after the scf output in the output from the program sole. An example from the final SCRF iteration of water in cyclohexane follows:

```
 start of program sole
   SCRF                 solvation energy
```

```
  iteration          Hartrees           kcal/mol
     0               0.0000000           0.0000
     1              -0.0061426          -3.8546
     2              -0.0065285          -4.0967
     3              -0.0065493          -4.1097

  stopping: solvation energy converged

  iterations:   3    sfinal:     -4.1097 kcal/mol

  end of program sole
```

The solvation energy is listed in Hartrees and in kcal/mol, and the note below it reads either `solvation energy not yet converged...` or `stopping: solvation energy converged`, depending on whether the solvation energy has changed by less than the Solvation convergence criterion, which is described in Section 3.8 on page 58. If the solvation energy has converged, the output from the `sole` program includes a line summarizing the solvation energy iterations and result.

The output from `ch` and `post` appears below the `sole` output. If the solvation energy has converged, the `ch` output reflects the system's final atomic charges. If the solvation energy has not converged, these charges and the Poisson-Boltzmann solver's files generated by the `post` program are passed to the solver again, and the solvation iterations continue as previously described, until solvation energy convergence is reached.

6.3.5.2 Output from an SM8 Calculation

When an SM8 solvation calculation is performed in Jaguar, the gas-phase wave function is first calculated, followed by the solution-phase wave function. Once the solution-phase calculation has completed, the various components of the solvation free energy are printed out. Atomic Löwdin populations, atomic redistributed Löwdin populations (if diffuse basis functions are used), Mayer bond orders, and CM4 partial atomic charges in the gas phase and solution phase are also computed as part of a normal SM8 calculation.

Below, the output from an SM8 calculation of methylamine in water (B97-1/6-31G* level of theory) is given. (SM6 output is similar.) The input file that was used to generate this output is given here:

```
&gen basis=6-31g* dftname=b97-1 isolv=6
&
&zmat
N    -0.760740     0.000000    -0.137251
H    -1.124354    -0.811712     0.387157
H    -1.124358     0.811709     0.387157
C     0.709471     0.000000     0.018344
H     1.090939     0.000000     1.056719
```

```
H     1.113065    -0.885399    -0.490174
H     1.113060     0.885402    -0.490171
&
```

After the gas-phase SCF calculation is complete, molecular properties are computed and printed out from the program `ch`. For SM6 calculations, these include, in addition to any other molecular properties that you specify, atomic Löwdin populations, atomic redistributed Löwdin populations (if diffuse basis functions are used), and CM4 partial atomic charges. This output is given here:

```
 start of program ch
   Atomic charges from Lowdin population analysis:

  Atom       N1           H2           H3           C4           H5
  Charge   -0.56394      0.25340      0.25340     -0.37504      0.12517

  Atom       H6           H7
  Charge    0.15351      0.15351

  sum of atomic charges:      .000000

Atomic charges from CM4:

  Atom       N1           H2           H3           C4           H5
  Charge   -0.71680      0.28492      0.28492     -0.01085      0.03511

  Atom       H6           H7
  Charge    0.06135      0.06135

   sum of atomic charges:      .000000

   end of program ch
```

Once all of the gas-phase molecular properties have been computed, the solution-phase SCRF calculation is performed. Output from this part of the calculation is printed out from the program `scf`. This output is given here:

```
...

  Energy components, in hartrees:
    (A)  Nuclear repulsion...........      41.63186140751
    (E)  Total one-electron terms.....    -210.88947153242
    (F)    Electron-nuclear...........    -305.78354569626
    (H)    Kinetic....................      94.89407416384
    (I)  Total two-electron terms.....      73.44171929988
    (J)    Coulomb....................      87.59516649117
    (K)    Exchange+Correlation.......     -14.15344719129
    (L)  Electronic energy...........    -137.44775223253  (E+I)
    (M)  -V/T........................       2.00971416465  (-(A+F+I)/H)
```

Just as for the gas-phase SCF calculation, the program `scf` lists energetic results for each iteration of the solution-phase calculation. Then the various components, (A)–(M), of the solution-phase energy or free energy, are given. These terms have the same meaning as for the converged gas-phase wave function, except the values listed in this table correspond to the converged solution-phase wave function. In particular, the quantity called electronic energy is not the sum of the internal electronic energy of the solute plus the polarization free energy; it is only the former. Similarly, the polarization free energy is not included in the one-electron electron-nuclear energy.

Immediately following this output, the various components of the solution-phase free energy are printed out. This output is given here:

```
Summary of solvation calculation by SM8
-----------------------------------------------------------------------------
solvent: water
-----------------------------------------------------------------------------
(0)  E-EN(g) gas-phase elect-nuc energy                 -95.816391725 a.u.
(1)  E-EN(liq) elec-nuc energy of solute                -95.815890825 a.u.
(2)  G-P(liq) polarization free energy of solvation            -3.760 kcal/mol
(3)  G-ENP(liq) elect-nuc-pol free energy of system     -95.821882303 a.u.
(4)  G-CDS(liq) cavity-dispersion-solvent structure
     free energy                                               -1.203 kcal/mol
(5)  G-P-CDS(liq) = G-P(liq) + G-CDS(liq) = (2) + (4)          -4.962 kcal/mol
(6)  G-S(liq) free energy of system = (1) + (5)         -95.823798991 a.u.
(7)  DeltaE-EN elect-nuc reorganization
     energy of solute molecule (7) = (1) - (0)                  0.314 kcal/mol
(8)  DeltaG-ENP elect-nuc-pol free energy
     of solvation (8) = (3) - (0)                              -3.445 kcal/mol
(9)  DeltaG-S free energy of solvation
     (9) = (6) - (0)                                           -4.648 kcal/mol
------------------------------------------------------------------------------
```

The value listed in the first line, `E-EN(g)`, is the total electronic and nuclear energy in the gas phase, as reported for the gas-phase SCF calculation. It is followed by the internal solute energy in the solution phase, `E-EN(liq)`, as reported for the solution-phase SCF calculation.

Next is the polarization free energy of solvation, `G-P(liq)`, that results from using the converged solution-phase partial atomic charges. The sum of the solute's internal energy in the solution phase, `E-EN(liq)`, and the polarization free energy, `G-P(liq)`, is listed in the next row (this sum is called `G-ENP(liq)`), followed by the first-solvation-shell contribution to the solvation free energy `G-CDS(liq)`. The next line lists the sum `G-P(liq) + G-CDS(liq)`, called `G-P-CDS(liq)`. After `G-P-CDS(liq)`, the output shows the total solution-phase free energy of the system, `GS(liq)`, which is the sum of `E-EN(liq)` and `G-P-CDS(liq)`.

The values for `Delta-EN`, `DeltaG-ENP`, and `DeltaG-S` are listed in the final three rows. `Delta-EN` is the energy difference between the total internal energy in the solution phase, `E-EN(liq)`, and the total energy in the gas phase, `E-EN(g)`. This energy should always be positive because it represents the energy that was spent distorting the optimized gas-phase molecular orbitals in order to polarize the molecular system. `DeltaG-ENP` is the solvation free energy computed without adding first-solvation-shell contributions. The final value listed this table, `DeltaG-S`, is the standard-state free energy of solvation of the system, for a temperature of 298 K and for a standard state that uses a concentration of 1 M in both the gaseous and solution phases.

After the various components of the solvation free energy have been printed out, molecular properties in the solution phase are printed out from the program `ch`. These have the same form as for the gas phase properties, shown above.

For expert users, advanced output can be obtained by specifying **ip378** = 2 in the **gen** section of the input file. When this option is specified, the gas-phase and solution-phase Mayer bond order matrix, the atomic radii used to calculate `G-P(liq)` and `G-CDS(liq)`, a breakdown of `G-CDS(liq)` and `G-P(liq)` by atom, and a table listing the individual contributions to `G-CDS(liq)` are printed out.

6.3.6 Geometry Optimization in Solution

Geometry optimizations in solution contain output in the formats described in the previous two subsections, but the optimization output and the solvation calculation output alternates as the calculation proceeds. First, by default, Jaguar computes a gas phase optimized geometry, for which the output is the same as that described above for a standard optimization. Next, the SCRF procedure is used to compute a wave function for the solvated system, as for a single point solvation energy calculation. When the solvation energy has converged, Jaguar runs the program `pbf` once more to get the solvation-related gradient. This `pbf` output does not contain the usual solvent accessible surface area and cavity energy terms. The programs `der1a`, `dsolv`, `rwr`, and `der1b` then compute the forces, with the force table in the `der1b` output in the usual manner, and the program `geopt` computes the new molecular structure, as usual. For each new structure generated during the optimization, Jaguar first performs the SCRF calculation, then obtains the forces (in solution), and finally generates a new structure. The calculation proceeds until the geometry optimization convergence criteria are reached. The convergence criteria for optimizations in solution are three times larger than they are for optimizations in the gas phase.

For solvated geometry optimizations, the solvation energy is computed as the difference between the energy of the optimized gas phase structure and the energy of the solvated structure that was optimized in solution.

6.3.7 Properties

If you make any non-default selections from the Properties tab, the program `ch` runs and writes the results to the output file after the SCF iterations, if any.

6.3.7.1 Multipole Moments and Charge Fitting

When multipole moments are calculated, the x, y, and z components of the dipole moment and the total magnitude of the dipole moment μ are reported in debye, followed by information on any requested higher-order moments and the corresponding traceless higher-order moment tensors. An example of the output for the dipole and quadrupole moments of water follows:

```
 Moments from quantum mechanical wave function:
 Dipole Moments  (Debye)
    X=     0.0000     Y=     2.1470     Z=     0.0000  Tot=     2.1470

 Quadrupole Moments  (Debye-Ang)
   XX=    -4.0828    YY=    -5.7670    ZZ=    -7.1340
   XY=     0.0000    XZ=     0.0000    YZ=     0.0000
 Traceless Quadrupole Moments  (Debye-Ang)
XX-YY=     1.6843             2ZZ-XX-YY=    -4.4182
   XY=     0.0000    XZ=     0.0000    YZ=     0.0000
```

If electrostatic potential charge fitting to atomic centers is performed, the output lists the number of grid points from the charge grid, which is used for the charge fit. It then describes the constraint or constraints for the fit, followed by the calculated atomic charges and their sum. The root mean square error of the charge fitting is also reported; this error is calculated from examining the Coulomb field at each grid point that would result from the fitted charges, and comparing it to the actual field.

If electrostatic potential fitting to atomic centers and bond midpoints is performed, the bond midpoints are treated as “dummy atoms” and their descriptions and coordinates are provided before the grid points information. The bond charges from the fit are provided, with the label “bond,” along with those on the atomic centers. An example for water follows:

```
 dummy atom x4   is between           2 and           1
 dummy atom x5   is between           3 and           1

                               angstroms
 atom           x                 y                 z
 O          0.0000000000     -0.1135016000      0.0000000000
 H1         0.7531080000      0.4540064000      0.0000000000
 H2        -0.7531080000      0.4540064000      0.0000000000
 x4         0.3765540000      0.1702524000      0.0000000000
 x5        -0.3765540000      0.1702524000      0.0000000000

 gridpoints used for charge fit        4162
```

```
    out of a possible maximum of         4188

   Electrostatic potential fitting constrained to reproduce
     total charge:                 yes
     dipole moment:                no
     traceless quadrupole moment: no
     traceless octupole moment:   no

  Atomic charges from electrostatic potential:

 Atom        O        H1       H2       x4       x5
 Charge   -0.31208  0.63681  0.63681 -0.48077 -0.48077

  sum of atomic charges:     0.000000

 RMS Error   8.26E-04 hartrees
```

If the fit is constrained to reproduce the dipole moment (or dipole and higher moments), or any other time both electrostatic potential fitting and multipole moment calculations are performed, new moments can be calculated from the fitted charges, as described in Section 3.9.1 on page 62. The output from ch begins with the moments calculated for the quantum mechanical wave function, in the format for multipole moment calculations. Next, the electrostatic potential fitting information is provided, as described above. Finally, the components and totals of the moments recalculated using the electrostatic potential charges are reported.

6.3.7.2 Polarizabilities and Hyperpolarizabilities

If you calculate polarizabilities and hyperpolarizabilities with the coupled perturbed HF method, the tensor elements in au appear in the output from the program cpolar, which runs after the SCF calculation. Alternatively, if you use the finite field method to calculate the polarizability and/or first hyperpolarizability of the molecule, the output includes data from all the SCF calculations involved. (See Section 3.9 on page 62 for details on the methods used to calculate polarizability and hyperpolarizability.) The data from the program scf includes the virial ration –V/T. Before each SCF calculation used for the polarizability evaluation, the program polar runs and outputs the electric field (in au) used for the SCF calculation whose output appears immediately afterwards. When all calculations needed for the finite difference method have been performed, the program polar outputs the polarizability tensor in au, the first hyperpolarizability tensor in au, if it has been calculated, and the dipoles from each SCF calculation, along with information about the electric fields used for the dipole calculations.

An example of output from a polarizability and hyperpolarizability calculation follows:

```
polarizability (in AU):
 alpha(x x)= 5.551 alpha(x y)= 0.000 alpha(x z)= 0.000
 alpha(y x)= 0.000 alpha(y y)= 5.245 alpha(y z)= 0.000
 alpha(z x)= 0.000 alpha(z y)= 0.000 alpha(z z)= 11.890
```

```
alpha= 7.562
Dalpha= 6.497

first hyperpolarizability (in AU):
beta(x,x,x)= 0.000
beta(y,y,y)= 0.000
beta(z,z,z)= -10.206
beta(x,y,y)= 0.000
beta(x,z,z)= 0.000
beta(y,x,x)= 0.000
beta(y,z,z)= 0.000
beta(z,x,x)= 0.435
beta(z,y,y)= 0.404
beta(x,y,z)= 0.000
sum beta(x)= 0.000
sum beta(y)= 0.000
sum beta(z)= -9.367
beta= -5.620

second hyperpolarizability (in AU):
gamma(x,x,x,x)= 9.110
gamma(y,y,y,y)= 11.758
gamma(z,z,z,z)= 28.020
gamma(x,x,x,y)= 0.000
gamma(x,x,x,z)= 0.000
gamma(x,x,y,y)= 1780.861
gamma(x,x,y,z)= 0.000
gamma(x,x,z,z)= -2.950
gamma(x,y,y,y)= 0.000
gamma(x,y,y,z)= 0.000
gamma(x,y,z,z)= 0.000
gamma(x,z,z,z)= 0.000
gamma(y,y,y,z)= 0.000
gamma(y,y,z,z)= -5.235
gamma(y,z,z,z)= 0.000
gamma= 718.848
```

After the tensor matrix elements the program prints various sums of these matrix elements. For the polarizability, the quantities α and $\Delta\alpha$ are reported as alpha and Dalpha, defined as follows:

$$\alpha = (\alpha_{xx} + \alpha_{yy} + \alpha_{zz}) / 3$$

$$\Delta\alpha = \sqrt{(\alpha_{xx} - \alpha_{yy})^2 + (\alpha_{yy} - \alpha_{zz})^2 + (\alpha_{zz} - \alpha_{xx})^2}$$

For the first hyperpolarizability, three sums are reported, which are defined by the following expression

$$\Sigma\beta_q = \beta_{qxx} + \beta_{qyy} + \beta_{qzz}$$

where q can be x, y, or z. The average hyperpolarizability β is defined by

$$\beta = \frac{3}{5\mu}(\mu_x\Sigma\beta_x + \mu_y\Sigma\beta_y + \mu_z\Sigma\beta_z)$$

where μ is the dipole moment. The average second hyperpolarizability γ is defined by

$$\gamma = \frac{1}{5}\sum_p\sum_q\gamma_{ppqq}$$

where p and q run over the three coordinates, x, y, and z.

6.3.7.3 Electron Density

If you choose to calculate the electron density, the output from the program `elden` appears below the SCF output. The output lists the number of grid points used for the electron density calculation and the total number of electrons found over the grid. The main output file does not include the charges and grid points for the calculation; that information can be found in the output file *jobname*`.chdens`, where *jobname*`.in` is the input file for the Jaguar job. The file *jobname*`.chdens` lists the Cartesian coordinates and the electron density in au, respectively, for each grid point.

6.3.7.4 Mulliken Populations

If you calculate Mulliken populations by atom, the atomic charges and their sum is given under the heading "Atomic charges from Mulliken population analysis." If you calculate them by basis function, the atomic charge output is preceded by a section labeled "Mulliken population for basis functions," listing the atom label, basis function index, basis function type (S for s, X for p_x, etc.), and calculated population. If you calculate them by bond, the populations by atom and basis function are given as well. An example for water in a 6-31G** basis set follows.

```
Mulliken Bond Populations:  first nearest neighbor
Atom1 Atom2  Pop.  Atom1 Atom2  Pop.  Atom1 Atom2  Pop.  Atom1 Atom2  Pop
H1    O     0.314  H2    O     0.314

Mulliken Bond Populations:  second nearest neighbor
Atom1 Atom2  Pop.  Atom1 Atom2  Pop.  Atom1 Atom2  Pop.  Atom1 Atom2  Pop
H2    H1   -0.025

Mulliken population for basis functions
atom  func.  type   population
O       1       S     1.9954
O       2       S     0.8942
O       3       X     0.8034
O       4       Y     0.9514
```

```
  O       5       Z     1.1426
  O       6       S     0.8865
  O       7       X     0.4669
  O       8       Y     0.6649
  O       9       Z     0.8332
  O      10      XX     0.0085
  O      11      YY     0.0024
  O      12      ZZ     0.0052
  O      13      XY     0.0142
  O      14      XZ     0.0000
  O      15      YZ     0.0021
  H1     16       S     0.4950
  H1     17       S     0.1263
  H1     18       X     0.0185
  H1     19       Y     0.0138
  H1     20       Z     0.0111
  H2     21       S     0.4950
  H2     22       S     0.1263
  H2     23       X     0.0185
  H2     24       Y     0.0138
  H2     25       Z     0.0111

  Atomic charges from Mulliken population analysis:

 Atom       O        H1       H2
 Charge   -0.67059  0.33530  0.33530

  sum of atomic charges:     0.000000
```

You may find it helpful to select the Gaussian function list (basis set) setting in the Output tab if you want to have more information about the basis functions. More information on this output option is given in Section 3.13 on page 82.

If both Mulliken populations and multipole moments are calculated, the multipole moments are calculated from the atomic Mulliken populations as well as directly from the wave function, as noted in Section 3.9.2 on page 64. The output lists the multipole moments from the wave function, as described earlier; the Mulliken populations, as described just above; and finally, the recalculated moments resulting from the Mulliken charges, in the same format used for the earlier moment output.

6.3.7.5 Atomic Fukui Indices

Jaguar calculates atomic Fukui indices according to the method specified in Refs 238 and 239. These indices are based on Fukui functions, which are partial derivatives of the electron or spin density with respect to a change in either the electron count or the unpaired spin count. A change in the electron count would be induced by reaction with another molecule, or by any external charge transfer mechanism. A change in the number of unpaired electron spins would be induced by electromagnetic radiation so as to produce an electronically excited state of different spin multiplicity.

Due to the assumption of the frozen-orbital approximation, the Fukui functions may be simply calculated from the MO expansion coefficients and the atomic basis functions. The integrated Fukui functions may then be calculated in terms of the MO expansion coefficients and the atomic overlap matrix elements. In this form the integrated Fukui functions are calculated the same way as Mulliken populations. It follows that the atomic Fukui index denoted f_NN for the HOMO is simply half of the atom's contribution to the Mulliken population of the HOMO. (The factor of one half is due to the fact that the Fukui function for the electron density integrates to 1, while the Mulliken population of a filled molecular orbital is 2.)

Atomic Fukui indices are an attempt to quantify the anticipated reactivity of a molecule in various types of reactions. For a given reference atom in a series of related molecules, one can rank the molecules according to the values of a particular Fukui index for that atom. Each index includes two subscripts, N or S, which represent the electron density and the spin density, respectively. The two indices indicate the nature of the partial derivative on which the index is based. The first index indicates the property that responds to a change in the property indicated by the second index. Thus, f_NS indicates the change in the electron density about an atom when the molecule undergoes a reaction in which its spin multiplicity changes. Usually the f_NN indices are of interest, because the other indices require a change in either the spin density or the spin multiplicity.

The f_NN HOMO indices are related to the f- Fukui function, and the f_NN LUMO indices are related to the f+ Fukui function. (The f- and f+ functions are calculated by the `fukui.py` script as discussed in Section 5.2 on page 117.) Both atomic Fukui indices and Fukui functions are intended to predict and quantify relative reactivity. Sometimes the Fukui indices are called "condensed-to-atom Fukui functions", but in practice the Fukui indices may or may not be derived from the Fukui functions themselves. In Jaguar, the atomic Fukui indices are calculated only for the N-electron system, using only the MO coefficients from the frontier orbitals. Fukui functions are molecular properties, and are calculated as differences in the electron density between the N-electron system on the one hand, and the N+δ and N-δ systems on the other hand (δ=0.01 by default—see Section 5.2 on page 117). Thus the Fukui functions depend upon all of the occupied orbitals, and they allow for the effects of electronic relaxation upon ionization of the molecule.

A high positive value of f_NN for the HOMO indicates that the atom can donate electrons, and thus behaves as a nucleophile. A high positive value of f_NN for the LUMO indicates that the atom can receive electrons, and thus behaves as an electrophile.

As an example, consider the following output for a B3LYP/6-31G** calculation on pyridine:

```
------------------------------------------------------------------------
  Atomic Fukui indices:

                         HOMO                              LUMO
             -------------------------------   -------------------------------
    Atom       f_NN    f_NS    f_SN    f_SS      f_NN    f_NS    f_SN    f_SS
  --------   ------- ------- ------- -------   ------- ------- ------- -------
  C1          0.0560 -0.0267  0.0000  0.0827    0.1094  0.0267  0.0000  0.0827
  N2          0.6873  0.2091  0.0000  0.4782    0.2690 -0.2091  0.0000  0.4782
  C3          0.0560 -0.0267  0.0000  0.0827    0.1094  0.0267  0.0000  0.0827
  C4          0.0541 -0.0131  0.0000  0.0672    0.0803  0.0131  0.0000  0.0672
  C5          0.0048 -0.1713  0.0000  0.1760    0.3473  0.1713  0.0000  0.1760
  C6          0.0541 -0.0131  0.0000  0.0672    0.0803  0.0131  0.0000  0.0672
  H7          0.0260  0.0127  0.0000  0.0133    0.0006 -0.0127  0.0000  0.0133
  H9          0.0259  0.0127  0.0000  0.0133    0.0006 -0.0127  0.0000  0.0133
  H10         0.0129  0.0063  0.0000  0.0067    0.0004 -0.0063  0.0000  0.0067
  H11         0.0099  0.0039  0.0000  0.0061    0.0022 -0.0039  0.0000  0.0061
  H12         0.0129  0.0063  0.0000  0.0067    0.0004 -0.0063  0.0000  0.0067
```

The nitrogen atom in pyridine is known to be somewhat nucleophilic, and this is reflected in the large positive value of f_NN for the HOMO. The HOMO is dominated by the lone pair of electrons on the nitrogen atom. When pyridine reacts, these electrons are donated to form the new bond to the other molecule, and thus are lost from the HOMO.

If pyridine were considered as an electrophile, you would look at the f_NN values for the LUMO. In this case the largest index is that for the para carbon atom. Note that the f_NN value for the nitrogen atom is large in both the HOMO and LUMO, suggesting that it could act as an electrophile or a nucleophile, whereas experimentally it is generally known as a nucleophile only. The large f_NN LUMO value is a consequence of the fact that the LUMO has large components on the out-of-plane p orbital on the nitrogen atom. In fact it is often the case that the same atoms will have large Fukui indices for both HOMO and LUMO, and this reflects the fact that the HOMO and LUMO tend to localize on the same atoms. For this reason, atomic Fukui indices are not reliable for identifying the most reactive nucleophilic or electrophilic site within a particular molecule. They are more useful in finding trends in reactivity for a chosen atomic site in a series of related molecules.

Some points to remember:

- The values for f_NN and f_SS sum to 1
- The values for f_NS and f_SN sum to 0
- The values for f_SN are zero for all atoms if you use a spin-restricted formalism when performing the calculation.

f_NN and f_SS are normally positive numbers for all atoms. Negative values are artifacts, and will usually be negligibly small in magnitude. These artifacts have the same origin as negative Mulliken populations.

6.3.7.6 NBO Calculations

Output for NBO calculations appears under the heading “Jaguar NBO 6.0.”

6.3.7.7 ESP Analysis

ESP analysis lists various statistical quantities related to the ESP surface [242], including the maximum and minimum values of the ESP on the isodensity surface, the variances of the ESP within the positive-valued and negative-valued regions, and the total variance, a “balance” parameter, which quantifies how the positive and negative variances compare to each other, and a measure of charge separation, called the “local polarity”, which is the average deviation of the ESP from its overall mean value. Even for symmetric molecules with no dipole moment, there can still be strong charge separation, which is reflected in the local polarity. Perfect balance is indicated by a value of 0.25, which is also the maximum possible value.

Subsets of these seven ESP-based properties have been shown to correlate well with various bulk-phase properties. See [242] and references therein, and Ref 243.

In addition to the above molecular properties, the maximum and minimum ESP values on the molecular surface for each atom are also reported. The ESP values on the surface are assigned to the nearest atomic nuclei. You may add dummy atoms to mark particular points in space where there are no real atoms (such as in the interior of a ring) to determine the range of ESP values nearby. The maximum and minimum atomic ESP values are written to the output Maestro file and can be used to label atoms in the Workspace. Because these values are calculated by Jaguar and not by Maestro, they will not be updated if you change the isovalue used to display the electron density in Maestro. If you want the ESP to be evaluated on an isodensity contour that is different from the default value of 0.001 electrons/bohr^3, you must set the desired contour value with the **den_isoval** keyword and run another Jaguar job.

Here is an example of the ESP analysis output for a HF/6-31g** calculation on H2O:

```
Analysis of ESP on isodensity surface:
  -------------------------------------

  Density isovalue:  0.00100 electrons/bohr**3,  0.00015 electrons/A**3
  Number of surface points:      1450
  Area of isosurface:    152.7 bohr**2,     42.8 A**2

  Minimum ESP value:                 -41.62 kcal/mol
  Maximum ESP value:                  47.56 kcal/mol
  Mean ESP value:                      0.51 kcal/mol
  Mean positive ESP value:            26.09 kcal/mol
  Mean negative ESP value:           -23.97 kcal/mol
  Number of positive ESP values:        709
  Number of negative ESP values:        741
  Variance of positive ESP values:   186.65 (kcal/mol)**2
  Variance of negative ESP values:   144.23 (kcal/mol)**2
```

```
Total ESP variance:                    330.87 (kcal/mol)**2
ESP balance:                             0.246
Local polarity:                          25.02 kcal/mol

  Atom    Max. ESP  Min. ESP
--------  --------  --------
o           -11.63    -41.62
h1           47.56    -22.18
h2           47.37    -21.57
```

6.3.8 Frequency, IR Intensity, and Thermochemistry Output

If you calculate vibrational frequencies, any SCF calculations during the run use the RMS density change convergence criterion described in Section 3.7 on page 53 instead of the usual energy convergence criterion. Therefore, these SCF calculations often proceed for several more iterations than single point energy calculations do.

To compute the Hessian for vibrational frequencies, Jaguar calculates the second derivatives either analytically or numerically as the derivatives of the analytical first derivatives, depending on the type of calculation (see Section 3.10 on page 69 for details). Whenever numerical second derivatives are computed after an SCF calculation—whether for frequency output, for an initial Hessian, or for updating during geometry optimization—the programs `nude`, `onee`, `hfig`, `grid`, `rwr`, `scf`, `der1a`, `rwr`, and `der1b` run, setting up and performing SCF calculations and evaluating analytic gradients at $6N_{atom}$ perturbed geometries (unless the number of perturbed geometries needed is reduced by the use of molecular symmetry). After the calculations at the perturbed geometry, Jaguar performs one final calculation at the unperturbed geometry. (The Jaguar programs run may vary slightly for non-HF calculations, as described earlier in this section.) After the data from all perturbed geometries is collected, the program `nude` prints the numerical first derivatives in a force table similar to the usual geometry optimization force table described earlier in this section. The output then lists the matrix indices of the most asymmetrical Hessian element before symmetrization. (The symmetrized numerical Hessian is not printed in the output, but can be found in the restart file, which is discussed in Section 7.5 on page 174.)

For either analytic or numerical frequency calculations, the output from the program `freq` contains the actual frequencies and normal modes from the computed Hessian, or from the last available Hessian (generally the initial Hessian guess) if you used the Use available Hessian choice to request vibrational frequencies. The output from the program `freq` first lists the harmonic frequencies in cm^{-1} and their symmetries (if symmetry is on for the job), then the normal modes. The system's thermochemical properties, the constant volume heat capacity (C_v), entropy (S), enthalpy (H), internal energy (U), Gibbs free energy, and logarithm of the partition function (ln Q) are then listed for the specified pressure and temperatures, as well as at 0 K. Here is an example of this output from a vibrational frequency calculation on FOOF:

```
start of program freq

harmonic frequencies in cm**-1, reduced masses in amu,
force constants in mDyne/A, and
normal modes in cartesian coordinates:
IR intensities in km/mol

frequencies       208.85   555.11   707.73  1135.28  1145.49  1162.53
symmetries       A        A        B        B        A        A
intensities         0.48     0.22     9.07    66.55    34.46     0.43
reduc. mass         5.84     4.35     4.66     4.39     4.47     7.14
force const         0.15     0.79     1.37     3.33     3.46     5.69
  f1       X     0.04113  0.08347  0.06928  0.00885 -0.00551 -0.00792
  f1       Y     0.11637  0.05538  0.05480 -0.07045  0.06790  0.02561
  f1       Z    -0.06710  0.04044  0.02627 -0.07517  0.07145  0.02804
  o1       X     0.01465 -0.03712 -0.08228 -0.01051 -0.03085  0.16720
  o1       Y    -0.03534  0.11710 -0.06509  0.08368 -0.10738  0.02016
  o1       Z     0.07971 -0.04803  0.10082  0.10699 -0.08487 -0.03330
  o2       X    -0.01465  0.03712 -0.08228 -0.01051  0.03085 -0.16720
  o2       Y     0.03534 -0.11710 -0.06509  0.08368  0.10737 -0.02016
  o2       Z     0.07971 -0.04803 -0.10082 -0.10699 -0.08487 -0.03330
  f2       X    -0.04113 -0.08347  0.06928  0.00885  0.00551  0.00792
  f2       Y    -0.11637 -0.05538  0.05480 -0.07045 -0.06790 -0.02561
  f2       Z    -0.06710  0.04044 -0.02627  0.07517  0.07145  0.02804

Thermochemical properties at    1.0000 atm

rotational symmetry number:  2
rotational temperatures (K):   1.122334   0.289429   0.255263

vibrational temperatures:
     mode:      1        2        3        4        5        6
temp. (K):   300.48   798.68  1018.26  1633.42  1648.10  1672.61

Thermodynamic properties calculated assuming an ideal gas

In the table below, the units for temperature
are kelvins, the units for U, H, and G are
kcal/mol and the units for Cv and S are cal/(mol K)

The zero point energy (ZPE):      7.026 kcal/mol
is not included in U, H, or G in the table below

T =   298.15 K

           U         Cv          S          H          G        ln(Q)
        ---------  ---------  ---------  ---------  ---------  ---------
trans.      0.889      2.981     38.654      1.481    -10.044   16.95158
  rot.      0.889      2.981     22.198      0.889     -5.730    9.67056
  vib.      0.568      4.495      3.038      0.568     -0.338    0.57052
```

```
   elec.      0.000      0.000      0.000      0.000      0.000    0.00000
   total      2.345     10.457     63.891      2.938    -16.111   27.19266


  Total internal energy, Utot (SCFE + ZPE + U):    -348.207340 hartrees
              Total enthalpy, Htot (Utot + pV):    -348.206396 hartrees
    Total Gibbs free energy, Gtot (Htot - T*S):    -348.236753 hartrees

  end of program freq
```

If infrared intensities were calculated, several additional programs are run after the first run of the program `scf`. These programs compute the derivatives of the dipole moment, which are needed to calculate the IR intensities. The IR intensities are listed in the frequencies table. If Raman intensities were calculated, they are listed in the frequencies table also.

6.3.9 CIS Calculations

Output from CIS calculations of excited states lists the selected excited states with the excitation energy from the ground state in eV and the transition wavelength, the orbital excitations involved in the CI wave function, and the transition dipole moment and oscillator strength. The following example is for H_2O.

```
Excited State 1: 9.3904 eV 132.03 nm

orbitals in
excitation   CI coeff.
------------ ---------
5 => 6 -0.95735
5 => 11 -0.26823

Transition dipole moment (debye):
X= 0.0000 Y= -1.3726 Z= 0.0000 Tot= 1.3726

Oscillator strength, f= 0.0671
----------------------------------------------------------------------------

Excited State 2: 11.5077 eV 107.74 nm

orbitals in
excitation   CI coeff.
------------ ---------
5 => 7 -0.82270
5 => 10 -0.55056
5 => 14 -0.11504
Transition dipole moment (debye):
X= 0.0000 Y= 0.0000 Z= 0.0000 Tot= 0.0000

Oscillator strength, f= 0.0000
----------------------------------------------------------------------------
```

From this example, the lowest excited state of water is mostly a transition from orbital #5 (the HOMO) to orbital #6 (the LUMO). The transition dipole moment is the amplitude that is used to evaluate the oscillator strength, which is a measure of the transition probability. Oscillator strengths near 1 indicate strongly allowed transitions. From the above output, an electronic transition to the second excited state of water is not a dipole-allowed transition, due to symmetry, while the transition to the first excited state is predicted to be only weakly allowed.

6.3.10 Basis Set

If you selected Gaussian function list (basis set) in the Output tab or set **ip1**=2 in the **gen** section, a list of atoms and the basis set used for each atom is given, followed by two tables that provide information about the basis set.

The functions in a basis set are made up of polynomials of the appropriate degree multiplied by linear combinations of Gaussian primitives of the form $N\exp(-zr^2)$, where *N* is a normalization constant and *z* is the exponent of the primitive Gaussian. If the linear combination has only one Gaussian primitive, the function is called uncontracted; otherwise, it is called a contracted Gaussian. Each shell is defined by a product of a polynomial and a Gaussian primitive.

The shell information table is printed first. An example for water with a 6-31G** basis set is given below. The first column indicates which atom the shell is centered on. The second column lists the shell numbers, which increase consecutively for each atom. The values in the third column mean different things depending on their sign. The positive numbers mean that the basis function currently being described is composed of that number of primitive Gaussians, starting with the primitive Gaussian for that row and including the appropriate number of rows immediately below it. The negative numbers' magnitudes indicate the first shell which contributes to the same contracted Gaussian function. In the example below, the first row has a jcont value of 6, indicating that the first basis function being described is a contracted Gaussian composed of that primitive Gaussian and the primitives in the next five rows. The jcont values of 1 in the next five rows indicate that the primitive Gaussians being described are components in a contracted function whose first primitive Gaussian term is listed in the first row.

The values in the column marked "ishl" take on nonzero values when basis functions corresponding to different *l* values, as described in the next column, use primitive Gaussians with the same exponents. Positive values indicate that the same exponents are used in the shell listed that number of rows down; a value of 1 indicates that the exponents are provided from a shell listed earlier. The *l* values in the next column indicate the angular momentum: a value of 1 corresponds to an s function, 2 indicates a p function, 3 a d function, and so on. The nfsh values are zero-based pointers to the basis function list.

The column labeled z lists the exponents for the primitive Gaussians, while the "coef" column lists the coefficient of their contribution to the linear combination comprising the basis function. Note that the uncontracted basis functions, those with jcont values of 1, have "coef" values of exactly 1. Finally, the product of the "coef" value and the normalization constant for the primitive Gaussian shell, *N*, is listed in the column labeled "rcoef."

```
  Gaussian Functions - Shell information

         s    j
         h    c  i      n
         e    o  s      f
         l    n  h      s
atom     l    t  l  l   h           z             coef             rcoef
----   ---  --- -- --  ---     ----------      ----------       ---------
O        1    6  0  1   0    5484.6716600       0.0018311       0.8317237
O        2   -1  0  1   0     825.2349460       0.0139502       1.5308156
O        3   -1  0  1   0     188.0469580       0.0684451       2.4771485
O        4   -1  0  1   0      52.9645000       0.2327143       3.2562811
O        5   -1  0  1   0      16.8975704       0.4701929       2.7928934
O        6   -1  0  1   0       5.7996353       0.3585209       0.9549377
O        7    3  3  1   1      15.5396162      -0.1107775      -0.6179340
O        8   -7  3  1   1       3.5999336      -0.1480263      -0.2757209
O        9   -7  3  1   1       1.0137618       1.1307670       0.8142076
O       10    3 -1  2   2      15.5396162       0.0708743       3.1169443
O       11  -10 -1  2   2       3.5999336       0.3397528       2.4014375
O       12  -10 -1  2   2       1.0137618       0.7271586       1.0543604
O       13    1  1  1   5       0.2700058       1.0000000       0.2669562
O       14    1 -1  2   6       0.2700058       1.0000000       0.2774320
O       15    1  0  3   9       0.8000000       1.0000000       1.1138249
H1       1    3  0  1  15      18.7311370       0.0334946       0.2149354
H1       2   -1  0  1  15       2.8253944       0.2347270       0.3645712
H1       3   -1  0  1  15       0.6401217       0.8137573       0.4150514
H1       4    1  0  1  16       0.1612778       1.0000000       0.1813806
H1       5    1  0  2  17       1.1000000       1.0000000       1.6057611
H2       1    3  0  1  20      18.7311370       0.0334946       0.2149354
H2       2   -1  0  1  20       2.8253944       0.2347270       0.3645712
H2       3   -1  0  1  20       0.6401217       0.8137573       0.4150514
H2       4    1  0  1  21       0.1612778       1.0000000       0.1813806
H2       5    1  0  2  22       1.1000000       1.0000000       1.6057611
```

The second table, an example of which follows below, shows information for the Cartesian components of each shell. For instance, the entries X, Y, and Z for the tenth shell correspond to p_x, p_y, and p_z functions. The normalization for each Cartesian component depends on the powers of *x*, *y* and *z* in the polynomial for the component. For $l = 2$ and higher, the normalization can be different for different components. The "rmfac" values provide the ratio of the normalization to that of the first component listed, which is the x^l component.

```
 Gaussian Functions - Normalized coefficients

      s
      h  t
      e  y
      l  p   f
atom  l  e   n          z          rcoef         rmfac      rcoef*rmfac
---- -- --  --      ---------    ----------    ----------   -----------
O     1  S   1    5484.671660     0.831724      1.000000       0.831724
O     2  S   1     825.234946     1.530816      1.000000       1.530816
O     3  S   1     188.046958     2.477149      1.000000       2.477149
O     4  S   1      52.964500     3.256281      1.000000       3.256281
O     5  S   1      16.897570     2.792893      1.000000       2.792893
O     6  S   1       5.799635     0.954938      1.000000       0.954938
O     7  S   2      15.539616    -0.617934      1.000000      -0.617934
O     8  S   2       3.599934    -0.275721      1.000000      -0.275721
O     9  S   2       1.013762     0.814208      1.000000       0.814208
O    10  X   3      15.539616     3.116944      1.000000       3.116944
         Y   4                                  1.000000       3.116944
         Z   5                                  1.000000       3.116944
O    11  X   3       3.599934     2.401438      1.000000       2.401438
         Y   4                                  1.000000       2.401438
         Z   5                                  1.000000       2.401438
O    12  X   3       1.013762     1.054360      1.000000       1.054360
         Y   4                                  1.000000       1.054360
         Z   5                                  1.000000       1.054360
O    13  S   6       0.270006     0.266956      1.000000       0.266956
O    14  X   7       0.270006     0.277432      1.000000       0.277432
         Y   8                                  1.000000       0.277432
         Z   9                                  1.000000       0.277432
O    15 XX  10       0.800000     1.113825      1.000000       1.113825
        YY  11                                  1.000000       1.113825
        ZZ  12                                  1.000000       1.113825
        XY  13                                  1.732051       1.929201
        XZ  14                                  1.732051       1.929201
        YZ  15                                  1.732051       1.929201
H1    1  S  16      18.731137     0.214935      1.000000       0.214935
H1    2  S  16       2.825394     0.364571      1.000000       0.364571
H1    3  S  16       0.640122     0.415051      1.000000       0.415051
H1    4  S  17       0.161278     0.181381      1.000000       0.181381
H1    5  X  18       1.100000     1.605761      1.000000       1.605761
         Y  19                                  1.000000       1.605761
         Z  20                                  1.000000       1.605761
H2    1  S  21      18.731137     0.214935      1.000000       0.214935
H2    2  S  21       2.825394     0.364571      1.000000       0.364571
H2    3  S  21       0.640122     0.415051      1.000000       0.415051
H2    4  S  22       0.161278     0.181381      1.000000       0.181381
H2    5  X  23       1.100000     1.605761      1.000000       1.605761
         Y  24                                  1.000000       1.605761
         Z  25                                  1.000000       1.605761
```

The table is followed by a list indicating the number of electrons in each atom that are treated with an effective core potential.

6.3.11 Jaguar Reactions

The output from the Jaguar - Reactions panel (or the jreactions.py script) that gives the reaction results is contained in the file *jobname*`_rxn.out`. A typical example is given below.

```
Date and time:          14 January 2014, 15:21:50

Reactant input files:
CH3CN.in
H2O.in

Product input files:
CH3CONH2.in

Reaction output file: hydrolysis_rxn.out

Reaction energetics in kcal/mol:
----------------------------------------
Delta E (gas phase)      = -33.35
Delta E (solution phase) = -30.66
Delta H (solution phase) = -26.99
Delta G (solution phase) = -14.56
```

6.3.12 Methods

If the DIIS convergence method is not used, the “maximum DIIS error” column is not printed for the table giving data from the SCF iterations. Also, if the OCBSE convergence scheme is selected, the Coulomb and exchange contributions to the total two-electron terms are listed in the SCF summary below the table.

If a fully analytic calculation is performed (see Section 3.2 on page 33), the programs `grid` and `rwr` are not run, because the all-analytic method does not use this code.

If you select a Final localization method, the output from the program `local` appears after the output from any SCF iterations and lists the orbitals that are localized. (If you want to print out the localized orbitals, you should make the appropriate selection in the Output tab, as described in Section 3.13.2 on page 86.)

6.4 The Log File

The log file, an output file which appears in the local job directory, provides information on the progress of a run. The current contents of a job's log file is displayed in the Monitor panel. The log file notes when each program within Jaguar is complete, as well as noting data from each SCF iteration. The data from the SCF iterations is shown in table form. Some of the text for the column headings should be read down rather than across.

For the table of SCF iteration information, the number of the iteration is provided first in each row, followed by a "Y" or "N" indicating whether the Fock matrix was updated or not. The Fock matrix is updated using the difference in density matrix between iterations to accumulate contributions.

The next entry indicates whether the DIIS convergence scheme was used for that iteration, also with a "Y" or "N." The DIIS method produces a new estimate of the Fock matrix as a linear combination of previous Fock matrices, including the one calculated during that iteration. DIIS, which is enabled by default, usually starts on the second iteration, and is not used on the final iteration. If the entry in this column reads "A," it indicates that DIIS was not used for that iteration, but the density matrix was averaged.

The cutoff set for each iteration is indicated under the "icut" heading. Cutoff sets are explained in the `.cutoff` file description in Section 10.5 on page 308.

The grid column lists the grid used for that iteration, which must be one of the grid types coarse (signified by a C), medium (M), fine (F), or ultrafine (U). See Section 9.5.27 on page 263 and Section 10.4 on page 305, for more information on grids and grid types.

The total energy for the molecule in Hartrees appears in the next column, followed by the energy change, which is the difference in energy from the previous iteration to the current one.

The RMS density change column provides the root mean square of the change in density matrix elements from the previous iteration to the current one.

Finally, the maximum DIIS error column provides a measure of convergence by listing the maximum element of the DIIS error vector. For HF calculations, the DIIS error vector is given by **FDS** – **SDF** in atomic orbital space, where **F**, **D**, and **S** are the Fock, density, and overlap matrices, respectively. For open shell cases, the definition of the error vector is given in reference 11.

If you are not running a default, single-point, Hartree-Fock calculation, the log file generally contains information generated from other Jaguar programs used for the run as well. This information is often a summary of what is written to the Jaguar output file. For a more detailed description of the information in the log file, see the previous sections of this chapter.

After all the individual programs necessary for that job have finished running, a note appears in the log file listing the name and location of the output file. When the job is finished, this too is noted in the log file.

Using Jaguar

This chapter provides some information on how to use Jaguar to obtain the results you want. It includes information on setting up an initial guess (especially for transition metal systems) convergence of SCF and geometry optimizations, setting up certain kinds of calculations, restarting jobs, and using Maestro and Jaguar to set up GAUSSIAN input.

For a summary of recommended settings for various calculations, see the quick reference sheet, *Recommended Settings for Jaguar*.

7.1 Choosing an Initial Guess

The speed of convergence, and sometimes simply obtaining convergence, depends critically on the initial guess. In the development of Jaguar, we have attempted to provide default initial guesses of high quality. These defaults are described below. However, there are situations in which the default initial guess for the molecule of interest does not lead to convergence or does not lead to the correct state. Some suggestions on how to obtain alternative initial guesses are provided in this section.

7.1.1 Overview

By default, for calculations on simple closed-shell systems with no transition metals, Jaguar constructs the initial wave function from orbitals that give the best overlap with previously calculated orbitals from atomic calculations.[1] The algorithm used is described in Ref. 14. This method compares well with the semi-empirical schemes that other ab initio programs use to obtain initial guesses. Other methods for the initial guess can be selected with the **iguess** keyword, including diagonalizing the one-electron Hamiltonian[2] (which is rarely the best guess), and using a superposition of atomic densities,[3] which provides a similar quality initial guess to the default. However, in cases where the default guess does not lead to the desired result, this guess could provide an alternative that does.

Jaguar also provides a unique initial guess feature to improve SCF convergence (both HF and DFT) for transition-metal-containing systems that is based upon ligand field theory. As described in Ref. 21, research at Schrödinger has established that poor convergence of these

1. Keyword **iguess** = 10 in the **gen** section.
2. Keyword **iguess** = 0 in the **gen** section.
3. Keyword **iguess** = 11 in the **gen** section.

systems is very often due to problems with the trial wave function's orbital shapes and occupations. This initial guess method takes advantage of user-provided information on charges and spins of fragments within the system, as described in Section 7.1.2 on page 168, although such information is not required. If you have an antiferromagnetic system, this algorithm does not work, but you can set keywords to ensure that you have the correct guess. See Section 7.1.2 on page 168 for details.

You can also consider performing a calculation on a related system for which you can more easily obtain convergence—for example, one that is ionized—and then use the results of this system as an initial guess for the system of interest. When you restart a calculation with an input file generated during a previous run, the wave function from the earlier run is read from the **guess** section and used as an initial guess. The **guess** section is described in Section 9.10 on page 283. Jaguar can read in an initial guess in one basis set and transform it to the basis set requested for the calculation (unless either basis set uses effective core potentials).

If you suspect that the initial guess is leading to the wrong state, you can run a calculation with the initial guess only, by choosing Initial Guess Only from the Jaguar submenu of the Applications menu, or setting **igonly**=1. You can then examine the orbitals in the output. If you also generate orbitals surfaces (in the Properties tab—see Section 3.11 on page 75), you can examine the orbitals in Maestro to see if they are what you expect. If the occupied orbitals are not correct, you can swap them by including an **orbman** section in the input file—see Section 9.15 on page 286 for more information. When you swap orbitals, you might also want to fix the orbital populations in each symmetry, by selecting Fixed symmetry populations in the SCF tab, or setting **ipopsym**=1. By default, populations in each symmetry are allowed to vary, so that if the default initial guess is incorrect, there is a chance of converging to the correct state. Of course, you must use symmetry[4] for this strategy to be effective.

7.1.2 Initial Guess for Molecules Containing Transition Metals

For transition-metal-containing systems, particularly organometallics, you can often obtain superior results by improving the initial guess wave function. Jaguar automatically generates high-quality initial guesses for transition-metal-containing compounds; if you supply the program with information about the charges and spins of the "fragments" in the compounds, it uses that information when generating the guess. Here, a fragment is defined as either a collection of one or more transition metals that are bonded together, or one or more non-transition-metal atoms bonded together. Put another way, each fragment is simply a group of atoms that would be bonded together even if all bonds between transition metal atoms and non-transition-metal atoms were broken. Typically, the system is broken into ligand fragments and transition metal fragments, or adsorbate fragments and cluster fragments. For example, for ferrocene, the iron atom is one fragment, and the two cyclopentadienyl ligands are two additional fragments.

4. Keyword **isymm** = 8 in the **gen** section.

To supply Jaguar with information on charges and spins for its high-quality initial guess for a transition-metal-containing system, you need to edit the input file, either from the Edit Job dialog box (which you open by choosing Edit from the Settings button) or from a terminal window. First, add the following lines to the bottom of the input file:

```
&atomic
atom   formal   multip
&
```

(The exact number of spaces between words does not matter.)

Fill in information for each fragment under the headings `atom`, `formal`, and `multip`. You should add a single line for each fragment with a formal charge or a non-singlet spin multiplicity. The first entry in the line (under the heading `atom`) should be the atom label of *any* atom in the fragment. The next entry (under the heading `formal`, and separated from the first entry by one or more spaces) should be the formal charge of the entire fragment. The third entry (under the heading `multip`) should be the spin multiplicity of the fragment. If C1 is in one ring of ferrocene and C6 is in the other ring, then the following **atomic** section could be used to help generate the initial guess:

```
atom   formal   multip
Fe      +2       1
C1      -1       1
C6      -1       1
&
```

Fragments with no formal charge and singlet spin (water, for example) do not need to be listed in the **atomic** section, because Jaguar assumes a default formal charge of 0 and multiplicity of 1 for each fragment. Note, however, that any charge or spin multiplicity settings in the **atomic** section must be compatible with any settings for overall charge and spin specified by the **molchg** and **multip** keywords in the **gen** section. For more information about the **atomic** section, see Section 9.8 on page 273.

After saving the input file with the **atomic** section, you can run it in Jaguar in the usual manner. You do not need to set **iguess**, because Jaguar will choose the most appropriate guess for the system under study.

If you have an antiferromagnetic system, the standard transition-metal initial guesses do not work. For an antiferromagnetic system containing two metal atoms that are not bonded, you can use a **2spin** column in the **atomic** section to set up the initial guess. When the metals are within bonding distance, or when there are more than two metals, you should assign the metal atoms to separate fragments using the **frag** column of the **atomic** section. Finally, add **formal** and **2spin** values in the **atomic** section.

Transition-metal systems can have multiple states based on different occupations of the d orbitals. If this is the case, the initial guess routine prints the possible states, and by default continues with the first state. However, this state might not be the lowest state. You should run calculations on all the possible states in turn to locate the true ground state. You can select states by setting the **istate** keyword in the **gen** section to the index of the state listed in the output from `hfig`. An example of this output is given in Section 6.2 on page 132. If you want to examine the coefficients of the MOs to see which state is which, set **ip105**=7.

7.2 SCF Convergence

Generally, Hartree-Fock wave functions for simple organic molecules converge in fewer than 10 iterations, while complex calculations involving higher-level methods or open shells may take a few extra iterations. Molecules which include transition metals generally converge more slowly, however. Make sure your job has really converged and did not simply end because it reached the maximum number of SCF iterations, a number set in the SCF tab.

If a job gives poor SCF convergence, you can try either modifying the convergence methods used or improving the initial guess. To modify the convergence methods, try any or all of the following settings:

- Set `nofail=1` in the **gen** section. The setting automatically applies one or more emergency measures when bad SCF convergence is detected. One of the measures involves an automatic restart with a different, usually smaller basis set. If the SCF equations successfully converge with that basis set, then the calculation automatically restarts with the original basis set and uses the converged wave function as a guess. You can specify the basis set that is used for the automatic restart with the keyword `restart_basis`. Its default value is `lacvp`.
- Change the Accuracy level setting in the SCF tab to Ultrafine. This corresponds to the keyword setting **iacc**=1, which causes the job to use denser pseudospectral grids and tighter cutoffs, and generally increases computational costs by a factor of two to three. Note, however, that as of Jaguar v7.0 this action is no longer necessarily because if Jaguar detects that the SCF is not converging, it automatically sets **iacc**=1.
- Select GVB-DIIS from the Convergence scheme option menu in the SCF tab. Generally, DIIS is the better choice, but the GVB-DIIS convergence scheme sometimes leads to convergence when DIIS does not. The GVB-DIIS scheme works by ensuring that each new set of occupied orbitals has maximum overlap with the previous set. The advantage of this is that oscillations due to frequently changing occupations are damped. The disadvantage is that if the initial guess occupies the wrong orbitals, the SCF can converge to an excited state.

- Set the SCF level shift in the SCF tab to 0.5 or 1.0. The higher the setting, the more the energies of the virtual orbitals are increased before diagonalization, and the more the mixing of the real and virtual orbitals is reduced. High SCF level shifts can slow convergence by several iterations, but can often help otherwise intractable cases to converge. Because jobs with SCF level shifts are slightly more likely to converge to excited states, you may also want to restart these jobs without any SCF level shift. As of Jaguar 7.0, a level shift of 1.0 is used automatically if Jaguar detects that the SCF is not converging.
- If the calculation is a DFT job, use finer DFT grids. You can adjust this setting from the Grid density option menu in the Theory tab. This setting also increases the computational cost.
- As a last resort, try setting **iacscf** to 1 or 4 (see Table 9.29 on page 246 for descriptions of these settings). This keyword can help when pseudospectral error is particularly high, but it works by removing more eigenvectors from the overlap matrix. This effectively reduces the size of the basis set, which means that if you want to compare energies for two different molecules, you need to use the same value for **iacscf**. You might need to increase the setting of **maxit** to 100 or more when using this keyword.

7.3 Geometry Optimization

If you have built a structure in Maestro, or suspect that the structure is not very close to the optimized geometry, it can be useful to perform a geometry cleanup. To clean up a geometry in Maestro using molecular mechanics with a universal force field (UFF, available for all elements), click the Geometry Cleanup button on the Build toolbar.

For flexible molecules, which might adopt one of several possible conformations, you might consider performing a conformational search with MacroModel. See Chapter 8 of the *MacroModel User Manual* for more information.

If you are performing a geometry optimization and are not starting from a high-quality initial molecular structure, you might want to do a “quick and dirty” calculation to obtain a somewhat better geometry, then perform a more accurate calculation by starting with the results you have generated already. For example, if you wanted to perform an LMP2 geometry optimization, you could start by performing a Hartree-Fock geometry optimization, then restart the calculation using the HF results in an LMP2 geometry optimization. See Section 7.5 for a description of restarting calculations and incorporating previous results in a later run.

Whenever you are doing a geometry optimization, make sure that you really do obtain a converged structure; the run ends before converging if you reach the maximum number of iterations allowed (as set in the Optimization tab). If it did not reach convergence, you can restart the run, as described in Section 7.5. The quality of the convergence is analyzed and printed by default (see Section 4.1.1 on page 92). If you want to automatically check for convergence, you can set the **check_min** keyword, and choose from the related options for convergence checking. At the end of the optimization, a procedure for perturbing the structure is performed, either by following negative eigenvalues of the analytical Hessian, or by perturbing along various torsional coordinates and reoptimizing. This feature is intended for geometry optimizations; it is not supported with dynamic constraints or for scans or IRC calculations.

Geometry optimizations can fail to converge properly due to pseudospectral noise. Jaguar provides a means of alleviating this problem by switching to analytic integral evaluation near convergence. The switch is done when convergence reaches a threshold defined by a multiple of the final convergence threshold. To use this approach, you can specify the multiple with the **nops_opt_switch** keyword, for which a value of 10.0 is appropriate. This approach is not compatible with the use of `check_min` or with IRC calculations.

To assist with failed convergences in geometry optimizations, you can use the `nofail=1` setting. This setting assumes that the geometry corresponding to the minimum energy traversed in the course of the geometry optimization is an acceptable solution to the minimization problem. Note that this geometry and energy pair is not necessarily the geometry and energy pair from the final geometry iteration step. In such cases, the lowest energy and the corresponding geometry are returned as the final result, a warning is printed, and the calculation exits successfully. If, however, the minimum is found, then `nofail=1` does nothing.

The `nofail=1` option is often a convenient practical compromise for various workflows that perform geometry optimizations as intermediate calculations. Normally, if one such geometry optimization fails, the whole workflow fails because it depends on the result generated in the geometry optimization and its successful exit status. The `nofail=1` option makes the workflow that suffers from geometry optimization problems finish successfully, albeit with a possibly lower quality of the final result. In our validation tests with numerous practical examples we have found that the geometry corresponding to the true energy minimum (obtained in a separate calculation) was only slightly different from the geometry returned in a failed geometry optimization that was performed with the `nofail=1` option. We have also noticed that this option greatly improves the robustness of geometry optimization-dependent workflows, often at a negligible loss of accuracy. For this reason, for example, our Jaguar pK_a workflow uses the `nofail=1` setting by default. Of course, you must be careful when using the `nofail=1` setting if a true stationary point on the potential energy surface is required (which is often the case for subsequent vibrational frequency calculations)

Sometimes it is important to prevent a chemical reaction from occurring during a geometry optimization, for example stopping a hydrogen migration. You can do this with the **stop_rxn** keyword, which turns on checking of the connectivity after each geometry optimization step, and restarts the optimization with repulsive potentials or harmonic restraints to prevent the change in bonding.

7.4 Transition-State Searches

Searching for a transition state is one of the more difficult tasks for molecular modeling software. In contrast to geometry optimization, in which moving downhill in energy to the nearest minimum does not require that any particular path be followed, in transition-state searching it is essential that the correct reaction coordinate be followed uphill to the desired transition state. Over the years we have accumulated experience with transition-state searching to make the following recommendations.

- Use the QST search method, in preference to LST or the "standard" method. The QST method requires the most user effort because it requires three input structures, but it is the most likely to find the desired transition-state structure because these three structures allow the optimizer to make the best choice of Hessian eigenvector to follow.
- Prepare the three QST input structures carefully. The extra time spent in preparing high-quality input will be more than compensated by reduced calculation time.

  Start by building your best guess at the TS structure. Use all of your chemical intuition about how molecular structures change as they undergo reactions. Not only do bonds lengthen and shorten, but bond angles also change as atoms change from planar hybridization to pyramidal or vice versa, and torsional motion occurs to avoid steric clashes or to improve orbital overlap. Make all of the changes that you think will distinguish the TS structure from that of the reactant or product. Another option is to perform a geometry optimization in which you constrain those coordinates whose values you feel that you can estimate fairly well, and let everything else relax. The final structure can be used as your guess at the transition state.

  Once you have the guess TS structure, duplicate it twice in the Project Table. Modify the duplicates to form the reactant and product structures. Do not optimize these structures. In order for the QST interpolation scheme to work well, it is essential not to use optimized product or reactant structures, because those structures lie too far away from the transition state structure to provide information about the reaction coordinate. Instead, try to build reactant and product structures that are part of the way downhill from the TS structure. When you modify bond lengths to create the reactant and product structures, the adjustments should be about 0.1 angstrom. When you change bond angles, the adjustments should be about 5 degrees. Torsional changes should be about 10-15 degrees.

- List any bonds that are created or broken in a **connect** section in the input file. This section ensures that the specified coordinates are used by the optimizer, which can greatly reduce the number of steps to find the transition state. Specifying these bonds is critical when bonds are broken or formed. Without this section, bonds that are too far outside the normal range are not generated as internal coordinates, and therefore are not used to guide the search.
- In the Optimization tab of the Jaguar panel, choose the Quantum Mechanical option from the Initial Hessian option menu, or set **inhess**=1 in the **gen** section of the input file. This is an expensive option, particularly for very large structures, but it also provides the best possible initial Hessian. All of the other initial Hessian algorithms are designed for optimizations, not transition state searches, and in particular the Hessians produced by these algorithms are always positive definite (no negative eigenvalues).
- Since TS searches are often slow to converge and can sometimes fail to find the desired TS, a two-stage approach can be employed. Use a relatively small basis set and cheap level of theory, like B3LYP/6-31G*, to get a transition state or close to it. Then switch to a more accurate method like M06-2X/cc-pVTZ(-f) for the final refinement. If the transition state found by the cheap method is very different from your starting guess structure, then it might be a good idea to build new product and reactant structures based on the new transition state structure.
- For atom-transfer reactions, it is usually better to perform a simple relaxed geometry scan rather than a TS search. This is because the reaction coordinate in this situation is completely dominated by the distance coordinate involving the atom being transferred. If your scan increment is small enough, you can accurately locate the transition state.

7.5 Restarting Jobs and Using Previous Results

Sometimes, you may find it useful to restart a job, either because you want to refine the results and do not want to start from the beginning of the calculation, because you want to alter the calculation slightly but want to use an initial guess or geometry from the previous run, or because you encountered some sort of problem that prevented the job from finishing. New input files, which are also called restart files, generated during each job can be used to restart the jobs. These files are automatically written to your local job directory at the end of a run; if the run did not finish, you can usually find the new input file by following the directions at the end of this section.

The restart file contains all the information needed for a new run, incorporating the results from the first run. This file contains the same job settings you made for the original input file for the job, but also contains the results of the job—the final wave function, the final geometry, and the like. Thus, if you want to restart the calculation with the wave function and other data already calculated, you can just read in the new input file. The file name is *jobname*`.**.in`, where the asterisks represent a two-digit number. This number is 01 if the name of the input file for the job from which it was generated is not in this form, and is otherwise set to the number after that assigned to the current input file. These files overwrite any other existing files of the same name.

As an example, if you run the job h2o, the restart file generated during the run is called `h2o.01.in`. You could then read this file, as described in Section 2.5 on page 20, and use it to continue on with the calculation, possibly after making some changes to the calculation requested. The new input file generated during this second run would be called `h2o.02.in`.

If you want to start a new job where the previous job left off, you need only read the new input file in, then make any changes you think are necessary—for example, you could change the SCF energy convergence criterion in the SCF tab. Similarly, if you want to perform an additional calculation once a geometry has been optimized, you can read in the restart file as input for the second job and make any necessary changes to it, such as selecting a DFT calculation instead of Hartree-Fock. Section 2.5 on page 20 contains information on reading input files in the GUI. See Chapter 9 if you would like more information on input files.

Note that if you restart a run, you may not get exactly the same results as you would if you had simply performed a longer run in the first place, even if the calculation type is the same. The methods used in Jaguar sometimes use data from previous iterations, if this information is available, but the data may not be stored in the new input file. For example, the DIIS convergence scheme uses Fock matrices from all previous iterations for the run, and Fock matrices are not stored in new input files. However, calculations should ultimately converge to the same answer within a standard margin of error whether they are restarted or not.

If your run aborted or was killed before completion, and you want to restart the calculation or start another calculation where that one left off, you can look for a file called `restart.in`. The file is located in a subdirectory whose name is the same as the job's, and which is found within the temp directory for the job, which was listed in the Start panel.

By default, the `restart.in` file is written out at the end of the Jaguar programs for calculating the initial guess, performing the SCF iterations, and calculating a new geometry for geometry optimizations, as well as at the end of each SCF iteration. (To turn off `restart.in` file generation, the input file output keywords **ip151** and/or **ip152** in the **gen** section should be set to 0.) The `restart.in` file is overwritten each time, so that the final version is written either at the end of the run or just prior to any problems encountered.

7.6 Conformational Searches

Jaguar and MacroModel can be combined into a very powerful tool for finding the preferred conformations of molecules. MacroModel is used to perform the search through conformational space to identify the lowest-energy conformations. Jaguar is then used to refine the geometries of the lowest-energy conformers. Redundant conformers can then be removed from the data set.

When performing conformational searching of ligands, we recommend either the mixed torsional sampling/low-mode algorithm, or the ligand torsional search method (ConfGen). The former algorithm combines a random search of torsional space with an extensive following of low-mode eigenvectors on the potential energy surface to ensure good sampling of low-energy torsions. The ligand torsion search method does not use any random sampling and therefore avoids the problem of generating the same conformers repeatedly. This method analyzes the molecular structure to identify torsions which might have associated minima, and it generates all potential minimum-energy structures systematically. However, the quality of the conformers it produces is not high, and more work may be needed to correct the structures. See Chapter 8 of the *MacroModel User Manual* and the *ConfGen User Manual* for more information about these conformational search methods.

By default, each conformer that is found by the search algorithm is minimized using the selected force field. Some minimizations may not converge within the limit of minimization steps. This is not necessarily a problem, but if you want to ensure that you only use fully minimized structures as input to Jaguar, there are two ways to do it. The first is to simply increase the limit on the number of minimization steps while performing the conformational search. However, this can be expensive when large numbers of conformers are generated, because you will probably refine only a subset of these with Jaguar. If you expect the number of conformers to be large, the second approach is to decrease the limit on the number of minimization steps while performing the search so that it finishes quickly. After the search has finished and the results have been incorporated into the Project Table, you can select a subset of them and minimize them all using a MacroModel multiple minimization. At this point you would use a high limit on the number of minimization steps to ensure that all minimizations converge.

After running the conformational search from Maestro, the results are incorporated into the Project Table, ranked so that the lowest-energy structures are uppermost. At this point you should examine the results and select an appropriate subset of conformers for refinement with Jaguar. Then choose Optimization from the Jaguar submenu of the Applications menu. At the top of the Jaguar panel, choose Selected entries from the Use structures from option menu. Jaguar's default calculation settings, B3LYP/6-31G**, are appropriate for refining the structures of conformers, so at this point you can simply click the Start button and launch your job. If the host or queue on which you want to run the job has as many CPUs available as there are conformers and you have sufficient license tokens, you can set the number of CPUs to the number of conformers. All conformers are optimized at the same time, on separate processors, for maximum efficiency.

When the Jaguar optimization results are incorporated into the project, you can use the MacroModel redundant conformer elimination tool to remove any duplicate structures, in case some of the conformers that appeared to be different at the MM level were refined to the same structure at the QM level. See Chapter 15 of the *MacroModel User Manual* for more information on this tool.

Tutorial example of conformational searches can be found in Chapter 4 and Section 6.1 of the *Jaguar Quick Start Guide*.

7.7 Timings for Typical Jobs

The tables below shows some timings for various job tasks at some common levels of theory. The timings are for a single core of a 2.27 GHz processor. A substantial speedup may be obtained by using multiple processors. Vibrational frequencies are the most expensive properties to calculate, and LMP2 is a more expensive level of theory than B3LYP. Calculations with functionals that include the kinetic energy density, like the M06 set, are somewhat slower than B3LYP, while calculations with functionals like PBE and BLYP that do not include any Hartree-Fock exchange are somewhat faster than B3LYP.

The time for an SCF (or a single-point energy calculation) depends on the number of cycles required to meet the convergence criteria. Similarly, the time required to optimize a molecular geometry depends on the number of steps required for convergence of the energy, gradients, and displacements. The number of steps, in turn, depends on the nature of the molecular and electronic structure. A rigid molecule, like an aromatic ring, will generally require fewer steps to optimize than a conformationally flexible molecule. Note that each step of a geometry optimization also requires an evaluation of the energy, so to estimate the time for 20 steps of a geometry optimization on a molecule containing 50 atoms, using B3LYP/6-31G**, you would need approximately 20 * (4 m + 2 m) = 120 minutes.

*Table 7.1. Timings for various tasks run with B3LYP/6-31G**.*

Atoms	Basis size	Energy	Geopt per step	NMR	Vibrational freqs	TDDFT per root	ESP chg + dipole
15	145	<1 m	<1 m	<1 m	16 m	<1 m	<1 m
49	485	4 m	2 m	4 m	2 h	2 m	<1 m
77	775	21 m	5 m	14 m	9 h	9 m	<1 m
104	1060	26 m	8 m	29 m	25 h	7 m	<1 m

Table 7.2. Timings for various tasks run with B3LYP/cc-pVTZ(-f).

Atoms	Basis size	Energy	Geopt per step	NMR	Vibrational freqs	TDDFT per root	ESP chg + dipole
15	233	1 m	<1 m	1 m	2 h	<1 m	<1 m
49	777	14 m	4 m	13 m	12 h	6 m	<1 m
77	1239	1 h	11 m	43 m	52 h	50 m	1 m
104	1692	2 h	20 m	2 h	144 h	1 h	1 m

Table 7.3. Timings for various tasks run with LMP2/cc-pVTZ(-f).

Atoms	Basis size	Energy	Geopt per step	ESP chg + dipole
15	233	1 m	2 m	1 m
49	777	1 h	3 h	2 h
77	1239	5 h	days	20 h

7.8 Noncovalent Interactions

The existence of interactions between atoms can be assessed by examining the electron density, and in particular the points at which the gradient of the electron density is zero, known as “critical points”. There are four types of critical points. The first is where the density is a maximum in all three orthogonal spatial directions (x, y, and z), and marks the position of an atomic nucleus, and is called an “atom critical point”. The second occurs along a bond, where the electron density is a minimum at some point along a line between two atoms (the “bond path”), but is a maximum in both directions perpendicular to the bond. This is called a “bond critical point”. The third occurs in a ring, such as benzene. At some point in the middle of the ring there is a point at which the electron density is a minimum in the directions that are in the plane of the ring, but is a maximum in the direction perpendicular to the ring. This is called a “ring critical point”. The fourth occurs in a cage structure, like cubane, where there is a minimum in the electron density in all three spatial directions. It is called a “cage critical point”. These descriptions are somewhat approximate, because the bond path might not be quite linear and the rings might not be planar, but the principle holds.

These critical points exist for all types of interactions. Every covalent bond has a bond critical point along the bond path that connects the two atoms in the bond, for example. The analysis is perfectly general, however: it is not limited to conventional bonds, which we represent by Lewis structures. Bond critical points exist for covalent bonds, ionic bonds, hydrogen bonds, pi-pi interactions, van der Waals complexes. The generality is what makes the method so powerful: it can identify all kinds of bonding interactions, based solely on the electron density.

Noncovalent interaction (NCI) plots can help resolve questions about whether bonding interactions truly exist between atoms that are not considered to be covalently bound to each other. Hydrogen bonds, for example, are known to be dependent upon both the H-acceptor atom distance and the donor-H-acceptor angle. When these parameters are far from ideal, one may question whether a bonding interaction truly exists. In other cases, a pi-pi interaction between aromatic rings may be controversial because the rings are tilted with respect to one another, or a halogen-oxygen bond may be disputed because the X-O distance is so large. When a bonding interaction truly exists between atoms, a bond critical point forms between them. This interaction is represented in an NCI plot as a small isosurface of low density gradient, colored to show that the interaction is favorable. The bond critical points are identified when you request NCI surfaces, and are marked with dummy atoms. The bond path is represented as zero-order bonds to the dummy atom.

Ring critical points form inside any closed loop of bond paths. In symmetrical rings like benzene, the ring critical point is in the center of the ring. But if one bond in the ring is weaker than the others, as in the case of an intramolecular hydrogen bond, the ring critical point is off-center and closer to the critical point in the weak bond. The weaker this bond is, the closer the ring and bond critical points are to each other. Sometimes a ring critical point and a bond critical point are close enough together that they appear in the same isosurface. In this case, part of the isosurface is colored to show the favorable interaction indicated by the bond critical point, while the rest of the isosurface is colored to show the unfavorable interaction indicated by the ring critical point.

To generate an NCI plot using the Jaguar panel in Maestro, select Surfaces in the Properties table of the Properties tab, then select Noncovalent interactions in the Surfaces section. When the job is incorporated, the reduced density gradient surface is displayed by default, but it is not colored by the interaction strength. You can do this as follows:

1. Open the Manage Surfaces panel.
2. Select the reduced density gradient in the table.
3. Click Display Options.
4. Select Map values from volume data.
5. Choose the strength surface in the list.
6. Choose the color ramp (the default is Red_White_Blue).
7. Click OK.

The bond critical points and paths are also displayed by default. You can display and undisplay them with the Maestro command `[un]display (atom.mtype Z0).`)

See Section 3.4 of the *Jaguar Quick Start Guide* for a tutorial example on noncovalent interactions and the density Laplacian.

7.9 Generating Input Files for GAUSSIAN

We recognize that some Jaguar users also use GAUSSIAN for calculations. Therefore, Jaguar can generate or read GAUSSIAN input files.

You can use the GUI as a convenient tool to create GAUSSIAN input files. The output file that is produced from the Jaguar run and whose name ends in `.gau` can be used as a GAUSSIAN input file. The `.gau` file requests an HF or ROHF (restricted open-shell Hartree-Fock) calculation, whichever is appropriate for the number of electrons in the system, unless you choose to specify another method.

To create a `.gau` file, select the Gaussian input file (.gau) option in the Output tab. If you are just creating a GAUSSIAN input file and you do not want to use Jaguar to generate a converged wave function, you can save some time by choosing Initial Guess Only from the Jaguar submenu of the Applications menu, or using the Edit Job dialog box to add the keyword setting **igonly**=1 (initial guess only) to the **gen** section of the input file.

The information in the `.gau` file depends on the information you have provided. The file always contains a molecular geometry (in Cartesian coordinates and angstroms); instructions for how to input geometries are available in Section 2.4 on page 13. The file also specifies the molecular charge and the spin multiplicity of the molecule. If you want either of these values to be non-zero, you can make the appropriate settings in the Molecule tab. You can also set the name of the basis set you want to provide in the `.gau` file (for example, STO-3G) using the Molecule tab. (The default basis set choice is 6-31G**.)

To actually generate the `.gau` file, you need to run the Jaguar job you have just specified. See Section 2.9 on page 25 for information on running jobs.

7.9.1 Other Jaguar Options for the `.gau` File

You can use a Jaguar input file to run a Jaguar job which generates a `.gau` file. See Chapter 9 for a description of input files. Selecting the Gaussian input file (.gau) output option described above corresponds to setting the output keyword **ip160** to 2 in the **gen** section of the input file.

You can create or edit Jaguar input files by hand, making keyword settings corresponding to all of the relevant options described above; see Chapter 9 for details. If you want, you can make some of the desired settings in the GUI, use the Jaguar Write dialog box to save a Jaguar input file, and edit it by hand later to set other keywords.

You can generate additional information for the `.gau` file by setting the output keyword **ip160** in the **gen** section of the input file to 3, 4, or 5. Setting this keyword to 3 lets you provide an initial guess within the `.gau` file. Setting it to 5 allows you to explicitly provide the basis set itself, rather than just the basis set name, within the `.gau` file. This option is useful for specifying basis sets which are included in Jaguar but not in GAUSSIAN. Setting **ip160** to 4 allows you to include both the initial guess and the basis set in the `.gau` file.

7.9.2 Writing Orbitals for GAUSSIAN

You can write orbitals from Jaguar in the format used by GAUSSIAN (for its “guess=cards” option) by choosing to print the appropriate orbitals from the Output tab, which is described in Section 3.13 on page 85. You must choose the f19.15 or f8.5 format from the Format option menu.

Theory

This chapter contains a description of some of the theory behind the methods used in Jaguar. Section 8.1 describes the pseudospectral method itself. Section 8.2 describes LMP2 calculations and how the pseudospectral method improves computational scaling and efficiency for these methods. Section 8.3 contains a brief description of density functional theory. Chapter 3 includes information about performing Jaguar calculations using the techniques described here.

8.1 The Pseudospectral Method

Like conventional ab initio electronic structure codes, Jaguar solves the Schrödinger equation iteratively, using self-consistent field methods to calculate the lowest-energy wave function within the space spanned by the selected basis set. For calculations on large molecules, both conventional and pseudospectral techniques must recalculate key integral terms for each SCF iteration, since storage costs for these terms are prohibitive.

Most of the fundamental integrals calculated in the pseudospectral method [1-9] are computed in physical space, on a grid, rather than in the spectral space defined by the basis functions. The pseudospectral method takes the density matrix from the wave function at the beginning of each SCF iteration and the values of the integrals on the grid points and manipulates them to produce the necessary operators on the grid, then assembles the Fock matrix by transforming these components back into spectral space, where the Fock matrix is used in the usual way to generate the wave function for the next iteration.

For medium and large molecules, the additional overhead of the pseudospectral method in computing the transformation between physical and spectral space is vastly outweighed by the advantages of evaluating the integrals in physical space. The matrix needed for the transformation from physical to spectral space [7] can be assembled before the SCF iterations by calculating the least-squares operator $\boldsymbol{Q}$, which is given by the equation

$$\boldsymbol{Q} = \boldsymbol{S}[\boldsymbol{R}^{\dagger}\boldsymbol{w}\boldsymbol{R}]^{-1}\boldsymbol{R}^{\dagger}\boldsymbol{w} \qquad (1)$$

where $\boldsymbol{S}$ is the analytic overlap matrix between the fitting functions and the basis set, $\boldsymbol{R}$ is the matrix of fitting functions evaluated at the grid points, and $\boldsymbol{w}$ is a diagonal matrix of grid weights. The fitting functions used to construct the matrix $\boldsymbol{R}$ include both basis functions and dealiasing functions, which are chosen in order to span the function space represented by the grid more completely than the basis functions alone. The operator $\boldsymbol{Q}$ can be calculated for the relevant basis functions using several different sets of grid points, where each set of points defines a grid type, ranging from coarse to ultrafine.

In practice, not all possible $\boldsymbol{Q}_{ig}$ elements are calculated for each basis function $\boldsymbol{i}$ and each grid point g, because most basis functions drop off sharply enough that they have no significant value on some or most grid points. These functions are classified as short-range functions and are grouped together by atom, while the remaining functions are classified as long-range functions, which are all considered to be in one single group [13].

Since $\boldsymbol{Q}$ does not depend on the wave function itself, it can be fully computed before the SCF procedure. However, since the $\boldsymbol{Q}$ for each grid type contains N_{basis} x N_{grid} elements, where N_{basis} is the number of basis functions and N_{grid} the number of grid points (which is generally larger than N_{basis}), we sometimes reduce memory demands by only computing and storing the N_{basis} x N_{fit} matrix $\boldsymbol{S}[\boldsymbol{R}^{\dagger}\boldsymbol{wR}]^{-1}$ in the program `rwr`, for cases where the $\boldsymbol{Q}$ for that grid type is only needed for one SCF iteration. We then assemble the full $\boldsymbol{Q}$ during the SCF iteration for which it is needed.

After the program `rwr` has generated the $\boldsymbol{Q}$ or $\boldsymbol{S}[\boldsymbol{R}^{\dagger}\boldsymbol{wR}]^{-1}$ matrix, the program `scf` takes the initial orbitals and iteratively modifies them with the pseudospectral method until convergence. This process involves calculating the values of the necessary integrals on the grid points, and actually assembling the Fock matrix from the computed information. The three-center, one-electron pseudospectral integrals on the grid points are defined by

$$A_{klg} = \int \frac{\varphi_k(1)\varphi_l(1)}{r_{1g}} d\mathbf{r}_{1g} \tag{2}$$

where φ_k and φ_l are basis functions and the index g represents a grid point. These integrals are calculated for all combinations of basis functions and grid points not eliminated by cutoffs, and the Fock matrix is assembled from its Coulomb and exchange matrix components J_{ij} and K_{ij}, which are calculated in physical space and transformed back into spectral space by the following equations:

$$J_{ij} = \sum_g Q_{ig}\left[\sum_{kl} A_{klg} D_{kl}\right] R_{jg} \tag{3a}$$

$$K_{ij} = \sum_g Q_{ig}\left[\sum_n A_{jng} \sum_m D_{nm} R_{mg}\right] \tag{3b}$$

where $\boldsymbol{D}$ is the usual spectral space density matrix, R_{jg} is the value of the function j at grid point g, and A_{klg} is given by Equation (2). The grid points used for each SCF iteration are determined by the grid type (coarse, medium, fine, or ultrafine) chosen for that iteration. The number of arithmetic operations involved in the assembly of the matrices $\boldsymbol{J}$ and $\boldsymbol{K}$ in Equation (3a) and Equation (3b) scales formally as N^3, as opposed to the N^4 scaling for the matrix assembly in the conventional spectral space algorithm.

Jaguar actually uses the pseudospectral method described above for the majority of the computationally intensive two-electron integral terms, but calculates the one-electron and some of the largest and most efficiently computed two-electron terms analytically [13]. For the Coulomb matrix elements, we calculate the analytic terms

$$\sum_{kl}(ij|kl)D_{kl}$$

for cases in which i, j, k, and l meet certain cutoff criteria and the two-electron integral $(ij|kl)$ is of the form $(aa|aa)$, $(aa|ab)$, $(aa|bb)$, $(ab|ab)$, or $(aa|bc)$, where *a, b,* and *c* indicate the atom upon which the function is centered. Similar correction terms are computed for the exchange operator, as detailed in ref. 13. The corresponding pseudospectral terms, as defined by Equation (3a) and Equation (3b) for the appropriate choices of *i, j, k,* and *l*, must be subtracted from the pseudospectral $\boldsymbol{J}$ and $\boldsymbol{K}$ elements as well. This combined pseudospectral/analytic approach allows Jaguar to take advantage of the strengths of both methods, since it can largely maintain the pseudospectral method speedups for a particular grid, and can also use a coarser grid than a purely numerical calculation would allow.

8.2 Pseudospectral Local MP2 Techniques

Second order Møller-Plesset perturbation theory (MP2) is perhaps the most widely used ab initio electron correlation methodology, recovering a large fraction of the correlation energy at a relatively low computational cost. The method greatly improves Hartree-Fock treatments of properties such as transition states, dispersion interactions, hydrogen bonding, and conformational energies. However, the scaling of conventional MP2 algorithms with system size is formally nN^4, where N is the number of basis functions and n the number of occupied orbitals, due to the necessity of carrying out a four index transformation from atomic basis functions to molecular orbitals. In principle, it is possible to reduce this scaling by using integral cutoffs, as for Hartree-Fock calculations. However, the reduction is noticeably less effective in MP2, particularly for the large, correlation-consistent basis sets that are required for accurate correlation effects on observable quantities. Thus, MP2 techniques have traditionally been used primarily for small molecules.

Several years ago, Pulay and coworkers [89, 90] formulated a version of MP2 in which the occupied orbitals are first localized (e.g., via Boys localization [92]) and the virtual space correlating such orbitals are then truncated to a local space, built from the atomic basis functions on the local atomic centers orthogonalized to the occupied space. Another critical advantage of LMP2 is that one can very precisely control which region of the molecule is correlated, reducing CPU costs enormously. The method has been shown to yield an accuracy for relative energies that is, if anything, superior to conventional MP2, due to elimination of basis set superposition error [91]. However, localized MP2 implementations in conventional electronic structure codes have not yet led to substantial reductions in CPU time, since the first few steps of the necessary four-index transformation are unaffected by localization of the occupied orbitals, and the localized orbitals have tails that extend throughout the molecule.

We have carried out extensive tests demonstrating the accuracy and computational efficiency of the pseudospectral implementation of LMP2, as detailed in ref. 16. In the pseudospectral approach, we assemble two-electron integrals over molecular orbitals directly and are thus able to fully profit from the huge reduction in the size of the virtual space in Pulay's theory. Formally, the PS implementation of LMP2 scales as nN^3; however, various types of cutoffs and multigrid procedures can reduce this to ~N^2. In fact, for calculations involving both the 6-31G** and Dunning cc-pVTZ basis sets, we find a scaling ~$N^{2.7}$ with system size.

The physical idea behind the LMP2 method is that if the molecular orbitals are transformed so that they are localized on bonds or electron pairs, correlation among the occupied pairs can be described by the local orbital pairs and their respective local pair virtual spaces defined from the atomic orbitals on the relevant atom or pair of atoms. The localized orbitals can be generated by any unitary transformation of the canonical orbitals. For LMP2, we use Boys-localized [92] orbitals, for which the term $\Sigma_{ij}\left|\langle\varphi_i|r|\varphi_i\rangle - \langle\varphi_j|r|\varphi_j\rangle\right|^2$ is maximized. The local virtual space for each atom is defined by orthogonalizing its atomic basis functions against the localized molecular orbitals. The correlating orbitals included in the local virtual space are thus mostly near the atom itself, but because of the orthogonalization procedure, they are not particularly well localized.

The Jaguar LMP2 program uses Pulay's method [89, 90, 91] to expand the first order wave function correction $\Psi^{(1)}$ as a linear combination of determinants formed by exciting electrons from localized orbitals i and j to local virtual space correlation orbitals p and q:

$$\Psi^{(1)} = \sum_{i \geq j}\sum_{pq} C_{ij}^{pq}\Psi_{ij}^{pq} \tag{4}$$

For local MP2, we must iteratively solve the following equation, which has been derived in detail by Pulay and Sæbo, for the coefficients C_{ij}^{pq} :

$$\boldsymbol{T}_{ij}^{(2)} = \boldsymbol{K}_{ij} + \boldsymbol{F}\boldsymbol{C}_{ij}\boldsymbol{S} + \boldsymbol{S}\boldsymbol{C}_{ij}\boldsymbol{F} - \boldsymbol{S}\left(\sum\left[\boldsymbol{F}_{ik}\boldsymbol{C}_{kj} + \boldsymbol{F}_{kj}\boldsymbol{C}_{ik}\right]\right)\boldsymbol{S} = 0 \tag{5}$$

Here ***F*** is the Fock matrix, ***S*** is the overlap matrix, and ***T*** is the residual matrix defined by this equation. The exchange matrix $\boldsymbol{K}_{ij}$ is restricted to the dimensions of the virtual space corresponding to the occupied localized molecular orbitals *i* and *j*. The simplest updating scheme for the coefficients is to obtain updated coefficients $\boldsymbol{C}_{ij}{}'$ iteratively from the equation:

$$(C_{ij}^{pq})' = C_{ij}^{pq} + \frac{T_{ij}^{pq}}{\varepsilon_i + \varepsilon_j - \varepsilon_p{}^* - \varepsilon_q{}^*} \tag{6}$$

where ε_i and ε_j are the matrix elements F_{ii} and F_{jj} in the localized molecular orbital basis and ε_p and ε_q are the eigenvalues of the Fock matrix in the local virtual basis.

From the $\boldsymbol{C}_{ij}$ coefficients and the exchange matrices $\boldsymbol{K}_{ij}$, Jaguar computes the second order energy correction $E^{(2)}$ from the equations:

$$E^{(2)} = \sum_{i \geq i} \langle \boldsymbol{K}_{ij}\tilde{\boldsymbol{C}}_{ji} \rangle \tag{7a}$$

$$\tilde{\boldsymbol{C}}_{ji} = (1 + \delta_{ij})^{-1}(4\boldsymbol{C}_{ij} - 2\boldsymbol{C}_{ji}) \tag{7b}$$

where the bracket in Equation (7a) denotes a trace and δ_{ij} is 1 if $i = j$ and 0 otherwise. Computing the exchange matrix elements for Equation (7a) is approximately 80% of the work for an energy correction computation, while generating the C_{ij} coefficients comprises about 20% of the work.

Jaguar performs localized MP2 calculations using pseudospectral methods, evaluating integrals over grid points in physical space in a manner similar to that described for HF calculations in Section 8.1 on page 183. The two-electron exchange integrals needed for Equation (7a) are evaluated over grid points *g* as follows:

$$K_{ij}^{pq} = \sum_{g} Q_{ig} A_{jqg} R_{pg} \tag{8}$$

where Q_{ig} is the least squares fitting operator for molecular orbital *i* on grid point *g*, R_{pg} is the physical space representation of virtual orbital *p*, and A_{jqg} is the three-center, one-electron integral over the occupied molecular orbital *j* and the local virtual orbital *q*. The last term is related to the three-center, one-electron integrals in atomic orbital space, A_{klg}, described in Equation (2), by

$$A_{jqg} = \sum_{kl} c_{kj} c_{lq} A_{klg} \tag{9}$$

The summation is performed in two steps, first summing over k to form intermediates A_{jlg},

$$A_{jlg} = \sum_{k} c_{kj} A_{klg} , \tag{10}$$

then summing over l to yield the integrals in molecular orbital space

$$A_{jqg} = \sum_{l} c_{lq} A_{jlg} . \tag{11}$$

Jaguar's local MP2 module also includes analytical corrections similar to those described earlier for Hartree-Fock calculations, and a length scales algorithm, both of which are explained in reference 13.

8.3 Density Functional Theory

Density functional theory (DFT) is based on the Hohenberg-Kohn theorem [178], which states that the exact energy of a system can be expressed as a functional depending only on the electron density. In the Kohn-Sham implementation of DFT [179], this density is expressed in terms of Kohn-Sham orbitals $\{\psi_i\}$:

$$\rho(\mathbf{r}) = 2\sum_{i}^{\text{occ.}} \left|\psi_i(\mathbf{r})\right|^2 \tag{12}$$

similarly to the density expression used for Hartree-Fock SCF calculations. For simplicity, we consider only closed shell systems in this overview of the method.

The Kohn-Sham orbitals are expressed as a linear combination of basis functions $\chi_i(\mathbf{r})$, and the coefficients for this expansion are solved iteratively using a self-consistent field method, as for Hartree-Fock. However, DFT includes exchange and/or correlation density functionals within the Fock matrix used for the SCF procedure. For DFT calculations, the Hartree-Fock exchange term K_{ij} in the Fock matrix is replaced by the exchange-correlation potential matrix elements V_{ij}^{xc} :

$$V_{ij}^{xc} = \int d\mathbf{r}\left(\left(\frac{\partial f_{xc}[\rho,\nabla\rho]}{\partial\rho}\right)\chi_i(\mathbf{r})\chi_j(\mathbf{r}) + 2\frac{\partial f_{xc}[\rho,\nabla\rho]}{\partial\gamma}\nabla\cdot(\chi_i(\mathbf{r})\chi_j(\mathbf{r}))\right) \tag{13}$$

where $f_{xc}[\rho,\nabla\rho]$ is an exchange-correlation functional and γ is $\sqrt{\nabla\rho\cdot\nabla\rho}$.

The exchange-correlation functional $f_{xc}[\rho,\nabla\rho]$ is usually separated into exchange and correlation functional components that are local or non-local in the density:

$$f_{xc}[\rho,\nabla\rho] = f_x[\rho] + f_{x,NL}[\rho,\nabla\rho] + f_c[\rho] + f_{c,NL}[\rho,\nabla\rho] \tag{14}$$

Under the local density approximation (LDA), the non-local functionals $f_{x,NL}[\rho,\nabla\rho]$ and $f_{c,NL}[\rho,\nabla\rho]$ are ignored; when either or both of these terms are included, the generalized gradient approximation (GGA), also known as the non-local density approximation (NLDA), applies. The local and non-local exchange and correlation functionals available within Jaguar are described in Section 3.3 on page 38 and its references.

The electronic ground state energy E_0 is given by

$$E_0 = 2\sum_i\int d\mathbf{r}\psi_i -\frac{1}{2}\nabla^2\psi_i + \int d\mathbf{r}V_{nuc}(\mathbf{r})\rho(\mathbf{r}) + \frac{1}{2}\int d\mathbf{r}J(\mathbf{r})\rho(\mathbf{r}) + \int d\mathbf{r}f_{xc}[\rho,\nabla\rho] \tag{15}$$

(in Hartree atomic units), where V_{nuc} is the nuclear potential and J is the Coulomb potential. Therefore, for a given exchange-correlation functional, it is possible to solve iteratively for Kohn-Sham orbitals $\psi_i(\mathbf{r})$ and the resulting density ρ to yield a final DFT energy.

A more detailed description of density functional theory can be found in Refs. 180 and 181.

8.3.1 Long-Range-Corrected Density Functionals

Most common density functionals cannot correctly describe the distance dependence of charge-transfer excitation energies since the non-local Hartree-Fock exchange is not 100% included. One way to solve this problem is to preserve the form of common GGAs and hybrid functionals at short range while applying 100% Hartree-Fock exchange at long range. This can be done with a partitioning of the electron-electron Coulomb potential into long- and short-range components [182, 183]:

$$\frac{1}{r_{12}} = \frac{\mathrm{erfc}(\omega r_{12})}{r_{12}} + \frac{\mathrm{erf}(\omega r_{12})}{r_{12}} \tag{16}$$

where erf and erfc are the error function and complementary error function. The first term on the right in Equation (16) is short range and decays to zero on a length scale of $1/\omega$, while the second term is long-range.

The DFT exchange-correlation energy for a hybrid functional is normally expressed as

$$E_{XC} = E_C + E_X + C_{HF}E_X^{HF} \tag{17}$$

where E_C is the correlation energy, E_X is the DFT exchange energy, and E_X^{HF} is the Hartree-Fock exchange energy with C_{HF} the weight of Hartree-Fock exchange. The two components of the range-separated Coulomb operator are weighted separately, to give a generic long-range-corrected (LRC) energy that can be expressed as

$$E_{XC} = E_C + E_X + C_{HF}E_X^{HF} + C_{HF}^{SR}E_X^{HF,SR} + C_{HF}^{LR}E_X^{HF,LR} \tag{18}$$

where *SR* and *LR* indicate that these components of the functional are evaluated using either the short-range (SR) or the long-range (LR) component of the Coulomb operator. The first three terms on the right are the same as in the uncorrected functional.

The short-range Hartree-Fock exchange is replaced, either entirely or in part, with a modified version of the DFT exchange functional in which the error function is applied to truncate it, either by the method of Iikura et al. [64] or that of Henderson et al. [66]. With the introduction of the short-range corrected functional, the DFT exchange-correlation energy can be expressed in general as

$$E_{XC} = E_C + E_X + C_{HF}E_X^{HF} + C_XE_X + C_X^{SR}E_X^{SR} + C_{HF}^{SR}E_X^{HF,SR} + C_{HF}^{LR}E_X^{HF,LR} \tag{19}$$

The coefficients are not independent, since as a consequence of Equation (16),

$$E_X^{HF} = E_X^{HF,SR} + E_X^{HF,LR} \tag{20}$$

Thus, if $C_{HF} = -C_{HF}^{SR}$ then there is no short-range Hartree-Fock exchange and only long-range Hartree-Fock exchange. The same is true for the DFT exchange. This form is useful, however, for writing the functional as the original functional (first three terms) plus a correction term (last four terms).

8.4 The ZORA Hamiltonian

The effect of relativity on the motion of electrons near the nucleus changes the electronic structure. This effect increases with Z, so that for the lower half of the periodic table, inclusion of relativistic effects is necessary to get quantitative accuracy and frequently even qualitative accuracy. Even for the first transition series, relativistic effects on the 4s electron are larger than the 1 kcal/mol “chemical accuracy” threshold.

The most rigorous introduction of relativistic effects is done with the Dirac Hamiltonian. A full implementation of the Hartree-Fock method with the Dirac Hamiltonian (Dirac-Hartree-Fock, or DHF) is 1 to 2 orders of magnitude more time-consuming than the usual nonrelativistic Hartree-Fock method. Consequently, many approximations to this method have been developed, derived in one way or another from the Dirac equation. For an introduction to relativistic effects in quantum chemistry, see the book by Dyall and Fægri [184].

One popular way of including relativistic effects in density functional calculations is through the ZORA (zeroth-order regular approximation) Hamiltonian [185,186]. It is derived by eliminating the so-called "small component" of the Dirac wave function, leaving a two-component Hamiltonian that includes spin-free and spin-dependent operators, including the spin-orbit interaction. The ZORA Hamiltonian can be written as follows:

$$H^{\text{ZORA}} = V + (\sigma \cdot \mathbf{p})\frac{c^2}{2mc^2 - V}(\sigma \cdot \mathbf{p}) \tag{21}$$

where c is the speed of light (approximately 137 in atomic units), and σ is the vector of Pauli matrices,

$$\sigma_x = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, \qquad \sigma_y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix}, \qquad \sigma_z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix}. \tag{22}$$

As these are 2x2 matrices, the wave function is a 2-component vector, with the components representing alpha and beta spin. One can think of writing the spin functions as

$$\alpha = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, \qquad \beta = \begin{bmatrix} 0 \\ 1 \end{bmatrix}. \tag{23}$$

The ZORA Hamiltonian can be rearranged by partitioning the denominator to give the nonrelativistic Hamiltonian with a relativistic correction,

$$H^{\text{ZORA}} = V + T + (\sigma \cdot \mathbf{p})\frac{V}{2m(2mc^2 - V)}(\sigma \cdot \mathbf{p})\,. \tag{24}$$

Finally, the properties of the Pauli matrices can be exploited to partition the relativistic correction into a spin-free and a spin-dependent part,

$$H^{\text{ZORA}} = V + T + \mathbf{p} \cdot \frac{V}{2m(2mc^2 - V)}\mathbf{p} + \sigma \cdot \mathbf{p}\frac{V}{2m(2mc^2 - V)} \times \mathbf{p}\,. \tag{25}$$

If the last, spin-dependent, term is left out, the Hamiltonian is called the scalar ZORA Hamiltonian, spin-free ZORA Hamiltonian, or one-component ZORA Hamiltonian. When the last term is included, the Hamiltonian is called the spin-orbit ZORA Hamiltonian or two-component ZORA Hamiltonian.

The ZORA Hamiltonian has some known issues. If you add a constant to the potential in any Hamiltonian, the energy should shift by the same constant. In ZORA this does not happen because the constant is also added in the denominator. This means that the ZORA Hamiltonian is not invariant to the electric gauge (the choice of the zero of potential, which involves shifting the zero by a constant). To alleviate this problem, the potential in the denominator (or in fact the entire relativistic correction) can be fixed, so that instead of the potential of the system it is a model potential [187]. Thus we write the ZORA Hamiltonian as

$$H^{\mathrm{ZORA}} = V + T + \mathbf{p} \cdot \frac{V^{\mathrm{MP}}}{2m(2mc^2 - V^{\mathrm{MP}})}\mathbf{p} + \sigma \cdot \mathbf{p}\frac{V^{\mathrm{MP}}}{2m(2mc^2 - V^{MP})} \times \mathbf{p}\,. \tag{26}$$

The second issue is what to include in the model potential. The ZORA Hamiltonian was originally derived from the one-electron Dirac equation, so V^{MP} is the bare nuclear potential. For use in many-electron systems, it should be derived instead from the Dirac-Hartree-Fock (DHF) equation or the Dirac-Kohn-Sham (DKS) equation. Derivation from the DHF equation results in a very complicated operator, and some approximation is needed [188]. By neglecting small terms, the potential V^{MP} includes only the nuclear and electronic Coulomb potentials,

$$V^{\mathrm{MP}} = V^{\mathrm{nuc}} + J\,. \tag{27}$$

Derivation from the DKS equation is straightforward, and simply involves adding the electronic Coulomb and exchange-correlation potential to the nuclear potential in V^{MP}.

$$V^{\mathrm{MP}} = V^{\mathrm{nuc}} + J + V^{\mathrm{xc}}\,. \tag{28}$$

The potential depends on the choice of density functional, and for a hybrid functional that includes exact exchange, which is not included in the model potential derived from the DHF equation. The compromise used in Jaguar is to add only the electronic Coulomb potential, as in Equation (27), as this accounts for the bulk of the many-electron effects. The potential is evaluated from the spherically averaged atomic initial guess, and is not changed subsequently.

The third issue is that the eigenvalues of the ZORA Hamiltonian (or Fock matrix) are substantially lower than those from the Dirac Hamiltonian (or DHF matrix). This is due to the approximations made to derive the ZORA Hamiltonian, which does not include all of the lowest-order relativistic effects. However, an a posteriori correction can be made to the eigenvalues and applied to the calculation of the total energy [189], as follows.

The ZORA equation is derived by neglecting part of the normalization of the Dirac wave function. The operator for the normalization (the metric operator) with the lowest order correction included is

$$G = 1 + (\sigma \cdot \mathbf{p}) \frac{c^2}{(2mc^2 - V)^2} (\sigma \cdot \mathbf{p}) \tag{29}$$

If we evaluate the matrix element of this operator for each orbital, and divide the eigenvalue, we get the scaled ZORA eigenvalues,

$$\varepsilon^{\text{scaled ZORA}} = \varepsilon^{\text{ZORA}} / \langle G \rangle \tag{30}$$

For a one-electron system, the scaled ZORA eigenvalues are exactly equal to the Dirac eigenvalues. For a many-electron system, this should still be a good approximation. To apply the scaling in the total energy, note that the energy can be written as

$$E = \sum_i h_{ii} + \frac{1}{2} \sum_{ij} V_{ij} = \sum_i \varepsilon_i - \frac{1}{2} \sum_{ij} V_{ij} \tag{31}$$

because the eigenvalue can be written as

$$\varepsilon_i = h_{ii} + \sum_j V_{ij} \tag{32}$$

The scaled ZORA energy is then calculated by replacing the ZORA eigenvalues in the energy expression with the scaled ZORA eigenvalues.

In the implementation of ZORA, as the potential is in the denominator, it is difficult (if not impossible) to calculate the integrals analytically. In Jaguar (as in other codes), the integrand is tabulated on a grid: the ratio involving the potential is tabulated on the grid, the momentum operators are applied to the basis functions and the result is tabulated on the grid. The integrals are then evaluated by numerical integration. The same numerical integration methods are used as for DFT, with grids adjusted to produce accurate integrals of the ZORA relativistic correction operator.

A further approximation is available, and is used as the default. As relativistic effects are localized near the nuclei, the ZORA integrals are evaluated as one-center integrals, with the model potential containing only the electronic Coulomb potential for that center. This is called the "local" approximation, and is gauge-invariant, and has zero derivatives with respect to nuclear displacements. The full evaluation of the integrals evaluated over all centers with potentials for all centers is labeled “global”. These integrals vary with internuclear distance.

The Jaguar Input File

This chapter describes the Jaguar input file and how to use it to run Jaguar from the command line. You might want to run Jaguar from the command line in order to submit a job at a later time when computers are less busy, to use batch scripts to run multiple jobs in succession, or to automate job submission with input files created by using other programs or by creating and editing input files yourself.

The sections in this chapter discuss the Jaguar input file format, describing the general file format first, then describing each section of the input file, starting with the geometry input (**zmat**) and the keyword (**gen**) sections.

In the tables of this chapter, default values of keywords are set in bold italic.

9.1 General Description of the Input File

The input file often begins with an optional line indicating the version number of Jaguar, such as v60012. The other parts of the input file are either single lines composed of options in capital letters followed by arguments on the same line; sections describing the molecule and the calculation, whose formats will be described later in this chapter; or comments.

9.1.1 Input File Format

The input file should have the following format, where square brackets denote optional entries, and entries in italics represent a character string with no spaces:

[*comments*]
{*sections describing molecule & calculation*}
[MAEFILE:*name*.mae[gz]]
[BASISFILE:*name*.basis]
[ATOMIGFILE:*name*.atomig]
[DAFFILE:*name*.daf]
[GRIDFILE:*name*.grid]
[CUTOFFFILE:*name*.cutoff]
[PBFRADIIFILE:*name*.ark]
[GPTSFILE:*name*]
[PBFPRMFILE:*name*]

The Maestro file specification (`MAEFILE`) is often included in the input file. It gives the location of the Maestro file that contains the structures corresponding to those in the Jaguar input file. The bonding information ("connection table") from this file is used as the basis for any Lewis structures that are required for the calculation. Properties from this file are copied to the output Maestro file. It is recommended that you always use the Maestro file where possible, particularly for solvation calculations. If you do not use it, Jaguar constructs a Lewis structure from the coordinates, but this does not always produce optimal Lewis structures.

The last eight lines are only rarely used. Therefore, your Jaguar input files will generally take a form as simple as

{*sections describing molecule & calculation*}

where only the **zmat** section, which contains the geometry, is actually required.

The `.basis`, `.atomig`, `.daf`, `.grid`, `.cutoff`, and `.ark` data files are described in Chapter 10. If you want to use non-default choices for any of these files, you can specify their paths and names on the appropriate lines of the input file. If a file name ends with `.Z` (for example, `BASISFILE: erwin.basis.Z`), Jaguar copies the file and uncompresses it. You can specify a file on another host, or under another account name on that host, by listing the file name in the format *host*`:`*fullpath* or *user*`@`*host*`:`*fullpath*.

The `GPTSFILE` line allows you to use grid points and weights from an input file for any one grid used during the calculation. The file should have a line for each grid point, and each line should list, in order, the *x*, *y*, and *z* Cartesian coordinates (in angstroms) and the weight for that grid point. Grid weights are only used in charge fitting, so if you don't want to use them, use 0 as a placeholder. For information about how to use this grid in a Jaguar calculation, see Section 9.5.27 on page 263.

The `PBFPRMFILE` line allows you to specify the path to the parameter file for the Poisson-Boltzmann solver. The default file is `$SCHRODINGER/jaguar-v`*version*`/data/pbf.prm`. This file contains parameters for the grids and numerical methods used.

Comments in the input file are ignored by Jaguar.

9.1.2 Sections Describing the Molecule and Calculation

The rest of the input file is composed of named sections. The sections may appear in any order. Character case (upper or lower) is ignored; therefore, either case, or a combination of the two, may be used. Equals signs (=), commas (,), blank spaces ( ), and tabs are all considered spacing characters; however, if you plan to use the GUI, we suggest that you use equals signs between a keyword and its value, and avoid using them anywhere else. Blank lines, or multiple spacing characters in a row, are equivalent to a single spacing character and thus may be used to improve readability.

The **gen** section contains a list of the general keywords which control the calculation. Defaults are provided for all unspecified keywords. The other sections contain lists, such as atomic coordinates. The sections are listed in Table 9.1. Each section has a distinct format; the formats are described in detail in the rest of this chapter. Keywords in the **gen** section can have integer, real, or character string values. Generally, valid integer values are limited to a small set which differs for each keyword. Real values can optionally include a “d” or “e” floating point power of ten. Character string keyword values may be limited to a small set, as for a basis set description, or may allow a general string like a file name.

Each section is delineated by a pair of “&” or “$” characters. The section name follows immediately after the first “&” or “$.” Thus, for example, the general keyword section may begin with “&gen” or “$gen” and ends with “&” or “$.” Within the **gen** section, allowed keywords are followed by numerical arguments giving their values, whose meanings are explained in Section 9.5 on page 204. At least one spacing character must precede and follow each keyword or keyword/value pair.

Table 9.1. Sections for Jaguar input files

Section	Description
zmat	Contains list of atomic coordinates describing molecular geometry, in Cartesian or Z-matrix format.
zvar	Sets values for **zmat** section variables.
coord	Specify particular internal coordinates to be used for optimization.
connect	Specify particular internal coordinates to be used when generating coordinates for optimization.
tvec	Specify reaction coordinate at transition state for IRC calculations.
gen	Sets general control keywords, including those describing the calculation performed, the grids, dealiasing functions, and cutoff parameters used, the electrostatic, geometry, and solvation properties calculated and the parameters used, and the output generated.
lmp2	Sets LMP2 pairs for local local MP2 calculations and delocalization of LMP2 pairs.
atomic	Sets atom-specific properties, including atomic masses (for isotopes), van der Waals radii for PBF solvation calculations, and basis functions for individual atoms.
hess	Allows input of initial nuclear Hessian.
guess	Allows input of initial wave function.
guess_basis	Specification of the basis set for the initial guess in the guess section.
pointch	Adds independent point charges.
efields	Adds electric field or fields.

Table 9.1. Sections for Jaguar input files (Continued)

Section	Description
egridpt	Grid points for calculation of electrostatic potential and electric field.
ham	Allows user input of Hamiltonian.
orbman	Allows orbitals to be reordered or linearly combined.
echo	One-word section indicating that the input file should be echoed in the output file.
path	Specifies execution path, listing order of Jaguar programs to be run.
nbo	Specifies NBO directives for NBO calculation.
core	Specifies `$CORE` keylist for NBO calculation.
choose	Specifies `$CHOOSE` keylist for NBO calculation.
nrtstr	Specifies `$NRTSTR` keylist for NBO calculation.

For example,

```
&gen iguess=0 molchg=1 &
```

sets the **iguess** and **molchg** keywords of the **gen** section to 0 and 1, respectively. Sections may span multiple lines, and more than one section may appear in a line. However, a **gen** section keyword and its value must be on the same line. The following example is interpreted in the same way as the **gen** section example given above:

```
This is a comment.
&gen iguess=0
   molchg=1 &
This is also a comment.
```

9.2 The zmat, zmat2, and zmat3 Sections

The molecular geometry must be described in the **zmat** section. Details on entering a geometry through the GUI can be found in Section 2.3 on page 11 and Section 2.4 on page 13. The units for the geometry are set by the **iunit** keyword of the **gen** section; by default, these units are angstroms and degrees.

If the geometry is in Cartesian coordinates, each line must contain four items: an atom name and the (x, y, z) coordinates. Each item should have at most 80 characters. The atom name should begin with the one- or two-letter element symbol, in either uppercase or lowercase characters. Other alphanumeric characters may be added, as long as the atomic symbol remains clear—for instance, `HE5` would be interpreted as helium atom 5, not hydrogen atom E5. Up to eight characters can be given in an atom name.

A sample Cartesian **zmat** section for a water molecule is:

```
&zmat
O   0.000000    0.000000   -0.113502
H1  0.000000    0.753108    0.454006
H2  0.000000   -0.753108    0.454006
&
```

If your input comes from the PDB, you may need to change some of the atom labels. An atom label like “CA” denotes calcium in the **zmat** section, but this symbol is the conventional PDB label for an alpha carbon in an amino acid residue. If you want to use PDB atom names in a Jaguar input file, you must insert an underscore after the atomic symbol, so that for example the alpha carbon would be represented as “C_A”. When the Jaguar input file contains a reference to a Maestro structure file, which is always the case when the input file is created using Maestro, a check will be performed to ensure consistency between the atomic symbol used for the atom label, and the atomic number. Thus, an atom label of “CA” whose associated atomic number is 6 produces an error.

A **zmat** section in Z-matrix format should not include lines defining variables (which are set in the **zvar** section described in Section 9.3 on page 200), and should not contain any comment lines, but otherwise should have the same format as described in Section 2.4.4, Section 2.4.5, and Section 2.4.6. Section 2.4.6 also includes a description of how to specify bond length or angle constraints on the Z-matrix coordinates for geometry optimizations.

You can orient the molecule or system using a label on the same line as the **zmat** section label. This orientation label should begin with the word `orient`, which is followed by an option in the form *ab*, *–ab*, *a–b*, or *–a–b*, where *a* and *b* are each either `x`, `y`, or `z` (for example, `&zmat orient x-y`). Jaguar places the first atom in the Z-matrix at the origin, the second along the *a*-axis (in the negative direction for *–a*), and the third atom in the *ab* plane, in the quadrant determined by the positive or negative signs of *a* and *b*.

To perform counterpoise calculations, you can specify counterpoise atoms, which have the usual basis functions for that element but include no nuclei or electrons, by placing an @ sign after the atom labels. For example, to place sodium basis functions at the Cartesian coordinates (0.0, 0.0, 1.0), you could include the following line in a Cartesian input file:

```
Na1@    0.0   0.0   1.0
```

You can also input counterpoise atoms for geometries in Z-matrix format.

If you are optimizing a molecular structure to obtain a minimum or a transition state, you might want to refine the Hessian used for the job. (See Section 4.3 on page 100 for information on the methods used for transition-state optimizations, including Hessian refinement.) If you add an asterisk (*) to a coordinate value, Jaguar computes the gradient of the energy both at the original geometry and at a geometry for which the asterisk-marked coordinate has been changed slightly, and will use the results to refine the initial Hessian to be used for the optimization. (To request refinement of a coordinate whose value is set using a variable, add an asterisk (*) to the end of the variable setting in the **zvar** section line that defines the variables.) For instance, a job that included Hessian refinement that was run with the following **zmat** section would use both O–H bonds and the H–O–H angle in the refinement:

```
&zmat
O1
H2   O1   1.1*
H3   O1   1.1*   H2   108.0*
&
```

Molecular symmetry or the use of variables, either of which may constrain several coordinate values to be equal to each other, can reduce the number of coordinates actually used for refinement. For instance, for the water input example shown above, only two coordinates will actually be refined (the O–H bond distance, which is the same for both bonds, and the H–O–H angle) if molecular symmetry is used for the job.

Some types of transition-state optimizations require two or three geometries (see Section 4.3 on page 100). For these jobs, you can input the second and third geometries (Geometry 2 and Geometry 3) in the **zmat2** and **zmat3** sections. The order of atoms in the input must be the same as in the **zmat** section. Alternatively, if the changing coordinates in the **zmat** section are set using variables, you can leave out the **zmat2** and **zmat3** sections and specify the second and third geometries by adding **zvar2** and **zvar3** sections, which will be used in combination with the **zmat** section to define the second and third geometries. See Section 9.3 for details.

9.3 The zvar, zvar2, and zvar3 Sections

The **zvar** section should contain equations setting the values of any variables in the **zmat** section, in the same units used for the **zmat** section. Here is a sample **zvar** section:

```
&zvar
ycoor=0.753108  zcoor=0.454006
&
```

For an optimization, to constrain (freeze) all bond lengths or angles set to a particular variable, you should add a # sign to the end of the **zvar** section equation setting that variable. Similarly, to request Hessian refinement of a coordinate whose value is determined by a variable setting in the **zvar** section, add an asterisk (*) to the end of the equation that sets the variable value in the **zvar** section. For example, the **zvar** section

```
&zvar
ycoor=0.753108#  zcoor=0.454006
&
```

would freeze all `ycoor` values to 0.753108 during an optimization job.

Certain types of transition-state optimizations require that you enter two or three geometries (see Section 4.3 on page 100 for details). For these jobs, you can specify variables for the second and/or third geometries in the **zvar2** and **zvar3** sections. If no **zmat2** or **zmat3** sections exist, these variables are used in combination with the **zmat** section to define the second and third geometries.

The equation that defines a variable can also specify a range of values for coordinate scans. You can assign a list of values to the variable in the format `at` *number-list*, or you can assign the initial value, specified by *number* or `from` *number,* and two values from the following list, in the order given in the list:

- The final value of the coordinate, specified by `to` *number*
- The step size, specified by `by` *number*
- The number of steps, specified by `in` *integer*

Here, *integer* means an appropriate integer and *number* means an appropriate real number. See Section 4.4.2 on page 108 for information and examples.

9.4 The coord and connect Sections

For some geometry or transition-state optimizations, you might want to specify that the optimizer use particular internal coordinates. For example, if you study a bond-forming reaction, you can require Jaguar to use the bond in question as an internal coordinate even when the bond distance is very long. You also might want to generate your own list of internal coordinates for cases that involve multiple separate (unbonded) fragments.

It is often useful to specify internal coordinates for pairs of atoms that are on separate sections of a large floppy molecule, but are close to being in van der Waals contact. Otherwise, small changes in a torsional coordinate far away from these atoms can then lead to steep changes in the energy. Adding explicit coordinates for these non-bonded contacts makes it possible for the optimization algorithm to control their approach more effectively.

For transition state searches, where bonds are being broken and formed, it is important to specify the atoms involved as being bonded, because the broken bonds might be outside the normal range for bonding, and not be included when internal coordinates are generated.

To control the internal coordinates used in an optimization, you should first make sure that Jaguar is going to generate internal coordinates for the job. Optimization jobs generate and use redundant internal coordinates unless you have set the keyword **intopt** in the **gen** section of your input file. (See Section 9.5.10 on page 217 for more details.)

To require particular bonds or angles to be included in the internal coordinates used for an optimization, use a **coord** section. Each line of a **coord** section should contain a list of atoms used to specify a bond, bond angle, or torsional angle coordinate to be included among the internal coordinates generated by Jaguar.

To specify atoms that are considered to be bonded, use a **connect** section. This section is used by the coordinate generator to generate internal coordinates The bonds in the connect section are used to generate angles and torsions involving those bonds.

9.4.1 Constrained Coordinates

If you want to hold a coordinate fixed at its initial value throughout the job, add the entry "#" to the end of the line (after one or more spaces).

As an example, the **coord** section

```
&coord
C1  C2
C1  C2  C3 #
C1  C2  C3  C4
&
```

requests that the set of internal coordinates include the C1–C2 bond, the C1–C2–C3 bond angle (which is to be held frozen throughout the optimization), and the C1–C2–C3–C4 torsion.

You can specify a value after the # sign, separated by a space. If this value is different from the current value of the coordinate according to the geometry, it will be used as a dynamic constraint. For example, consider the following **zmat** section for water, in which the distance between the two hydrogen atoms is 1.507 angstroms:

```
&zmat
o
h1    o  0.95
h2    o  0.95    h1  105
&
```

Now suppose you want to optimize the geometry subject to the constraint that the distance between the hydrogen atoms is 2.0 Å. Then you would add the following **coord** section:

```
&coord
h1 h2 # 2.0
&
```

You can specify a variable after the # sign, separated by a space. The values that the variable takes must be given in a **zvar** section. The following example defines a variable HH as the distance between H1 and H2.

```
&coord
H1 H2 # HH
&
```

For torsional (dihedral) angles, you can fix the “natural torsional angle” by specifying the bond about which rotation can take place, followed by #nt or #NT (no spaces), as in the following example.

```
&coord
C1 C2 #nt
&
```

The natural torsional angle is the average of all the torsional angles that can be defined using this bond and the atoms bonded to either end of it.

Harmonic constraints can be set on the Cartesian position of an atom or on any bond length, angle, or dihedral angle by adding #hc or #HC after the coordinate. The Cartesian position is specified by a single atom label, as in the following example.

```
&coord
C1 #hc 10.0
&
```

Two atom labels specify a bond, three labels a bond angle, and four labels a dihedral angle, as usual. The value of a force constant for the harmonic potential must follow #hc or #HC, in units specified by the gen section keywords **iunit** and **eunit**. The force constant can be followed by the radius *a* of a region in which the constraining potential is zero, and this radius can be followed by a target value for the coordinate if it is an internal coordinate. A target value cannot be specified for a Cartesian harmonic constraint. The following example specifies a harmonic constraint on a bond length, with a force constant of 10.0 kcal mol^{-1} $Å^{-2}$, a radius of 0.5 Å, and a target bond length of 1.5 Å:

```
&coord
C1 C2 #hc 10.0 0.5 1.5
&
```

For information on recommended harmonic constraint ranges, see Section 4.2.2 on page 97.

9.4.2 Specifying Bonds for Internal Coordinates with a connect Section

You can use a **connect** section to specify the bonds used by Jaguar in its generation of internal coordinates. This is particularly important for transition state searches, where bonds are broken and formed. Each line of a **connect** section should list two atoms by either their atom labels (such as H2 for a hydrogen) or their atom numbers (such as 3 for the third atom listed in the **zmat** section input). Here is a sample **connect** section:

```
&connect
C1  C2
C2  C3
&
```

The two atoms on each line of the **connect** section are then treated as nearest neighbors by the program when it generates redundant internal coordinates for the optimization. Consequently, the internal coordinates generated by Jaguar include the bond between those two atoms and angles between those two atoms and any other atoms that are nearest neighbors to either of them. For the sample **connect** section above, for instance, the redundant internal coordinates would include the C1–C2 bond, the C2–C3 bond, and the C1–C2–C3 angle in addition to whatever internal coordinates would be generated without the **connect** section.

9.5 The gen Section

The keywords of the **gen** section allow control over how the calculation is performed. Many of these keywords can be set from the GUI. See Chapter 3 and Chapter 6 for details.

In this section, the default values for keywords are indicated in bold italics. The keywords for geometry input are described first, followed by those relating to correlation methods, optimization to a minimum-energy structure or transition state, calculations in solution, calculation of various molecular properties, basis sets, SCF methods, and output. These subsections correspond to the order of information in Chapter 3 and Chapter 6. Finally, keywords relating to grids and dealiasing functions, cutoff parameters, and memory usage are described.

9.5.1 Units Keywords

The keywords **iunit**, **eunit**, and **espunit** set units for geometry, energy, and electrostatic potential units. The **iunit** keyword, whose default value is 1, describes what units the geometry is assumed to have, as indicated in Table 9.2. It also sets the length units for plot grids and for the location of point charges. The **eunit** keyword sets the energy units for thermochemical properties and for harmonic potentials used for harmonic constraints. The **espunit** keyword sets the units for output of electrostatic potentials.

Table 9.2. Options for the units keywords

Keyword	Value	Description
iunit	0	Geometry units are bohr and radians
	1	Geometry units are angstroms and degrees
	2	Geometry units are bohr and degrees
	3	Geometry units are angstroms and radians
eunit	***1***	Energy units are kcal/mol
	2	Energy units are kJ/mol

Table 9.2. Options for the units keywords (Continued)

Keyword	Value	Description
espunit	1	Electrostatic potential in units of kcal mol^{-1} electron^{-1}. (Useful for comparing molecules)
	2	Electrostatic potential in units of kT electron^{-1} at 298.15 K. (Useful for comparing molecules)
	3	Electrostatic potential in atomic units.
	4	Electrostatic potential in units of kcal mol^{-1}.
	5	Electrostatic potential in units of kT mol^{-1} at 298.15 K.
	6	Electrostatic potential in units of electron volts (eV).

9.5.2 Covalent Bonding Keyword

The real-valued keyword **covfac** determines which atoms are considered to be bonded. Two atoms are bonded if they are closer to each other than **covfac** times the sum of their covalent radii, which are listed in Table 9.48. The default value for this variable is 1.2.

9.5.3 Molecular State Keywords (Charge and Multiplicity)

The keywords that describe the input molecule's charge and spin multiplicity are shown in Table 9.3. These keywords correspond to GUI options described in Section 2.6 on page 21.

Table 9.3. Keywords to describe the molecular state

Keyword	Value	Description
molchg	any	Overall charge on molecule, excluding point charges set in **pointch** section (default is 0)
multip	>0	Spin multiplicity: 1 for singlet, 2 for doublet, etc. (default is 1, except for **ihamtyp**=0, when **multip**=2 by default)

9.5.4 Atomic Mass Keyword

The keyword **massav** determines the atomic masses used for any atoms whose masses or isotopes are not specifically set in the **atomic** section (see Section 9.8 on page 273). The masses used are from ref. 194.

Table 9.4. Keyword to describe the atomic masses used

Keyword	Value	Description
massav	***0***	Use masses of most abundant isotopes as atomic masses
	1	Use average isotopic masses as atomic masses, where averages are weighted according to natural abundance of isotopes

9.5.5 Symmetry-Related Keywords

By default, for most calculations, Jaguar takes advantage molecular symmetry to reduce computing time, as described in Section 2.7.2 on page 23. Several integer-valued keywords shown in Table 9.5 describe how the program uses symmetry.

Table 9.5. Symmetry-related keywords in Jaguar

Keyword	Value	Description
isymm	0	Do not use symmetry
	1	Rotate atomic grids to match molecular symmetry, if possible
	2	Change grids to get molecular symmetry, if necessary
	8	Use symmetry in preprocessing and SCF
ipopsym	***0***	Allow change in number of electrons in each irreducible representation (default for HF and DFT closed-shell jobs)
	1	Don’t allow number of electrons in each irreducible representation to change (default for non-HF, non-DFT and open-shell calculations)
idoabe	***0***	Allow non-Abelian point group symmetry assignment
	1	Allow only Abelian point group symmetry assignment

9.5.6 LMP2 Keywords

The **mp2** keyword allows you to request a local Møller-Plesset perturbation theory (LMP2) calculation. By default, LMP2 is off. For more information on the local MP2 method, see Section 3.5 on page 50 and Section 8.2 on page 185. LMP2 keywords are given in Table 9.6.

LMP2 calculations require a basis set that is set up for the pseudospectral method—see Table 3.1 on page 34 and Table 3.2 on page 36 for basis set information. Because the available basis sets are not designed for core correlation, you should not use the **mp2**=1 setting. Calculations performed with this setting are likely to be misleading. This is because core correlation requires basis functions that have a similar mean radius to the core functions that are correlated, including higher angular momentum functions. For example, to correlate the 1s in a

Table 9.6. Keyword settings for local MP2 calculations

Keyword	Value	Description
mp2	***0***	Do not run LMP2 calculation
	1	Correlate core and valence electrons (not recommended: see text)
	3	Run LMP2 calculation (for valence electrons only)
idelocv	***0***	Do not delocalize any pairs listed in **lmp2** section (default for all calculations except those with **iqst**>0 and/or **ireson**>0)
	1	Treat all LMP2 pairs, but delocalize any pairs in **lmp2** section as indicated there, or (default for QST-guided transition-state searches) delocalize any pairs on atoms with breaking or forming bonds
	2	Perform a local local MP2 calculation, treating only pairs listed in the **lmp2** section at the LMP2 level, and also delocalize any pairs in **lmp2** section as indicated there
ireson	***0***	Do not delocalize LMP2 pairs over other atoms
	1	Delocalize each bond pair in an aromatic ring over the neighboring atoms in the ring
	2	Delocalize bond pairs in an aromatic ring over all atoms in the ring
iheter	***0***	Treat all atoms with LMP2 if LMP2 is on unless **lmp2** section exists; if LMP2 is on and **lmp2** section exists, set atom pairs in **lmp2** section
	1	Treat only heteroatom pairs (atoms in bonds with atoms of other elements, except C atoms bonded only to C and/or H) and any pairs set in **lmp2** section at LMP2 level, other atoms at HF level
loclmp2c	***0***	Do not localize core orbitals for LMP2 calculation
	1	Perform Boys localization on core orbitals for LMP2 calculation
	2	Perform Pipek-Mezey localization on core orbitals for LMP2 calculation, maximizing Mulliken atomic populations
	3	Perform Pipek-Mezey localization on core orbitals for LMP2 calculation, maximizing Mulliken basis function populations
loclmp2v	1	Perform Boys localization on valence orbitals for LMP2 calculation
	2	Perform Pipek-Mezey localization on valence orbitals for LMP2 calculation, maximizing Mulliken atomic populations
	3	Perform Pipek-Mezey localization on valence orbitals for LMP2 calculation, maximizing Mulliken basis function populations. This scheme produces results that are significantly different from the other two schemes.

double-zeta basis, an s function and a p function with a similar mean radius to the 1s orbital are needed. The pseudospectral basis sets do not have such functions.

Local MP2 calculations use the LMP2 method for all atoms unless the **lmp2** section (described in Section 9.6 on page 270) is used to set local LMP2 pairs or unless the keyword **iheter** is set to 1. The **iheter** and **mp2** keyword settings are described in Table 9.6.

For LMP2 calculations, Jaguar needs to obtain localized orbitals. By default, Jaguar uses the Pipek-Mezey method to perform the localization. If Pipek-Mezey localization does not converge for a particular case, you might want to try Boys localization by changing the settings for the keywords **loclmp2c** and **loclmp2v**, as indicated in Table 9.6. If you are performing a set of calculations to compare against each other, you should use the same localization method for all of the calculations. In particular, the **loclmp2v**=3 scheme is known to affect the LMP2 energy significantly compared to the other two localization schemes.

9.5.7 Relativistic Hamiltonian Keyword

To use a Hamiltonian that includes relativistic effects, you can set the **relham** keyword. The allowed values and the selection that they make are given in Table 9.7. The **relham** keyword is not compatible with the use of effective core potentials, so you should not choose a basis set with an effective core potential. (Effective core potentials can be relativistic, but they are automatically included along with the basis set, not via a keyword.) You can make settings for some details of the ZORA Hamiltonian in the **relativity** section—see Section 9.7. Although you can use the ZORA Hamiltonian with any basis set, we recommend that you use the basis sets that are contracted specifically for ZORA (with ZORA in the name).

Table 9.7. Allowed values of the relham keyword

Value	Description
`zora-scalar` | `zora-1c`	Use the spin-free ZORA Hamiltonian (also called scalar ZORA or one-component ZORA).

9.5.8 DFT Keywords

To use density functional theory (DFT), you should set the **dftname** keyword. Most DFT options described here are also available from the GUI (see Section 3.3 on page 38. Keywords for DFT grids are given in Section 9.5.27 on page 263.

The **dftname** keyword can be given as a standard functional name, as listed in Table 9.8, or it can be constructed from a set of functional name strings for exchange and correlation functionals, which are listed in Table 9.9. For example, **dftname**=bp86 specifies the BP86 functional, and is a combination of `b` for exchange and `p86` for correlation.

For many of the functionals, the D3 dispersion correction of Grimme et al [50, 51] is available, and can be specified by appending `-d3` to the functional name. The functionals for which this correction is available are marked in the table. Long-range corrected functionals are listed; ω is the range separation parameter in bohr^{-1}.

Table 9.8. Standard functional names for the **dftname** *keyword*

Name	Description
`hfs`	Slater local exchange [34]
`xalpha`	Xα local exchange [34].
`hfb`	Slater local exchange [34], Becke 1988 non-local gradient correction to exchange [37].
`hfpw`	Slater local exchange [34], Perdew-Wang 1991 GGA-II nonlocal exchange [36].
`bp86`	Exchange: Slater local functional [34], Becke 1988 non-local gradient correction [37]; correlation: Perdew-Zunger 1981 local functional [39], Perdew 1986 gradient correction [40]; D3 correction available.
`bp86-vwn5`	Exchange: Slater local functional [34], Becke 1988 non-local gradient correction [37]; correlation: Vosko-Wilk-Nusair (VWN) local functional [35], Perdew 1986 gradient correction [40].
`pwpw91`	Exchange: Slater local functional [34], Perdew-Wang 1991 gradient correction [36]; correlation: Perdew-Wang 1991 GGA-II local and nonlocal functionals [36].
`hcth407`	Hamprecht-Cohen-Tozer-Handy functional including local and nonlocal exchange and correlation, reparametrized with a training set of 407 molecules by Boese and Handy [46].
`pbe`	Perdew-Burke-Ernzerhof local and nonlocal exchange and correlation [47]. D3 correction available.
`pbe0` `pbe1pbe`	Functional due to Adamo and Barone [53] based on PBE functional. Exchange: 0.25 HF exchange, 0.75 PBE non-local exchange. Correlation: Perdew-Burke-Ernzerhof [47] local and nonlocal correlation. Also known as PBE1PBE. Either name can be used for the keyword value. D3 correction available.
`blyp`	Exchange: Slater local functional [34], Becke 1988 nonlocal gradient correction [37]; correlation: Vosko-Wilk-Nusair (VWN) local functional [35], Lee-Yang-Parr local and nonlocal functional [38]. D3 correction available.
`olyp`	Slater local exchange [34], OPTX nonlocal exchange of Handy and Cohen [48] with Lee-Yang-Parr local and nonlocal correlation functionals [38]. D3 correction available.
`b3lyp`	Exchange: exact HF, Slater local functional [34], Becke 1988 nonlocal gradient correction [37]; correlation: Vosko-Wilk-Nusair (VWN) local functional [35], Lee-Yang-Parr local and nonlocal functional [38]. D3 correction available.

Table 9.8. Standard functional names for the **dftname** *keyword (Continued)*

Name	Description
`x3lyp`	Extension of B3LYP by Xu and Goddard to include Perdew-Wang 1991 gradient correction exchange functional [36], with exchange parametrized to fit Gaussian exchange density [55].
`o3lyp`	Hybrid functional using HF exchange, Slater local exchange [34], and OPTX nonlocal exchange of Handy and Cohen [48], with Lee-Yang-Parr local and nonlocal correlation functionals [38]
`b97-d`	Modified Becke 1997 functional [42] with Grimme's long-range dispersion correction, useful for noncovalent interactions [49]. D3 correction also available (`b97-d3`).
`b3pw91`	Exchange: exact HF, Slater local functional [34], Becke 1988 non-local gradient correction [37]; correlation: Perdew-Wang 1991 local and GGA-II nonlocal functional [36]. D3 correction available.
`b3p86`	Exchange: exact HF, Slater local exchange functional [34], Becke 1988 nonlocal gradient correction [37]; correlation: Vosko-Wilk-Nusair (VWN) local functional [35] and Perdew 1986 nonlocal gradient correction [40]
`b1b95`	Becke's hybrid functional from 1996 containing 28% HF exact exchange [41].
`bb1k`	Reparametrization of Becke's 1996 hybrid functional for kinetics (reaction barriers), containing 42% HF exact exchange [56].
`bhandh`	50% exact HF exchange, 50% Slater local exchange functional [34].
`bhandhlyp`	Exchange: 50% exact HF exchange, 50% Slater local exchange functional [34]; correlation: Lee-Yang-Parr local and nonlocal functional [38].
`b97-1`	Reparametrization of Becke's 1997 hybrid functional [42] by Hamprecht, Cohen, Tozer, and Handy [45].
`b98`	Becke's 1998 hybrid functional including the Laplacian of the density and kinetic energy density terms as well as gradient terms [43].
`sb98`	Schmider and Becke reparametrization of Becke's 1998 functional [44].
`mpw1pw91`	Hybrid functional with modification of Perdew-Wang gradient correction exchange functional by Adamo and Barone [52]. Exchange: 25% exact HF exchange, 75% Slater local functional [34] and Perdew-Wang 1991 gradient correction functional [36]; correlation: Perdew-Wang 1991 GGA-II local and nonlocal functionals [36].
`mpw1k`	Reoptimization of *m*PW1PW91 functional parameter for prediction of barrier heights, by Lynch, Fast, Harris, and Truhlar [54].
`pwb6k`	Reoptimization of MPWB1K functional for simultaneous accuracy of bond energies, barrier heights, and nonbonded interactions, by Zhao and Truhlar [56].

Table 9.8. Standard functional names for the **dftname** *keyword (Continued)*

Name	Description
`pw6b95`	Reoptimization of MPW1B95 functional for simultaneous accuracy of bond energies, barrier heights, and nonbonded interactions, by Zhao and Truhlar [56]. D3 correction available.
`m05`	Hybrid functional parametrized for broad accuracy, including noncovalent interactions, kinetics, and interactions with metals, by Zhao, Schultz, and Truhlar [58, 59]. D3 correction available.
`m05-2x`	Hybrid functional with larger HF exchange component, similar to M05 but parametrized for nonmetals, by Zhao, Schultz, and Truhlar [58, 59]. D3 correction available.
`m06`	Zhao and Truhlar functional, parametrized with metallic systems, for organometallic and inorganic chemistry and noncovalent interactions [62]. D3 correction available.
`m06-2x`	Zhao and Truhlar functional, parametrized for nonmetals, for main-group thermochemistry, kinetics, noncovalent interactions, and electronic excitation energies to valence and Rydberg states [62]. D3 correction available.
`m06-l`	Zhao and Truhlar gradient-corrected functional [60].
`m06-hf`	Zhao and Truhlar functional with full HF exchange and M06 local functionals that eliminates long-range self-interaction [61]. D3 correction available.
`m08-hx`	Zhao and Truhlar functional, constraining the reduced density gradient to exact exchange and correlation functional forms through second order. Parametrized on a broad range of properties [63].
`m08-so`	Zhao and Truhlar functional, constraining the reduced density gradient to exact exchange and correlation functional forms through second order. Parametrized on a broad range of properties [63].
`lrc-blyp`	Yanai, Tew, and Handy long-range corrected functional [65]. Short-range exchange: SR-Becke 88 [37]; long-range exchange: LR-HF; ω=0.33; correlation: LYP [38].
`cam-b3lyp`	Yanai, Tew, and Handy long-range corrected functional [65]. Short-range exchange: 19% SR-HF, 81% SR-Becke 88 [37]; long-range exchange: 65% LR-HF, 35% LR-Becke 88 [37]; ω=0.33; correlation: LYP [38]; D3 correction available.
`upbe`	Long-range corrected PBE functional. Short range exchange: SR-PBE; long range exchange: LR-HF; ω=0.30; correlation: PBE [47].
`upbe0`	Long-range corrected PBE hybrid. Short range exchange: 25% SR-HF, 75% SR-PBE; long range exchange: LR-HF; ω=0.30; correlation: PBE [47].
`wpbe`, `lc-wpbe`	Vydrov and Scuseria long-range corrected functional [67]. Short range exchange: SR-PBE; long range exchange: LR-HF; ω=0.4; correlation: PBE [47]; D3 correction available.

Table 9.8. Standard functional names for the **dftname** *keyword (Continued)*

Name	Description
`wpbeh`	Rohrdanz, Martins, and Herbert long-range corrected functional [68]. Short range exchange: 20% SR-HF, 80% SR-PBE; long range exchange: LR-HF; ω=0.2; correlation: PBE [47].
`hse03`	Heyd, Scuseria, and Ernzerhof functional [69]. Short range exchange: 25% SR-HF, 75% SR-PBE; long-range exchange: LR-PBE; ω=0.33[a]; correlation: PBE [47].
`hse06`	Heyd, Scuseria, and Ernzerhof functional [71]. Short range exchange: 25% SR-HF, 75% SR-PBE; long-range exchange: LR-PBE; ω=0.11; correlation: PBE [47].
`m11`	Peverati and Truhlar functional [72]. Short range exchange: 42.8% SR-M11 [72] and HF; long-range exchange: HF; ω=0.25; correlation: M11 [72].
`m11-l`	Peverati and Truhlar range separation of non-hybrid functionals [73]. Short range exchange: SR-M11 [73]; long-range exchange: LR-M11 [73]; ω=0.25; correlation: M11-L [73].
`wb97`	Chai and Head-Gordon functional [73]. Short range exchange: SR-Becke 97 [42]; long-range exchange: HF; ω=0.4; correlation: Becke 97 [42].
`wb97x`	Chai and Head-Gordon functional [73]. Short-range exchange: 16% SR-HF, SR-Becke 97 [42]; long-range exchange: HF; ω=0.3; correlation: Becke 97 [42].
`bnl`	Baer, Neuhauser, and Livshits functional [75, 76]. Short-range exchange: 90% SR-Savin; long-range exchange: HF; ω=0.3; correlation: LYP.

a. This is the value used in NWChem; it is not the value reported in the paper or the values actually used in the paper: see the erratum in Ref. 70.

Table 9.9. Functional name strings for construction of the **dftname** *keyword*

Name String	Functional Description
`s`	Slater local exchange
`xa`	Xα local exchange
`b`	Becke 1988 nonlocal exchange, Slater local exchange
`pw`	Perdew-Wang 1991 GGA-II nonlocal exchange, Slater local exchange
`vwn`	Vosko-Wilk-Nusair local correlation
`vwn5`	Vosko-Wilk-Nusair 5 local correlation
`pl`	Perdew-Zunger 1981 local correlation
`p86`	Perdew-Zunger 1981 local correlation, Perdew 1986 nonlocal gradient correction
`pw91`	Perdew-Wang GGA-II 1991 local and nonlocal correlation
`lyp`	Lee-Yang-Parr local and nonlocal correlation

The *ulg* a posteriori correction of Kim, Choi, and Goddard [87] is available for all functionals and can be specified by appending `-ulg` to the functional name. This functional is available for all elements up to Z=103.

The a posteriori localized orbital correction (LOC) [78-86] is available for the B3LYP functional. It is specified by `b3lyp-loc`.

If you want to build a custom functional, you can use the keywords **xhf**, **xexl1**, **xexl9**, and **xexnl***n*, which determine the contributions of the exact exchange and the exchange functionals, and the keywords **xcorl***n* and **xcornl***n*, which control the contributions of the correlation functionals, as listed in Table 9.10.

If you want to evaluate the energy of the final, post-SCF wave function using a particular functional or combination of functionals, you can use the **pdftname** keyword, which has the same values as the **dftname** keyword. You can also use the keywords **yhf**, **yexl1**, **yexl9**, **yexnl***n*, **ycorl***n*, and **ycornl***n*, which correspond to the keywords in Table 9.10 (e.g., **xexl1**). If you do a post-SCF DFT energy evaluation, you should not perform a geometry optimization or calculate the solvation energy, polarizability, or any other properties.

Table 9.10. Functional coefficient keywords

Keyword	Corresponding Functional (or Exact Exchange)
xhf	exact exchange (Hartree-Fock)
xexl1	Slater local exchange functional
xexl9	Xα local exchange functional
xexnl1	Becke 1988 nonlocal gradient correction to exchange
xexnl3	Becke 1998 (B98) local and nonlocal exchange functional
xexnl4	Perdew-Wang GGA-II, 1991 nonlocal exchange functional
xexnl6	Schmider and Becke 1998 (SB98) local and nonlocal exchange functional
xexnl7	HCTH407 local and nonlocal exchange functional
xexnl8	B97 local and nonlocal exchange functional
xexnl9	PBE local and nonlocal exchange functional
xexnl11	Perdew-Wang 1991 exchange as modified by Adamo and Barone (mPW)
xexnl12	mPW as modified by Zhao and Truhlar for PWB6K
xexnl13	mPW as modified by Zhao and Truhlar for PW6B95
xexnl14	M05 local and non-local exchange functional
xexnl15	M05-2X local and non-local exchange functional

Table 9.10. Functional coefficient keywords (Continued)

Keyword	Corresponding Functional (or Exact Exchange)
xexnl16	M06-L local and non-local exchange
xexnl17	M06-HF local and non-local exchange
xexnl18	M06 local and non-local exchange
xexnl19	M06-2X local and non-local exchange
xexnl20	OPTX non-local exchange
xcorl1	VWN local correlation functional
xcorl2	VWN5 local correlation functional
xcorl3	Perdew-Zunger 1981 local correlation functional
xcorl4	Perdew-Wang GGA-II 1991 local correlation functional
xcornl1	Perdew 1986 non-local gradient correction
xcornl2	Lee-Yang-Parr local and nonlocal correlation functional
xcornl3	HCTH407 local and nonlocal correlation functional
xcornl4	Perdew-Wang GGA-II 1991 nonlocal correlation functional
xcornl6	Becke 1998 (B98) local and nonlocal correlation functional
xcornl7	Schmider and Becke 1998 (SB98) local and nonlocal correlation functional
xcornl8	B97-1 local and nonlocal correlation functional
xcornl9	PBE nonlocal correlation functional
xcornl11	B95 nonlocal correlation functional [41]
xcornl12	B95 as modified by Zhao and Truhlar for PWB6K
xcornl13	B95 as modified by Zhao and Truhlar for PW6B95
xcornl14	M05 local and non-local exchange
xcornl15	M05-2X local and non-local exchange
xcornl16	M06-L local and non-local correlation
xcornl17	M06-HF local and non-local correlation
xcornl18	M06 local and non-local correlation
xcornl19	M06-2X local and non-local correlation

For DFT jobs on metal-containing systems (p-block, d-block, or f-block), the keyword **vshift** is set to 0.2 for hybrid methods or 0.3 for non-hybrid methods by default, and the keyword **idenavg** is set to 1 by default, to aid convergence.

More complete descriptions and references for each DFT functional and hybrid are given in Section 3.3 on page 38.

Keywords for the B3LYP-MM and the ulg a posteriori corrections and keywords for printing a breakdown of the a posteriori corrections are described in Table 9.11.

Table 9.11. Keywords for a posteriori corrections.

Keyword	Value	Description
cpoise	***0***	Include counterpoise correction in parametrization of MM correction of B3LYP-MM [65]. Calculations with ghost atoms are not needed for BSSE-corrected dissociation energies.
	1	Use parametrization of MM correction that does not include counter-poise corrections in B3LYP-MM [65]. Calculations with ghost atoms must be performed for BSSE-corrected dissociation energies. Useful with `fall_through=1` for calculations with non-parametrized elements.
fall_through	***0***	Stop B3LYP-MM calculation if input includes non-parametrized elements, i.e. elements other than H, C, N, O, F, S, Cl, Br, and metal ions.
	1	Calculate B3LYP-MM corrections [65] for parametrized elements only, setting corrections for non-parametrized elements to zero.
ulg_slg	***0.7***	Prefactor s_{lg} for dispersion term in the *ulg* correction [87]. Default for PBE is 0.7012.
ulg_blg	***0.7***	Prefactor b_{lg} for R_0^6 term in the denominator of the dispersion term for the *ulg* correction [87]. Default for PBE is 0.6966.
dryaposteri	***0***	Do not calculate the a posteriori correction during initialization.
	1	Calculate the a posteriori correction during initialization (in `pre`), and add the contents of the a posteriori log file containing the energy corrections breakdown to the output file.
printaposteri	***0***	Do not print the a posteriori energy corrections breakdown log file at the end of the SCF stage.
	1	Print the a posteriori energy corrections breakdown log file at the end of the SCF stage.

Constrained DFT calculations, in which charge density is constrained to a selected set of atoms, can be performed by adding the keyword **icdft** and setting it to 1. You must also add a **cdft** section to specify the constraints. This method is used in electron transfer calculations.

9.5.9 TDHF and TDDFT Keywords

The time-dependent Hartree-Fock (TDHF) and time-dependent density-functional theory (TDDFT) methods can be used after an SCF calculation to generate information on excited states. The output includes energies, oscillator strengths and transition dipole moments for excitations from the ground state.

The keywords for the TDHF or TDDFT calculation are listed in Table 9.12. A TDDFT calculation is performed if a density functional has been specified, otherwise a TDHF calculation is performed. The rest of the calculation should be set up as an SCF calculation.

For a closed-shell system, you can choose to generate singlet excited states or triplet excited states, or both. The states are spin-adapted. You can run TDHF or TDDFT calculations on open-shell systems with a UHF or UDFT reference function. These calculations generate all singly-excited states, including states of higher or lower spin arising from a single spin flip, but there is no spin adaption.

Table 9.12. Keywords for CIS and TDDFT calculations

Keyword	Value	Description
icis	***0***	Do not do a CIS or TDDFT calculation with the old implementation.
	1	Do a CIS or TDDFT calculation with the old implementation. Deprecated. Use **itddft** instead.
itddft	***0***	Do not do a TDDFT or TDHF calculation.
	1	Do a TDDFT or TDHF calculation.
itda	***0***	Use the full linear response in TDDFT calculations
	1	Use the Tamm-Dancoff approximation (TDA) in TDDFT calculations, do a CIS calculation for HF wave functions.
rsinglet	***0***	Do not calculate restricted singlet excited states.
	1	Calculate restricted singlet excited states. Only applies if **iuhf**=0, and only valid for a closed-shell reference.
rtriplet	***0***	Do not calculate restricted triplet excited states.
	1	Calculate restricted triplet excited states. Only applies if **iuhf**=0, and only valid for a closed-shell reference.
nroot	>0	Number of roots to find. Default value is 1.
dconvci	***1.0e-2***	Convergence criterion for the norm of the residual vector (default is 1e-5 for a non-pseudospectral calculation)
econvci	***1.0e-5***	Convergence criterion for the change in energy (default is 1e-8 for a non-pseudospectral calculation)

Table 9.12. Keywords for CIS and TDDFT calculations

Keyword	Value	Description
nrestart	>0	Number of iterations before restarting
	0	Determine number of iterations before restarting automatically
maxciit	***32***	Maximum number of iterations used

You should not normally need to set **nrestart**, because the program determines how many iterations it can do with the amount of memory available.

You should use the **ittdft** keyword, as it selects a much faster and more accurate algorithm. The old **icis** keyword is deprecated and may be removed in a future release.

9.5.10 Geometry Optimization and Transition-State Keywords

Many of the keyword settings for optimization of minimum-energy structures and transition states described in this subsection can be made from the GUI, as described in Chapter 4, which also contains more details about the methods used for optimizations.

Table 9.13 contains general optimization keywords that apply to all kinds of optimizations. Most default values for the integer keywords are indicated in bold italics, and only the values listed in the table are allowed. In cases where the default is different for optimizations to minimum-energy structures than it is for transition-state optimizations, both defaults are in bold italics, and the cases for which each is a default are explained in the keyword description.

Table 9.13. General geometry optimization keywords

Keyword	Value	Description
igeopt	***0***	Do not optimize molecular geometry
	1	Optimize minimum-energy structure
	–1	Calculate forces but do not perform geometry optimization
	2	Optimize transition-state geometry
nogas	***0***	For optimizations in solution, perform gas phase geometry optimization first (to get accurate solvation energy)
	1	For optimizations in solution, skip gas phase geometry optimization and compute solvation energies using **esolv0** value (from input file) as gas phase energy (should yield same *structure* as **nogas**=0)
	2	For optimizations in solution, skip gas phase geometry optimization and compute solvation energies using energy of initial structure as gas phase energy (should yield same *structure* as **nogas**=0)

Table 9.13. General geometry optimization keywords (Continued)

Keyword	Value	Description
maxitg	>0	Maximum number of optimization iterations (maximum number of structures generated); default is ***100***
nmder	***0***	If calculating forces, compute analytic derivatives of energy
	1	If calculating forces, compute numerical derivatives of energy (obtained from calculations on 6 N_{atom} perturbed geometries by moving each atom **pertnd** bohr in positive or negative x, y, or z direction)
	2	Calculate frequencies numerically
intopt	0	Use Cartesian coordinates for optimization
	1	Use internally generated redundant internal coordinates for optimization (including any from **coord** or **connect** sections, if they exist)
	2	Use internal coordinates from input Z-matrix for optimization (note: if geometry input is in Cartesian format or contains a second bond angle rather than a torsional angle for any atom, **intopt** is reset to 1)
nooptr	***0***	Optimize all bond lengths not specifically constrained in **zmat** section
	1	Constrain (freeze) all bond lengths for optimization
noopta	***0***	Optimize all bond angles not specifically constrained in **zmat** section
	1	Constrain (freeze) all bond angles for optimization
nooptt	***0***	Optimize all torsional angles not specifically constrained in **zmat** section
	1	Constrain (freeze) all torsional angles for optimization
ip472	***0***	Do not save intermediate structures in the `.mae` output file.
	2	Save intermediate structures in the `.mae` output file.
props_each_step	***0***	Do not calculate properties at each step in a geometry optimization.
	1	Calculate specified properties at each step in a geometry optimization. The properties you can calculate are ESP, Mulliken, and stockholder charges, multipole moments, and Fukui indices. The relevant keywords for the properties must also be included in the **gen** section (**icfit**, **mulken**, **stockholder_q**, **ldips**, **fukui**).

Table 9.13. General geometry optimization keywords (Continued)

Keyword	Value	Description
needgwd	***0***	Do not compute DFT grid weight derivatives
	1	Compute DFT grid weight derivatives (and second derivatives for CPHF)
	2	Compute DFT grid weight derivatives and gradient from grid translation (symmetry will be turned off)
	3	Compute DFT grid weight derivatives and gradient from grid translation and rotation (symmetry will be turned off)
iaccg	***2***	Use default convergence criteria shown in Table 9.19
	3	Perform quicker, coarser calculation by multiplying convergence criteria shown in Table 9.19 by 5
	4	Solution-phase criteria; a factor of 3 times the criteria in Table 9.19
	5	Use tight criteria; a factor of 0.1 times the criteria in Table 9.19
geoconv_mode	standard	Use standard criteria for convergence, in which all five criteria must be met (energy difference, maximum gradient, rms gradient, maximum displacement, rms displacement).
	flexible	Assess convegence based on a score, with one point for meeting a criterion, an extra point if a gradient or displacement criterion is met to 0.2 of the threshold, an extra point if the energy change is less than 0.05 of threshold and another if it is less than 0.005 of threshold. The optimization is converged if the score is 5 or more points.
nops_opt_switch	>0	Specifies convergence cutoff for switching from pseudospectral to analytic integral evaluation in a geometry optimization. The value multiplies the standard convergence criterion, and a switch to `nops=1` is made when convergence is reached with the scaled convergence cutoff. The value must be greater than 1 for switching to take place. Not available with IRC or `check_min > 0`.
ngeorestart	≥0	Maximum number of automatic restarts used for a “stuck” geometry optimization. Restarting discards the Hessian history and sets `iacc=1`. Default is ***1***
nogdiis	0	Use GDIIS method (Geometry optimization by Direct Inversion in the Iterative Subspace) [193] to get new geometry
	1	Don’t use GDIIS method
ilagr	<0	Zero out gradients along frozen coordinates
	0 or 1	Project out gradient components in constrained coordinates
	>1	Apply constraints using Lagrange multipliers

Table 9.14 lists keywords for analyzing geometry convergence and checking whether a minimum has been obtained when requested. The checking feature is not supported with dynamic constraints, or for scans or IRC calculations.

Table 9.14. Geometry optimization keywords for analyzing and checking convergence

Keyword	Value	Description
check_min	*0*	Do not check if the stationary point is a minimum.
	1	Check for a minimum-energy structure by perturbing the initial converged geometry using the QM Hessian, which is calculated.
	2	Check for a minimum-energy structure by perturbing all rotatable proper and improper torsions (excluding those in rings and double bonds).
	3	Check for a minimum-energy structure by perturbing overlapping torsions and planar inversions (excluding those in rings and double bonds), and torsions involving potential internal hydrogen bonds in which the hydrogen atom (donor) is symmetrically placed between two acceptors.
check_min_update	*0*	Do not update the output `.mae` file until final convergence. The initial converged geometry is not replaced until all perturbations to check for a minimum have finished.
	1	Update the geometry in the `.mae` file after initial convergence, while performing optimizations on the perturbed geometries.
check_min_return	*0*	Do not check that the perturbations return to the initial minimum on optimization.
	1	Check that the perturbations return to the initial minimum on optimization. If optimization of a perturbed geometry does return to the initial minimum, report the return and exit from the optimization for the perturbation. The criteria for checking are given by the keywords **check_min_ergtol** and **check_min_geotol**.
check_min_eigcut	*-5.0*	Cutoff value for negative Hessian eigenvalues, in cm^{-1}. Eigenvalues above this cutoff do not count towards the number of negative eigenvalues used to perturb the geometry when **check_min**=1.
check_min_ergtol	*0.00008*	Energy tolerance in hartrees for determining if the geometry is returning back to the initial converged structure. If the energy difference between the perturbed geometry at any optimization step and the initial minimum is less than this amount, the geometry is considered to have returned to the initial structure.

Table 9.14. Geometry optimization keywords for analyzing and checking convergence

Keyword	Value	Description
check_min_geotol	***0.02***	Geometry tolerance in bohrs/radians for determining if the geometry is returning back to the initial converged structure. If the rms displacement between the perturbed geometry at any optimization step and the initial minimum is less than this amount, the geometry is considered to have returned to the initial structure.
check_min_pert	***0.6***	Magnitude of perturbation along the Hessian eigenvalue for **check_min**=1, in atom ic units.
check_min_target	0	Add perturbation values to the current angles for **check_min**=2 or 3. Default for **check_min**=2.
	1	Use the perturbation value as a target angle for **check_min**=2 or 3. Default for **check_min**=3.
check_min_p_pert	***20.0***	Perturbation in degrees for torsional perturbations. Interpreted as a target value or an increment, depending on **check_min_target**.
check_min_p_tol	***8.0***	Tolerance in degrees for checking whether to perturb a torsion for **check_min**=3. If the value of the torsion deviates from planarity by less than this amount, the torsion is perturbed, or for H-bond perturbations, if the two torsions differ by more than this amount.
check_min_amide	0	Do not perturb amide bonds when **check_min**=3.
	1	Perturb amide bonds when **check_min**=3.
check_min_hb_tar	0	Add perturbation values to H-bond dihedrals for **check_min**=3.
	1	Use the perturbation value as a target H-bond dihedral for **check_min**=3.
check_min_hb_lim	***90.0***	Upper limit for potential H-bond dihedral angles. If the H-bond dihedral angles are greater in absolute value than this amount, they are not considered potential H-bonds and are not perturbed.
optverdict	0	Do not run analysis of geometry convergence.
	1	Run analysis of geometry convergence and print report: 0: Monotonic convergence. Convergence OK. 1: Non-monotonic convergence. No erratic convergence was detected and the structure converged to an optimal structure. Convergence OK. 2: Erratic convergence but the optimization converged to an optimal structure. Convergence OK.

Table 9.14. Geometry optimization keywords for analyzing and checking convergence

Keyword	Value	Description
		3: Convergence to a non-optimal structure. The difference between the converged energy and the minimum energy is less than 0.1 kcal/mol. Convergence OK for solvation, OK for pseudospectral if the difference is less than 0.01 kcal/mol, but not otherwise. 4: Converged to a non-optimal structure, whose energy is above the minimum energy by more than 0.1 kcal/mol. Convergence not OK. No convergence category is written if the optimization ran out of iterations.
	2	Print report on convergence at each iteration.

Table 9.15 lists keywords specific to transition-state optimizations. Table 9.16 lists keywords that are used to specify the initial Hessian, control Hessian updating, and modify the Hessian when using it to update the geometry

Table 9.15. Keywords for transition-state optimizations

Keyword	Value	Description
iqst	***0***	Perform standard (non-QST) transition-state search
	1	Use quadratic synchronous transit (QST) methods to guide transition-state search. Sets **itrvec** to –5
qstinit	***0.5***	Distance of LST transition-state initial guess between reactant and product geometries. Range is 0.0 to 1.0.
ifollow	***0***	For each optimization iteration, select a new eigenvector to follow
	1	For each optimization iteration, follow eigenvector that most closely correlates with one followed previously
itrvec	***0***	Select lowest Hessian eigenvector as transition vector
	>0	Select eigenvector number **itrvec** as transition vector (see Section 4.3 on page 100). Sets **ifollow** to 1.
	–1	Select lowest non-torsional eigenvector as transition vector
	–2	Select lowest stretching eigenvector as transition vector
	–5	Select eigenvector which best represents reaction path

Table 9.16. Keywords for control of the Hessian

Keyword	Value	Description
inhess	–1	Use Fischer-Almlöf guess for Hessian
	0	Use Schlegel guess for Hessian (default only if no **hess** section exists)
	1	Use unit matrix for initial Hessian
	2	Use Cartesian input Hessian found in **hess** section (**inhess**=2 automatically if non-empty **hess** section exists)
	4	Compute and use quantum mechanical Hessian
nhesref	>0	Number of lowest-frequency Hessian eigenvectors used in Hessian refinement (default is ***0***)
pertnd	***0.05***	Displacement (in atomic units) used for Hessian refinement or calculations of numerical forces or frequencies.
irefhup	2	Refine initial Hessian using Powell updates [195]
	3	Refine initial Hessian using mixed Murtagh-Sargent/Powell updates [196]
	4	Refine initial Hessian using Murtagh-Sargent updates [197]
ihuptyp	0	Don't update Hessian
	1	Update Hessian each iteration using BFGS (Broyden-Fletcher-Goldfarb-Shanno) method [198] (default for minimum-energy structure optimizations)
	2	Update Hessian using Powell method [195]
	3	Update Hessian using mixed Murtagh-Sargent/Powell method [196] (default for transition-state optimizations)
	4	Update Hessian using Murtagh-Sargent method [197] (not recommended)
	–1	Compute quantum mechanical Hessian at each geometry; sets **inhess**=4
irfo	0	Before using Hessian to update geometry, modify it by sign-flipping or reverting to an older Hessian [194]
	1	Before using Hessian to update geometry, modify it by RFO (rational function optimization) level shifting [199]. Default for geometry optimizations that do not use dynamic constraints.
	2	Before using Hessian to update geometry, modify it by P-RFO (partitioned rational function optimization) level shifting [199]. Default for transition-state searches. Automatically set for geometry optimizations that use dynamic constraints.

In order to avoid changing the geometry too much because of an unusually shaped potential well or an inaccuracy in the Hessian, Jaguar restricts the norm of the changes to the Cartesian or internal coordinates to be less than a certain trust radius, which is defined in atomic units (bohrs, radians). The trust radius can vary from one iteration to another (**itradj**=1) or it can be fixed (**itradj**=0). Keywords controlling the use of a trust radius are listed in Table 9.17.

Table 9.17. Keywords for trust radius adjustment

Keyword	Value	Description
itradj	***0***	Use same trust radius throughout optimization (default for minimum-energy structure optimizations)
	1	Adjust trust radius using Culot/Fletcher heuristic [198, 200] (default for transition-state optimizations)
	–1	Adjust trust radius using Simons' cubic potential model [201] (not recommended with Jaguar)
itrcut	0	Apply trust radius by truncating Newton-Raphson step(s)
	1	Apply trust radius by level shifting of Hessian to reduce resultant step size
trust	***0.3***	Initial trust radius, in atomic units (bohr and/or radians): if norm of proposed displacements exceeds trust radius, step size is reduced as described by **itrcut** (default is ***0.3***, except for transition-state optimizations, when it is ***0.1***)
tradmx	***0.3***	Maximum trust radius allowed during optimization for **itradj**>0; see **trust** information (default is ***0.3,*** except for solvated cases, when it is ***0.1***)
tradmn	***0.01***	Minimum trust radius allowed during optimization for **itradj**>0; see **trust** information
tremx	***0.25***	Trust radius reduction criterion; if relative error between actual and predicted energy changes is more than **tremx** and **itradj**>0, trust radius is reduced
trgmx	***0.0***	Trust radius reduction criterion; for **itradj**>0 and **trgmx**>0, if absolute error in a component of predicted gradient exceeds **trgmx** hartrees/bohr, trust radius is reduced
treok	***0.2***	Criterion for increasing trust radius; if **itradj**=2 and relative error between actual and predicted energy changes is less than **treok**, trust radius is increased (**treok** default is ***0.2***)
trescal	***2.0***	Scale factor for trust radius adjustment; used only when **itradj**=2

If the trust radius is fixed, it remains the same throughout the optimization except when Jaguar determines that changing it will lead to better convergence for problem jobs. This setting is the default for optimizations to minimum-energy structures. If the trust radius is allowed to vary (the default for transition-state optimizations), Jaguar keeps geometry changes within the region that is well-described by the Hessian by increasing the trust radius when the Hessian is correctly predicting energy changes and decreasing it when the predictions are inaccurate.

Prevention of chemical reactions during a geometry optimization can be done with the **stop_rxn** keyword. This keywords turns on checking of the connectivity after each geometry step. If bonds have been broken or formed, the optimization is restarted with additional repulsive or harmonic restraint potentials to prevent the bonding change, based on the value of **stop_rxn**. The repulsive potential for keeping atoms apart has the form

$$V_{repulsive} = \frac{\varepsilon e^{-\sigma r^2}}{r^2}$$

while the potential for keeping atoms together is a flat-bottomed harmonic potential,

$$V_{restrain} = \begin{array}{ll} \frac{1}{2}k(r-r_0-a)^2, & r > r_0 + a \\ 0, & r_0 + a > r > r_0 - a \\ \frac{1}{2}k(r-r_0+a)^2, & r < r_0 - a \end{array}$$

where r is the distance between the two atoms whose bonding change is to be prevented, r_0 is the initial distance, and a is the half-width of the flat bottom. The parameters can be set by keywords, which are described in Table 9.18. The energies from these added terms are not included in the reported total energies, but are used, along with the forces from these terms, for optimization purposes. As the added energies are artificial and are often added to a region of the potential surface that is quite steep, the results can be sensitive to the parameters used and to other circumstances of the calculations.

Table 9.18. Keywords for prevention of chemical reactions

Keyword	Value	Description
stop_rxn	***0***	Do not prevent chemical reactions during geometry optimizations.
	1	Add repulsive terms if bonds are formed and harmonic constraints if bonds are broken.
	2	Add repulsive terms if bonds are formed only
	3	Add harmonic constraints term if bonds are broken only.

Table 9.18. Keywords for prevention of chemical reactions

Keyword	Value	Description
stop_rxn_epsilon	***35.0***	Maximum strength of repulsive potential ε in kcal mol^{-1} Å^2.
stop_rxn_sigma	***0.15***	Width parameter for repulsive potential σ in Å^{-2}.
stop_rxn_kval	***1000.0***	Force constant for harmonic restraint *k* in kcal mol^{-1} Å^{-2}.
stop_rxn_offset	***0.0***	Width of flat bottom *a* in Å.

The keywords shown in Table 9.19 may be used to specify the geometry convergence criteria. The values may be scaled to five times their default values with the keyword setting **iaccg**=3 for a quicker, coarser calculation, or to a tenth of the default values for tighter convergence with **iaccg**=5. The first four keywords in Table 9.19 have units of hartrees/bohr, **gconv5** and **gconv6** have units of bohrs, and **gconv7** has units of hartrees.

Table 9.19. Geometry convergence criteria keywords

Keyword	Default value	Convergence Criterion For
gconv1	4.5×10^{-4}	Maximum element of gradient
gconv2	3.0×10^{-4}	rms of gradient elements
gconv5	1.8×10^{-3}	Maximum element of nuclear displacement
gconv6	1.2×10^{-3}	rms of nuclear displacement elements
gconv7	5.0×10^{-5}	Difference between final energies from previous and current geometry optimization iterations

SCF calculations performed for each new structure generated during an optimization are judged to be converged when they meet the criterion for the root mean square of the change in density matrix elements, which is controlled by the keyword **dconv**; the usual SCF energy convergence criterion (**econv**) is ignored for optimizations.

If you want to save the structure at each step of a geometry optimization in a Maestro-formatted file, set **ip472**=2. You can also extract the structures from the output file to a Maestro file with the command

```
$SCHRODINGER/utilities/jagconvert -ijout jobname.out -omultimae filename.mae
```

If you import these structures into Maestro, you should set the color scheme manually: it is not set correctly on import.

To calculate certain properties (charges, multipole moments, Fukui indices) at each step in a geometry optimization, set **props_each_step**=1. This allows you to examine these properties during the course of a geometry optimization for points at which they change significantly.

9.5.11 Geometry Scan Keywords

Keywords for defining the initial wave function, geometry, scan direction, constraint violation, and Hessian for geometry scans (coordinate scans) are listed in Table 9.20. These keywords can also be used with IRC scans.

Table 9.20. Keywords for geometry scans

Keyword	Value	Description
scanhess	*0*	Use the final Hessian from the last geometry step as the initial Hessian for the next step.
	1	Use the Hessian specified by the **inhess** setting as the initial Hessian for the next step. The Hessian in a **hess** section is only used for the first step unless **inhess** is explicitly set to 2.
scangeo	*0*	Apply scan increments to the final geometry of the previous scan step.
	1	Apply scan increments to the input geometry. Automatically set for scans of bond angles that are near-linear or pass through180° or 0°.
	2	Apply scan increments to the initial geometry of the previous scan step (before symmetrization).
scanguess	*0*	For geometry scans, use converged wave function from previous step as initial guess for current geometry
	1	For each step in a geometry scan, generate the initial guess wave function according to the **iguess** setting
scan_independent	*0*	Scans move the last atom in the scan coordinate from the initial value to the final value. This defines the direction of the scan.
	1	Scans move the last atom in the scan coordinate halfway from the initial value to the final value, then move the first atom the rest of the way to the final value. If one of the directions violates constraints or interferes with other scan coordinates, the other is used for the full range; if neither is possible, the scan fails. Thus the scan can occur in either direction, i.e. is independent of the defined direction.
scan_nocheck	*0*	Check scan coordinates to ensure that no constraint violations occur during the scan, that scan coordinates are not in a ring or are identical to another scan coordinate. If a problem is found, the scan fails.
	1	Turn off scan coordinate checking.

9.5.12 Intrinsic Reaction Coordinate (IRC) Keywords

IRC scans have been implemented using the methods described in Ref. [235]. The implementation includes IRC and minimum energy path (MEP) calculations. The calculations start at a transition state and move downhill in energy along the reaction path toward a minimum of the potential energy surface. They are mainly used to check that the given transition state is indeed the expected transition state for the reaction of interest. The keywords for IRC and MEP calculations are listed in Table 9.21. You can also use the keywords listed in Table 9.20.

Table 9.21. Keywords for IRC calculations

Keyword	Value	Description
irc	***0***	Do not do IRC calculation
	1	Do IRC calculation with non-mass-weighted coordinates (minimum energy path scan)
	2	Do IRC with mass-weighted coordinates
ircmode	forward	Find IRC points in "forward" direction from the transition state
	reverse	Find IRC points in "reverse" direction from the transition state
	downhill	Find IRC points by moving downhill from an initial geometry that is not a transition state
	both	Find IRC points in both "forward" and "reverse" directions from the transition state
three_pt_irc	***0***	Do not do three-point IRC calculation
	1	Do three-point IRC calculation. The first point on each side of the transition state is located as usual, then the constraint on the distance from the transition state is relaxed and the points are optimized to find the end points, corresponding to the reactant and product complexes. Requires **irc** to be set as well.
ircmax	***6***	Maximum number of IRC points to be found in any direction. Must be a positive integer.
ircmxcyc	***30***	Maximum number of geometry iterations used to calculate each IRC point. Must be a positive integer.
ircstep	***0.1***	Step size taken for each IRC point, in bohrs amu$^{-1/2}$ or radians amu$^{-1/2}$
ircgcut	***1.0***	Scale factor for the cutoff that is used to determine whether the gradient is too small to locate the next IRC point. Use values that are less than 1.
ip472	0	Do not save the IRC structures in the `.mae` output file.
	2	Save the IRC structures in the `.mae` output file and write the reaction coordinate value as a property.

The “forward” and “reverse” directions are defined as follows. The first set of conditions that constitutes a valid definition is used.

1. If two additional geometries are entered in the **zmat2** and **zmat3** sections, they are assumed to be the geometries for the reactant (in **zmat2**) and product (in **zmat3**). The forward direction is defined as moving from reactant to product.
2. If a vector is entered in the **tvec** section, it defines the forward direction. For example,

   ```
   &tvec
   C2 H3 * 0.5
   O1 C2 H3 * -1.0
   &
   ```

   This definition produces a composite coordinate that is the sum of 0.5 times the distance between atoms C2 and H3 and –1.0 times the angle O1-C2-H3.

   Coordinates comprising this composite can be any combination of bond stretches (2 atoms listed), angle bends (3 atoms), and dihedral angles or torsions (4 atoms). Atom labels or index numbers for the atoms can be used in specifying atoms. Coordinate coefficients, specified by including an asterisk followed by a value after the last atom are optional. The default coefficient value is 1.0. The forward direction is the direction that makes the composite coordinate defined in this section larger.
3. The Hessian eigenvector for the imaginary frequency mode with the most negative eigenvalue of the Hessian is used to define the forward direction. The phase of the eigenvector is chosen so that the largest coefficient is positive, and the forward direction is the direction that increases the coordinate for the largest coefficient.

IRC calculations can be done in either Cartesian coordinates (specified with **intopt**=0), or redundant internal coordinates (**intopt**=1), which is the default.

IRC in any mode but “downhill” requires a Hessian, which must either be entered in the **hess** section, or calculated analytically with **inhess**=4 in the **gen** section. Initial guess Hessians are not useful, as they do not have any imaginary frequencies. If a Hessian is entered in the **hess** section (whether directly or from a restart file for a calculation that performed a Hessian evaluation) and symmetry is on, the IRC calculation might not produce any points or might not produce points on the actual reaction path if the transition state has higher symmetry than the reaction path. If this is the case, you should turn symmetry off (**isymm**=0 in the **gen** section). If you evaluate the Hessian with **inhess**=4 in the **gen** section, symmetry is turned off for analytic Hessian calculations, and the subsequent IRC calculations are done without symmetry.

The IRC calculation can fail if the step size is too small. The warning message states that the vector used to determine the step is too small. You can increase the step size by setting **ircstep**.

The IRC calculation can also fail if the potential energy surface is very flat, and the gradient that is used to find the next IRC point is too small. You can reduce the magnitude of the cutoff for determining when the gradient is too small with the keyword **ircgcut**, which should be set to a value less than 1.0. If you set this keyword, you should be careful not to set it so small that it does not filter out “noise” in the gradient.

The restart file for an IRC job includes the geometry of the last found IRC point in the **zmat** section. An **ircmode**=downhill setting is included in the **gen** section regardless of the initial setting, as a restart job proceeds downhill from the last found IRC point. If the job has not gone far enough to have found another IRC point, no **ircmode**=downhill setting is included.

9.5.13 Solvation Keywords

Most of the solvation keywords correspond to GUI options described in Section 3.8 on page 58. The allowed values for the solvation keywords are described in Table 9.22. Defaults for some of these keywords are not indicated in bold italics, since the keywords’ default values generally depend on other keywords. (By default, Jaguar calculations are performed in the gas phase, so **isolv**=0 and all other solvation keywords are irrelevant.) The default values for the real-valued parameters are for water. The solvent can be specified with the **solvent** keyword instead of the dielectric constant and probe radius; this keyword sets these two quantities.

For solvated geometry optimizations, the **trust** keyword, which is described in Section 9.5.10 on page 217, has a default value of 0.1 instead of its usual default of 0.3.

If you set **isolv**=2 to use the Poisson-Boltzmann solver and use water as the solvent, you have the choice of two methods of defining the cavity. The default is to use the van der Waals radii, as defined in the `pbf_radii.ark` file (or in the old `default.lewis` file, if **pbf_radii**=0). The alternative is to use an isosurface of the density: that is, the cavity is the surface at a specified, small value of the electron density. To use this method, set **pbf_isodens**=1. This method removes the need to define fixed van der Waals radii for each atom type, and automatically adjusts to changes in the electron density (as a real solvent would). It includes an explicit solvent correction factor based on the assumption that the interaction with the solvent is a function of the electric field on the molecular surface [250]. When you use this method, you should include diffuse functions in the basis set, which are important for polar groups and for anions. This method is particularly recommended for computing the solvation energy of anions.

Note: The explicit correction is known to overestimate the interaction with the solvent for systems containing nitro groups or multiple fluorine atoms.

Several keywords are available for the control of the finite-element mesh used by the Poisson-Boltzmann solver. To increase the resolution, you can increase the values of **pbf_nshells**, **pbf_molsurf_idx**, **pbf_protosph_res**, and **pbf_boundary_res** and decrease the values of **pbf_fem_mxres_cg**.

Table 9.22. Keywords for solvation calculations

Keyword	Value	Description
isolv	***0***	Do not perform a solvation calculation
	2	Perform a solvation calculation using a Poisson-Boltzmann solver
	5	Perform a solvation calculation using the SM6 model, which applies only to water. This is the only solvation keyword that should be specified when performing SM6 calculations.
	6	Perform a solvation calculation using the SM8 model. Can be used with the **solvent** keyword for any of the recognized solvents; other keywords are not relevant.
solvent	string	Solvent name: `water`, `benzene`, `chlorobenzene`, `cyclohexane`, `carbon_tet` (carbon tetrachloride), `dichloroethane` (1,2-dichloroethane), `methanol`, `nitrobenzene`, `dmso` (dimethyl sulfoxide), `chloroform`, `dichloromethane`, `dimethylformamide`, `tetrahydrofuran`.
icavity	0	Do not include solute cavity energy term in solvation calculation
	1	Include solute cavity energy term (default for water)
	2	Force calculation of cavity energy term
isurf	0	Do not include the first shell correction factor term in the solvation energy
	1	Include the first shell correction factor term in the solvation energy (default for most calculations in water; turns on **ivanset**=1 by default). Ignored if **pbf_isodens**=1.
ivanset	0	Do not set van der Waals radii according to Lewis structure
	1	Set van der Waals radii according to the Lewis structure specified by **lewstr** (1 by default); see Section 9.5.6 on page 206. Only used if **pbf_isodens**=0 and **pbf_radii**=0.
kesep	0	Combine terms for all one-electron matrices
	1	Keep kinetic energy terms, nuclear attraction integrals, and point charge terms separate (Note: if **isolv**=1 or 2, **kesep**=1 by default)
isolvg	0	Compute gradients in solvent with method used in Jaguar 3.5 and earlier.
	1	Compute gradients in solvent with robust Jaguar 4.0 method.
	2	Compute gradients in solvent with improved Jaguar 7.8 method.
pbf_nshells	***9***	Number of shells around each atom in the finite-element mesh.

Table 9.22. Keywords for solvation calculations (Continued)

Keyword	Value	Description
pbf_molsurf_idx	*4*	The index of the shell to use for the molecular surface.
pbf_protosph_res	*21*	The Lebedev angular scheme (see Table 10.1 on page 307) on the prototype atomic sphere.
pbf_boundary_res	*14*	The Lebedev angular scheme (see Table 10.1 on page 307) at the dielectric boundary.
pbf_fem_mxres_cg	*10^{-4}*	The maximum residual for the charge scaling solver.
pbf_isodens	*0*	Do not use the density isosurface to construct the PBF grid.
	1	Use the density isosurface to construct the PBF grid, rather than using preset van der Waals radii. This method is not available with ECPs, and can only be used for water. If you set this keyword, **pbf_radii**, **ivanset**, and **isurf** are ignored.
pbf_molsurf_iso	*0.001*	Value of the density for the isosurface.
pbf_boundary_iso	*10^{-50}*	Value of the isodensity on the molecular boundary.
pbf_radii	0	Use the `default.lewis` file to define the van der Waals radii.
	1	Use the `pbf_radii.ark` file to define the van der Waals radii, and determine the connectivity from the Maestro file.
	2	Use the `pbf_radii.ark` file to define the van der Waals radii, and determine the connectivity from the coordinates in the Jaguar input file.
res_inner	*8.0*	Factor controlling the resolution of the inner surfaces.
res_outer	*16.0*	Factor controlling the resolution of the outer surfaces, excluding boundary.
res_boundary	*0.2*	Resolution on the boundary shell.
res_molsurf	*2.2*	Resolution on the molecular surface.
isotrymax	*30*	Maximum number of attempts to reduce an inner density that is too large.
epsout	*80.37*	Outer dielectric constant of solvent
epsin	*1.0*	Inner dielectric constant of solvent [230]
radprb	*1.40*	Radius of solvent probe molecule
sconv	*1.5×10^{-4}*	Solvation energy convergence criterion in Hartrees
esolv0	number	Gas phase energy of molecule, in Hartrees; used in some restart (new input) files for solvation jobs

The SM8 model accepts a large range of solvents. The solvent name values that can be used with the **solvent** keyword for SM8 are listed below.

111trichloroethane	bromoethane	methylbenzoate
112trichloroethane	bromooctane	methylethanoate
12dichloroethane	butanal	methylmethanoate
124trimethylbenzene	butanoicacid	methylphenylketone
14dioxane	butanone	methylpropanoate
1bromo2methylpropane	butanonitrile	methylbutanoate
1bromopentane	butylethanoate	methylcyclohexane
1bromopropane	butylamine	methylformamide
1butanol	butylbenzene	mxylene
1chloropentane	carbondisulfide	heptane
1chloropropane	carbon_tet	hexadecane
1decanol	chlorobenzene	hexane
1fluorooctane	chlorotoluene	nitrobenzene
1heptanol	cis12dimethylcyclohexane	nitroethane
1hexanol	decalin	nitromethane
1hexene	cyclohexane	methylaniline
1hexyne	cyclohexanone	nonane
1iodobutane	cyclopentane	octane
1iodopentene	cyclopentanol	pentane
1iodopropane	cyclopentanone	ochlorotoluene
1nitropropane	decane	ocresol
1nonanol	dibromomethane	odichlorobenzene
1octanol	dibutylether	onitrotoluene
1pentanol	dichloromethane	oxylene
1pentene	diethylether	pentadecane
1pentyne	diethylsulfide	pentanal
1propanol	diethylamine	pentanoicacid
222trifluoroethanol	diiodomethane	pentylethanoate
224trimethylpentane	dimethyldisulfide	pentylamine
24dimethylpentane	dimethylacetamide	perfluorobenzene
24dimethylpyridine	dimethylformamide	phenylether
26dimethylpyridine	dimethylpyridine	propanal
2bromopropane	dmso	propanoicacid

2chlorobutane	dipropylamine	propanonitrile
2heptanone	dodecane	propylethanoate
2hexanone	e12dichloroethene	propylamine
2methylpentane	e2pentene	pxylene
2methylpyridine	ethanethiol	pyridine
2nitropropane	ethanol	pyrrolidine
2octanone	ethylethanoate	secbutanol
2pentanone	ethylmethanoate	tbutanol
2propanol	ethylphenylether	tbutylbenzene
2propen1ol	ethylbenzene	tetrachloroethene
3methylpyridine	ethyleneglycol	tetrahydrofuran
3pentanone	fluorobenzene	tetrahydrothiophenedioxide
4heptanone	formamide	tetralin
4methyl2pentanone	formicacid	thiophene
4methylpyridine	hexadecyliodide	thiophenol
5nonanone	hexanoic	toluene
aceticacid	iodobenzene	transdecalin
acetone	iodoethane	tribromomethane
acetonitrile	iodomethane	tributylphosphate
aniline	isobutanol	trichloroethene
anisole	isopropylether	Trichloromethane or chloroform
benzaldehyde	isopropylbenzene	triethylamine
benzene	isopropyltoluene	undecane
benzonitrile	mcresol	water
benzylalcohol	mesitylene	z12dichloroethene
bromobenzene	methanol	

Additional solvents for SM8 can be specified by means of several descriptors, for which the keywords are listed in Table 9.23. To use a custom solvent, the dielectric constant, index of refraction, acidity and basicity can be derived from experiment, or from interpolation or extrapolation of data available for other solvents, and the fraction of aromatic and halogen atoms can be computed from the chemical formula of the solvent. Solvent parameters for common organic solvents are tabulated in the Minnesota Solvent Descriptor Database, available at: http://comp.chem.umn.edu/solvation/mnsddb.pdf.

Table 9.23. Keywords for SM8 custom solvents.

Keyword	Description
epsout	Dielectric constant ε
sm8_sola	Abraham's hydrogen bond acidity α, also called $\Sigma\alpha_2{}^H$
sm8_solb	Abraham's hydrogen bond acidity β, also called $\Sigma\beta_2{}^H$
sm8_solc	Aromaticity: the fraction of non-hydrogen solvent atoms that are aromatic carbon atoms
sm8_solg	Macroscopic surface tension at the air/solvent interface at 298 K, in cal mol^{-1} $Å^{-2}$
sm8_solh	Electronegative halogenicity: fraction of non-hydrogen solvent atoms that are F, Cl, Br
sm8_soln	index of refraction at optical frequencies at 293 K, $n \equiv n_{20}^{D}$

9.5.14 pKa Keywords

Keywords for controlling aspects of pK_a calculations are listed in Table 9.24.

Table 9.24. Keywords for pKa calculations

Keyword	Value	Description
ipkat	*atom*	Specify the atom for which the pK_a is calculated, using either the atom number or the atom label. The atom should be the acidic hydrogen atom in an acid, or the basic atom in a base.
ipkasites	*list*	Specify multiple atoms for which to calculate the pK_a. The list is a comma-separated list of atom numbers or atom labels (or a mix), with no spaces. The atoms should be the acidic hydrogen atoms in acidic groups, or the basic atoms in basic groups.
ipkasearch	*list*	Search for pK_a atoms in the given list of atoms. The list is a comma-separated list of atom numbers or atom labels (or a mix), with no spaces, or it can be `all`, meaning the entire molecule. A pK_a is calculated for each atom found.
ipka_prot_deprot	***both***	When searching for sites, look for both protonation and deprotonation sites.
	`protonation`	When searching for sites, look for protonation sites only.
	`deprotonation`	When searching for sites, look for deprotonation sites only.
ipka_lower_cut	***2.0***	Specify cutoff for low pKa values. When searching for sites, any atom for which all of the training set pKa values for the functional group are lower than the cutoff is not included in the list.

Table 9.24. Keywords for pKa calculations (Continued)

Keyword	Value	Description
ipka_upper_cut	*12.0*	Specify cutoff for high pKa values. When searching for sites, any atom for which all of the training set pKa values for the functional group are higher than the cutoff is not included in the list.
ipka_tier_cutoff	*1*	Specify the tier in the shell model for which functional groups are excluded the search. All groups in and above this tier are excluded. Tier 0 is the specific groups, tier 1 is the generic groups, tier 2 is everything.
ipkaraw	0	Calculate corrected pK_a values.
		Calculate raw pK_a values with no empirical correction. These can be used for fitting to obtain new corrections.
ipka_max_conf	*1*	Specify the maximum number of conformers to use in the pK_a calculations. When multiple conformers are specified, a macroscopic pK_a is calculated.
ipka_erg_window	*12.0*	Discard conformers if their energy relative to the lowest energy conformer is higher than the specified value, in kcal/mol.
ipkaverbose		Print more output for macroscopic pK_a calculations, including intermediate p$K_a(i,j)$ values
ipka_csrch_acc	*0*	Use the standard accuracy parameters for MacroModel conformational searches (100 steps, 500 minimization iterations).
	1	Use higher accuracy parameters for MacroModel conformational searches (3000 steps, 2500 minimization iterations). This is useful for reducing or eliminating dependence on input structures.
ipka_solv_opt	*0*	Run Jaguar optimizations in the gas phase.
	1	Run Jaguar optimizations in solution.

9.5.15 Properties Keywords

Keywords to request calculation of molecular properties, including multipole moments and charge fitting properties are listed in Table 9.25, most of which correspond to GUI options described in Section 3.9 on page 62. Only the values listed in the table are allowed

Analytic polarizabilities α and hyperpolarizabilities β and γ are available for HF, UHF, DFT, and UDFT methods. The definition of β changed with Jaguar 5.5, and differs by a factor of

0.5 from that used in previous versions of Jaguar. The new definition is now consistent with that used in GAUSSIAN. The definitions of polarizabilities α, first hyperpolarizabilities β, and second hyperpolarizabilities γ, are:

$$\alpha_{ij} = -\frac{d^2E}{dF_i dF_j} \qquad \beta_{ijk} = \frac{d^3E}{dF_i dF_j dF_k} \qquad \gamma_{ijkl} = \frac{d^4E}{dF_i dF_j dF_k dF_l}$$

If you want to calculate polarizabilities with the old definition, you must set **iopt332**=332 in the **gen** section, and you can only calculate α and β for closed-shell wave functions.

The finite field methods corresponding to **ipolar** > 0 differ in the data they use for numerical differentiation. The 3-point method uses the results from seven SCF calculations: one with no field, one with a field of E (whose input is described below) in the *x* direction, one with a field of –E in the x direction, and four others with fields of +E and –E in the *y* and *z* directions. The 5-point method uses the same data as the 3-point method, as well as data from SCF calculations using fields of +aE and –aE in the *x*, *y*, and *z* directions, where a is some constant. Similarly, the 7-point method uses the same data as the 3-point method, plus data obtained using fields of +aE, –aE, +bE, and –bE in the *x*, *y*, and *z* directions, where a and b are some constants. By default the magnitude of the electric field E is 0.024 au. If you want to use a different value, set the **efield** keyword to the desired value.

Table 9.25. Keywords for molecular property calculations

Keyword	Value	Description
icfit	***0***	Do not do electrostatic potential fitting
	1	Fit electrostatic potential to atomic centers (default for PBF solvation calculations)
	2	Fit electrostatic potential to atomic centers and bond midpoints
cfiterr	1.0×10^{-6}	Allowed error in electrostatic potential charge fitting when fitting is constrained to reproduce multipole moments
wispc	***0.75***	Spacing in bohrs of rectangular grid for electrostatic potential fitting
incdip	***0***	Use only total charge as constraint in electrostatic potential fitting
	1	Use charge and dipole moment as constraints in electrostatic potential (ESP) fitting
	11	Use charge, dipole moment, and quadrupole moment as constraints in electrostatic potential (ESP) fitting
	111	Use charge, dipole moment, quadrupole moment, and octupole moment as constraints in electrostatic potential (ESP) fitting

Table 9.25. Keywords for molecular property calculations (Continued)

Keyword	Value	Description
	ijk	Compute ESP fitted charges using total charge as a constraint, also constraining to dipole moment if $k=1$, to quadrupole moment if $j=1$, and to octupole moment if $i=1$
	–1	Do all **incdip** options sequentially
ldips	***1***	Do not calculate any multipole moments
	2	Calculate dipole moments
	3	Calculate dipole and quadrupole moments
	4	Calculate dipole, quadrupole, and octupole moments
	5	Calculate dipole, quadrupole, octupole, and hexadecapole moments
ipolar	***0***	Do not calculate polarizabilities or hyperpolarizabilities
	–2	Use CPHF to calculate polarizabilities α and first and second hyperpolarizabilities β and γ
	–1	Use CPHF to calculate polarizabilities α and hyperpolarizabilities β
	1	Calculate α using 3-point finite field method
	2	Calculate α and β using 3-point finite field method
	5	Calculate α and β using 5-point finite field method
	7	Calculate α and β using 7-point finite field method
efield	***0.024***	Electric field for polarizability and hyperpolarizability calculations, in au (default is 0.006 for **ipolar**=1)
ldens	***0***	Do not calculate electron density
	1	Calculate electron density on grid (grid choice set by grid keyword **geldens**; ultrafine grid used by default)
denspc	***0.75***	Spacing in bohrs of rectangular grid for electron density calculation
mulken	***0***	Do not calculate Mulliken populations
	1	Calculate Mulliken populations by atom
	2	Calculate Mulliken populations by basis function and by atom
	3	Calculate Mulliken bond populations
nmr	***0***	Do not calculate NMR shielding constants
	1	Calculate NMR shielding constants, in ppm

Table 9.25. Keywords for molecular property calculations (Continued)

Keyword	Value	Description
fukui	***0***	Do not calculate atomic Fukui indices.
	1	Calculate atomic Fukui indices [238, 239].
stockholder_q	***0***	Do not calculate stockholder charges by Hirshfeld partitioning.
	1	Calculate stockholder charges by Hirshfeld partitioning [237].
hirshfeld_basis	***6-31G***	Specify basis used for promolecule in Hirshfeld partitioning of electron density for stockholder charge calculation.
espanal	***0***	Do not perform statistical analysis of the electrostatic potential
	1	Perform statistical analysis of the electrostatic potential on isodensity surface, including the electrostatic potential at the nuclei (sets **iplotden**=1 and **iplotesp**=1, also **epn**=1).
den_isoval	***0.001***	Isovalue, in electrons/bohr^3, for creating the isodensity surface on which the ESP or the ALIE analysis is performed.
epn	***0***	Do not calculate the electrostatic potential at the nuclei.
	1	Calculate the electrostatic potential at the nuclei (EPN). This property is stored in the Maestro file as an atom property. Values at ECP centers are erroneous due to the missing core density, and should be ignored.
ielec	***0***	Do not calculate the electrostatic potential and the electric field at a set of grid points.
	1	Calculate the electrostatic potential and the electric field at a set of grid points. The source of the grid must be specified with the **negrid** keyword. The values are written to the file *jobname*`.efld`.
negrid	–1	Read grid points for calculation of the electrostatic potential and the electric field from the **egridpt** section of the input file.
	–6	Read grid points for calculation of the electrostatic potential and the electric field from the file specified by `GPTSFILE`.
nbo	***0***	Do not run an NBO analysis at the end of the calculation.
	1	Run an NBO analysis at the end of the calculation, with NBO keywords `NRT` and `PRINT=3` set.

Jaguar prints polarizabilities and hyperpolarizabilities in atomic units. The atomic unit of polarizability is $e^2 a_0^2 E_h^{-1}$, where e is the charge on an electron, a_0 is the bohr radius, and E_h is the atomic unit of energy, or hartree. To convert to SI units, C^{2}m^{2}J^{-1}, multiply by 1.6488×10^{-41}. The first-order hyperpolarizability β is in atomic units of $e^3 a_0^3 E_h^{-2}$, and the second-order hyperpolarizability γ is in atomic units of $e^4 a_0^4 E_h^{-3}$. The conversion factors to SI units are 3.2064×10^{-53} and 6.2354×10^{-65}, respectively.

All polarizability methods are run with symmetry off: the keyword **isymm** is set to 0 automatically if **ipolar** ≠ 0. In all polarizability calculations, the keyword **econv**, which gives the energy convergence criterion, is set by default to 1.0×10^{-6} (although if the calculation first satisfies the criterion dictated by the keyword **dconv**, the energy convergence criterion is ignored).

When charge fitting is constrained to reproduce multipole moments (that is, when **incdip**>0), the keyword **cfiterr** determines whether the multipole moment constraint is too restrictive to produce adequate charges: if the error in the total resultant charges is more than **cfiterr**, the charge fitting is rerun with a lower multipole moment constraint. The keyword **wispc** is used to set the spacing of the rectangular grid for electrostatic potential fitting when the grid keyword **gcharge**=–2. Similarly, the keyword **denspc** is used to set the spacing of the electron density rectangular grid when **ldens**=1 and the grid keyword **geldens**=–3. The **efield** keyword allows you to specify an electric field for finite field polarizability and hyperpolarizability calculations. The default value shown in Table 9.25 applies when **ipolar** > 1. For **ipolar**=1 (3-point, polarizability-only calculations), the default value is 0.006 au.

If you want to print out the electrostatic potential at grid points that you specify, add the keyword settings **gcharge**=–6 and **ip172**=2 to the **gen** section of your input file. The **gcharge**=–6 setting instructs Jaguar to use the grid points and weighting factors in a file whose name and location are specified by the `GPTSFILE` line in the input file (see Section 9.1 on page 195). The **ip172**=2 setting instructs Jaguar to write out a file named *jobname*`.resp` containing the electrostatic potential data (see the text under Table 9.34).

If you want to print out the electric field as well as the electrostatic potential at a specified set of grid points, set **ielec**=1 in the gen section. You then have to specify the source of the points, which can come from an external file or from the input file. To specify an external file, add a `GPTSFILE` line to the input file (see Section 9.1 on page 195) and set **negrid**=–6 in the **gen** section. To specify points in the input file, add an **egridpt** section and set **negrid**=–1 in the **gen** section. For both source types, the points must be specified as Cartesian coordinates in angstroms, space separated, with one point per line. (Any grid points that coincide with an atom are shifted by 0.001 bohr in each coordinate, to avoid infinities.) The output is written to a file named *jobname*`.efld`. The number of points is given on the first line, and the remaining lines contain the results. Each of these lines has the Cartesian coordinates first, followed by the electrostatic potential, and then the Cartesian electric field components. The electrostatic potential and the electric field are given in atomic units.

To print out the electron density at a specified set of grid points, set **ldens**=1 and **geldens**=–6 in the gen section, and add a `GPTSFILE` line to the input file that specifies the file containing the grid points. This file should list the Cartesian coordinates in angstroms and a weight factor, which should be zero, with one grid point per line. The output is in the file *jobname*`.chdens`.

Gas-phase NMR shielding constants in ppm are available for closed-shell and unrestricted open-shell wave functions. Chemical shifts are calculated for ^{13}C and ^{1}H from the shieldings, using a linear fit to experiment for selected basis sets and density functionals (see Section 3.9.6 on page 67). To calculate chemical shifts directly, you should calculate NMR shielding constants for the reference molecules for each element of interest, in the same basis set and with the same method as for the molecule of interest.

By default, shieldings are calculated for all atoms, including those with ECPs. Shielding constants for atoms whose core is represented by an ECP should be treated with caution, because the main contributions come from the core tail of the valence orbitals, which is largely absent at ECP centers. Chemical shifts derived from these shielding constants might display the correct trends, but are likely to have the wrong magnitude.

If you set **espanal**=1, a statistical analysis of the electrostatic potential on the molecular surface is performed [242]. This analysis is also automatically performed if you request a plot of the electrostatic potential (**iplotesp**=1, see Section 9.5.29 on page 267). The molecular surface is represented by an isosurface of the electron density, which by default is created using an isovalue of 0.001 au, as in the original publication. You can change the isovalue using the **den_isoval** keyword. See Section 6.3.7.7 on page 156 for information on the properties generated. The minimum, maximum, variances, balance property and local polarity property are written to the output Maestro file. In addition, the maximum and minimum ESP values for each atom are written to the output Maestro file and can be used to label the atoms in the Workspace. A similar analysis is automatically performed for average local ionization potential (ALIE) surfaces (**iplotalie**=1).

If you want to track properties such as charges, multipole moments, or Fukui indices during a geometry optimization, you can set **props_each_step**=1 in the gen section.

9.5.16 Frequency-Related Keywords

For jobs that include a calculation of vibrational frequencies, various frequency-related properties can also be computed by setting the appropriate keywords. Most of these keywords, which are listed in Table 9.26, correspond to GUI options described in Section 3.10 on page 69. Only the values listed in the table are allowed.

The thermochemical properties are listed in cal/mol K and kcal/mol by default. Use the output option **eunit**=2 for output in J/mol K and kJ/mol.

When the calculation of vibrational frequencies is requested with **ifreq**=1 for Hartree-Fock or DFT calculations, intensities for the IR-active vibrational modes are automatically calculated (**irder** is set to 1 automatically). For any other level of theory, you must explicitly set **irder**=1, and the derivatives must be calculated numerically by setting **nmder**=2. The calculation of IR intensities involves the calculation of the dipole moment derivatives. Raman activities and intensities can be calculated by setting **iraman**=1. They involve the calculation of polarizability derivatives.

If you only want to calculate dipole moment derivatives using the Hartree-Fock method but don't want to do the frequency calculation that is normally required to get them, you must set up a special **path** section (see Section 9.19 on page 288) with the appropriate sequence of executables to run.

The **path** section to use is:

```
&path pre onee hfig probe grid rwr scf ira rwr irb &
```

You must also set **irder**=1, **isymm**=0, and **ifreq**=1. The **ifreq** setting is necessary to force tight accuracy in the SCF, but no frequency calculation is actually performed.

To compute partial frequencies for a fragment, you must first define the fragments in an **atomic** section, then make the setting **freqfrag**=*fragno* in the **gen** section for the frequency calculation. These settings are in addition to any other frequency-related settings.

To calculate the vibrational circular dichroism (VCD) spectrum [247, 248], set **ivcd**=1. VCD spectra can be used to determine the chirality of a molecule, by comparing the predicted spectrum with the experimental spectrum. The VCD spectrum of a molecule is the mirror image of the VCD spectrum of its enantiomer, so the assignment of a spectrum to a particular enantiomer is usually straightforward. Jaguar writes the VCD spectral data to a file called *jobname*`_vcd.spm`, which can be read into the Spectrum Plot panel in Maestro to generate a simulated spectrum.

Table 9.26. Keywords for frequency-related properties

Keyword	Value	Description
ifreq	***0***	Do not calculate frequencies (second derivatives)
	1	Calculate frequencies from second derivatives of energy (sets **irder**=1 for HF and DFT wave functions).
	–1	Calculate frequencies from most recent Hessian (from end of optimization or from initial Hessian if initial Hessian was never updated)
irder	***0***	Do not compute dipole derivatives or IR intensities
	1	Compute derivatives of dipole moment and IR intensities (see text for details)
iraman	***0***	Do not compute Raman intensities
	1	Compute Raman intensities
ivcd	***0***	Do not calculate vibrational circular dichroism spectra.
	1	Calculate vibrational circular dichroism spectra (sets **ifreq**=1 and **nmr**=1).

Table 9.26. Keywords for frequency-related properties (Continued)

Keyword	Value	Description
maxitcp	***35***	Maximum number of CPHF iterations
rmscp	5×10^{-5}	CPHF convergence threshold
imw	***0***	Print normal modes in cartesian coordinates without mass-weighting
	1	Print normal modes in mass-weighted cartesian coordinates
isqm	***0***	Do not scale frequencies using Pulay's Modified Scaled Quantum Mechanical Force Fields (SQM) method
	1	Scale frequencies using Pulay's SQM method, and use scaled frequencies for thermochemical calculations (only allowed for B3LYP calculations with the 6-31G* basis set)
auto_scale	***0***	Do not automatically scale frequencies.
	1	Automatically scale frequencies using a tabulated set of scale factors for various combinations of method and basis set (see Table 3.4 on page 71). If factors are not available for the method and basis set, frequencies are not scaled.
scalfr	>0	Scale vibrational frequencies by this factor (default is 1.0), and use scaled frequencies for thermochemical calculations
freqcut	***10.0***	Threshold in cm^{-1} for inclusion of frequencies in zero-point energy calculations and thermochemical analysis.
press	>0	Pressure for thermochemical calculations from frequencies, in atm. Default: 1.0
tmpini	>0	Initial temperature for thermochemical calculations, in K; default is 298.15
tmpstp	>0	Temperature step size (difference between consecutive temperatures) for thermochemical calculations, in K; default is 10.0
ntemp	>0	Number of temperatures at which thermochemical properties are computed; default is 1

Because different conformers have significantly different VCD spectra, it is important to compute a conformationally averaged spectrum when the molecule of interest has two or more conformers with similar energies. You can use MacroModel to perform the conformational search, then refine the geometries with Jaguar, and eliminate redundant or high-lying conformers with MacroModel. This procedure is automated in the VCD Workflow—see Section 5.3 on page 119. After calculating their VCD spectra, the Spectrum Plot panel can be used to generate an averaged spectrum using Boltzmann weights based on Jaguar's conformational energies. See Section 6.1 of the *Jaguar Quick Start Guide* for a tutorial example.

9.5.17 Electron Transfer Keywords

The **etransfer** keyword allows you to calculate the coupling matrix element for charge transfer between two structures that are in a specified geometric arrangement. This capability is important for charge mobility calculations on materials. Setting **etransfer**=1 requests calculation of the electronic coupling matrix elements.

The coupling is calculated between an initial state and a final wave state, according to the following expression:

$$T_{fi} = (H_{if} - S_{if}(H_{ii} + H_{ff})/2)/(1 - S_{if}^{2})$$

where H_{ii}, H_{ff}, and H_{if} are matrix elements of the system Hamiltonian, and S_{if} is the overlap matrix element. As different wave functions are used for the initial and final states, the orbitals for these states are not orthogonal, so the Hamiltonian and overlap matrix elements for the transfer (H_{if} and S_{if}) are calculated in a biorthogonal set, generated by singular value decomposition of the overlap matrix between the initial and final orbital sets.

The calculation of the coupling matrix element requires input wave functions for the initial state, with the charge localized in one part of the composite system, and the final state, with the charge localized on the other part of the composite system. You can either perform calculations on each structure, both neutral and charged, or use constrained DFT (CDFT) to perform calculations on the composite system, one with the charge constrained on one structure and one with the charge constrained on the other. The first method does not include the effect of polarization of the uncharged structure by the charged structure, as they are treated independently; the second method does include polarization effects.

The wave functions are taken from restart files (*jobname*`.01.in`), which are specified with directives, placed at the beginning of the input file. Four restart files are needed if calculations on the individual structures are used, specified with the `ET1A`, `ET2A`, `ET1D` and `ET2D` directives, and two restart files are needed if CDFT calculations on the composite system are used, specified with the `ET1DA` and `ET2DA` directives. The directives are given in Table 9.27. The calculations on the initial and final state structures must use the same coordinates, i.e. the structure is frozen. The input file for the charge transfer calculation must be set up for the composite system, and if separate calculations are done on each structure, the coordinates of the donor must be placed before the coordinates of the acceptor in the input file.

9.5.18 Basis Set Keywords

The **basis** keyword allows you to override the default basis set (6-31G**). This keyword should be a string describing the standard basis plus any polarization and diffuse functions. The string describing the standard basis should be chosen from the first column of Table 3.1 on

Table 9.27. Directives for specifying the wave function for charge transfer calculations.

Directive	Description
ET1A	Jaguar restart file for the charge acceptor in its initial state.
ET2A	Jaguar restart file for the charge acceptor in its final state.
ET1D	Jaguar restart file for the charge donor in its initial state.
ET2D	Jaguar restart file for the charge donor in its final state.
ET1DA	Jaguar restart file for the donor-acceptor complex from a CDFT calculation, with charge localized on one structure (initial state).
ET2DA	Jaguar restart file for the donor-acceptor complex from a CDFT calculation, with charge localized on the other structure (final state).

page 34 or Table 3.2 on page 36. These tables list the basis sets along with the atom ranges and accepted options. Lowercase or uppercase letters can be used. Polarization and diffuse functions are specified by adding *, **, +, or ++ immediately after the basis name. The meaning of these symbols is also described in Section 3.2 on page 33. Neither polarization nor diffuse functions are used if none of these options are specified.

The other keyword relating to the basis set, **numd**, allows you to choose whether to use five or six d functions in each d shell. If you do not set **numd** explicitly, the number of d functions is set automatically, depending on the basis set, as described in Section 3.2. Possible settings for **numd** are shown in Table 9.28.

Table 9.28. Keyword to determine the number of d functions

Keyword	Value	Description
numd	5	Use 5 d functions, regardless of basis set
	6	Use 6 d functions, regardless of basis set

9.5.19 Keywords for SCF Methods

Many of the keywords that control the SCF calculation can be set from the SCF tab as described in Section 3.7 on page 53. (The other keyword settings corresponding to SCF tab settings are described in Section 9.5.21.)

The SCF keywords are described in Table 9.29. The keyword settings for convergence are somewhat complicated, and the defaults vary, depending on the settings of other keywords.

Table 9.29. Keywords for methods used in the SCF convergence procedures

Keyword	Value	Description
iuhf	0	Restricted open-shell (ROHF or RODFT) calculation. Default for closed shells, LMP2, open-shell singlet (or other **ihamtyp** ≠ 0).
	1	Unrestricted (UHF or UDFT) calculation. Default for open shell other than LMP2 or when **ihamtyp** ≠ 0.
	2	Run restricted calculations on closed-shell systems (**multip**=1), and unrestricted calculations on open-shell systems (**multip**>1).
iconv	0	Convergence via Fock matrix diagonalization
	1	DIIS convergence scheme (default choice for most calculations)
	3	OCBSE convergence scheme
	4	GVB-DIIS convergence scheme (closed-shell calculations only)
maxit	0	Calculate energy, but do not update wave function (do only one iteration)
	>0	Perform a maximum of maxit SCF iterations (default 48)
newcon	0	Use physical constants and conversion factors equivalent to those used in GAUSSIAN 86
	1	Use physical constants and conversion factors equivalent to those used in GAUSSIAN 88, 90, 92
iacc	1	Only ultrafine grid used; "tight" cutoffs Set automatically when large energy changes and oscillations are detected.
	2	Accurate: mixed grid types, accurate cutoffs (default choice for transition metals, sometimes for other atoms beyond Ar)
	3	Quick: mixed grid types, looser cutoffs
iacscf	***0***	Make no special adjustments (variable vshift, cutoff adjustments, etc.) for convergence
	1	Start with level shift (**vshift)** of 5.0 and decrement over 10 iterations to 0.8; remove overlap eigenvectors if their associated eigenvalues are less than 5e-3; keep number of canonical orbitals fixed during optimization; run at ultrafine accuracy level and with tight cutoffs
	2	Start with level shift (**vshift**) of 6.0 and decrement over 12 iterations to 0.8; vary level shift during run by raising it when SCF is restarted (here, when the energy rises by 0.0001 au)
	3	Use extreme cutoffs (maximal analytic corrections) while still allowing medium pseudospectral grids for some iterations
	4	Same as **iacscf**=1, except with maximal analytic corrections

Table 9.29. Keywords for methods used in the SCF convergence procedures (Continued)

Keyword	Value	Description
iusediffuse	0	Do not use grids and dafs from the ++ basis set for a + basis set.
	1	Use grids and dafs from the ++ basis set for a + basis set (i.e. diffuse functions on heavy atoms only). This option can improve convergence.
retryscf	0	Do not retry SCF if convergence failure is detected.
	1	Retry SCF with **iacc**=1 and **vshift**=1.0 if a convergence failure is detected.
noabortscf	***0***	Abort SCF if checks indicate convergence failure.
	1	Ignore checks that would abort the SCF before it has reached the maximum iterations.
allow_scf_failure	***0***	Abort SCF if convergence is not achieved in the requested number of cycles. No output or properties are generated, but the restart file is written so you can restart the SCF.
	1	Treat the SCF as converged if it has reached the maximum iterations without satisfying the convergence criteria.
jksep	***0***	$2\boldsymbol{J} - \boldsymbol{K}$ formed for core when convenient
	1	$\boldsymbol{J}$ and $\boldsymbol{K}$ for core are kept separate
maxdiis	***10***	Number of Fock matrices to use in DIIS extrapolation
noatcor	***0***	Analytic corrections calculated
	1	No analytic corrections calculated
nops	***0***	Use pseudospectral method to calculate $\boldsymbol{J}$ and $\boldsymbol{K}$ operators
	1	Construct $\boldsymbol{J}$ and $\boldsymbol{K}$ from analytic two-electron integrals (no grid used)
noupdat	***0***	Fock matrix updating [202] set on or off automatically
	1	No Fock matrix updating (set **iacc**=1 if you set **noupdat**=1)
iteravg	***0***	Do not average density matrices and adjust orbitals accordingly (unless **istavg** keyword requests averaging)
	>0	For iterations whose number is n***iteravg** + 1, where n is an integer, revise orbitals so that they correspond to average of density matrices from preceding and current iterations

Table 9.29. Keywords for methods used in the SCF convergence procedures (Continued)

Keyword	Value	Description
istavg	*0*	Do not average density matrices and adjust orbitals accordingly (unless **iteravg** keyword requests averaging)
	>0	For iterations whose number is ≤ **istavg**, revise orbitals so that they correspond to average of density matrices from preceding and current iterations
noauto	***0***	Grid choice is automatic
	1	All calculations done on coarse grid
	2	All calculations done on medium grid
	3	All calculations done on fine grid
	4	All calculations done on ultrafine grid
idenavg	***0***	Converge in the usual fashion
	1	Do density averaging (damping) before DIIS starts, mixing in some of old orbitals with new orbitals (default for DFT calculations)
lastwv	***0***	Skip diagonalization of Fock matrix on last iteration
	1	Diagonalize Fock matrix on last iteration
nosuper	*0*	Evaluate integrals simultaneously over s and p basis functions with the same exponents (“superblocks”)
	1	Evaluate integrals separately for s and p basis functions which have the same exponents
	2	Use superblocks for all integrals except for gradient
itwice	***1***	Do A_{mng} integrals once in SCF routine
	2	Do A_{mng} integrals twice in SCF routine. Optional for HF
ichange	≠0	Change all cutoffs (except those related to ***S*** eigenvalues, bc pairs, or ab distance cutoff for exchange) by a factor of 10 to the **ichange** power
dconv	5×10^{-6}	SCF density convergence threshold: RMS change in density matrix.
econv	5×10^{-5}	SCF energy convergence threshold in hartrees: maximum difference in energy between one SCF iteration and the next required for convergence. Default is 1e-6 for polarizability and hyperpolarizability.
stdiis	***100.0***	DIIS initiation threshold: DIIS is started when the largest value of the DIIS error vector is less than this value.
large_diis	***3.0***	Threshold for considering the DIIS error (largest value of the DIIS error vector) to be “large”. The SCF is restarted if the DIIS error is large and the energy change is large after the third iteration.

Table 9.29. Keywords for methods used in the SCF convergence procedures (Continued)

Keyword	Value	Description
huge_diis	***10.0***	Threshold for considering the DIIS error (largest value of the DIIS error vector) to be "huge". The SCF is restarted if the DIIS error is huge in any iteration.
decut1		Maximum SCF energy decrease in any iteration (negative, in hartrees). If the energy decreases by more than this amount, it is considered "large". The SCF is restarted if the energy decrease is large and the DIIS error is large after the third iteration. The default is the least negative of 0.12 times the total energy and –40; the factor can be changed by setting opt(288).
decut2		Maximum SCF energy increase in any iteration (in hartrees). If the energy increases by more than this amount, it is considered "large". The SCF is restarted if the energy decrease is large and the DIIS error is large after the third iteration. The default is the smaller in magnitude of 0.2 times the total energy and 100; the factor can be changed by setting opt(289).
vshift	***0.0***	Level shift for virtual orbitals: the amount the virtual orbital energies are increased before diagonalization, in hartrees. Default for p-, d-, and f-block metals for pure DFT is 0.3, for hybrid DFT is 0.2.
ifdtherm	***0***	Do not use thermal smearing in DFT or HF calculations
	1	Use fractional occupation number (FON) method for thermal smearing [203]
	2	Use pseudo-fractional occupation number (pFON) method for thermal smearing [203]
fdtemp	***10000***	Initial temperature in K for thermal smearing
icanorb	0	Allow number of canonical orbitals to vary during calculation
	1	Fix number of canonical orbitals during calculation
ncanorb	>0	Number of canonical orbitals to keep during calculation
nofail	***0***	Do not use special measures to prevent SCF or geometry convergence failure.
	1	Use special measures when bad SCF convergence is detected, including automatic restart with a smaller basis followed by restart with the original basis. Also applies to geometry optimization convergence: if a geometry optimization does not converge, use the geometry that gives the lowest energy as the final result.
restart_basis	***lacvp***	Basis set used for restarting calculations for `nofail=1`.

The keywords for convergence thresholds are **econv**, the energy convergence criterion, which is the maximum difference in energy between one SCF iteration and the next required for convergence, and **dconv**, the criterion for the root mean square change in density matrix elements. The default value of **econv** is normally 5.0×10^{-5} hartrees. However, for polarizability or hyperpolarizability calculations, **econv** is 1.0×10^{-6} hartrees by default. When the root mean squared change in density matrix elements for a polarizability, hyperpolarizability, or geometry optimization calculation is less than **dconv**, whose default value is 5.0×10^{-6}, the calculation is considered to have converged.

By default, calculations use the DIIS (or GVB-DIIS) convergence scheme, which generates an estimate of the Fock matrix that is a linear combination of current and previous Fock matrices determined to minimize the norm of the error vector. The keyword **maxdiis**, sets the maximum number of Fock matrices (default 10) that are used for this scheme during any iteration. The keyword **stdiis** gives an error criterion: DIIS is started when the largest value of the DIIS error vector is less than the value of **stdiis (**hartrees), which is 100.0 by default. In general, after GVB-DIIS starts, any density matrix averaging requested by the keywords **iteravg** and **istavg** (explained in Table 9.29) is turned off.

The keyword **vshift** sets the amount the virtual orbital energies are increased before diagonalization, in atomic units. This keyword can be used to reduce mixing of the real and virtual orbitals, which sometimes helps convergence. By default, **vshift** is zero, except for DFT calculations on p-block, d-block, and f-block metals, when the default is 0.2 (for hybrid methods) or 0.3 (for non-hybrid methods). Non-default values should probably be on the order of 0.1–0.5.

The **iacscf** keyword may be employed if other attempts to converge the SCF fail. This can be useful especially for transition metal-containing systems or clusters. Start with **iacscf** = 1 and if that does not work then try **iacscf** = 4. **iacscf** = 2 was developed especially for hemes and related molecules, while **iacscf** = 3 was effective for graphitic systems. Energies obtained with **iacscf** = 2 or 3 can be directly compared to energies obtained without using **iacscf**. Energies obtained using **iacscf** = 1 or 4 can only be compared to other energies obtained using the same value of **iacscf**, because these two settings reduce the number of canonical orbitals used.

Another method for controlling SCF convergence is thermal smearing [203], which improves convergence in difficult cases by using a fictitious temperature to fractionally occupy all orbitals, occupied and virtual, and then decrease the temperature until convergence is reached. The orbital occupation numbers are represented by a Fermi-Dirac function, $n = 1/(1 + \exp((\varepsilon - \varepsilon_F)/kT))$. Two methods for determining the occupation numbers have been implemented, FON (fractional occupation number) and pFON (pseudo-FON). In the FON method, the Fermi energy is determined so that the resulting occupations sum to the total number of electrons. In the pFON method, a Fermi energy is assigned halfway between the HOMO and LUMO energies and then the resulting occupations found from this Fermi energy are renormalized so that they sum to the total number of electrons.

Thermal smearing is turned on using the keyword **ifdtherm**. A value of 1 selects the FON method; a value of 2 selects the pFON method. You can use thermal smearing with the RHF, ROHF, UHF, DFT, RODFT, and UDFT methods. The number of alpha and beta electrons is kept the same during thermal smearing. Thermal smearing can be used with or without symmetry, and it can be used with the **ipopsym** keyword.

You can set the initial temperature using the **fdtemp** keyword. The units of **fdtemp** are Kelvin. The default initial temperature is 10,000 K. The temperature decreases as a function of the rms density change. When the density is close to the convergence threshold, the temperature is set to zero.

The number of canonical orbitals kept in an SCF calculation is controlled by the **cut20** keyword. Eigenvectors of the overlap matrix, i.e. canonical orbitals, are discarded in a calculation if their eigenvalues are less than **cut20**. It may be necessary to fix the number of canonical orbitals during a calculation, such as during a geometry optimization or scan, or between calculations, such as when comparing energies of related structures. You can set the number of canonical orbitals with the **ncanorb** keyword, and you can fix the number of canonical orbitals to the number determined for the initial structure by setting **icanorb**=1. When **ncanorb** is set to a value less than the number of basis functions, the canonical orbitals with the lowest eigenvalues of the overlap matrix are discarded until there are **ncanorb** orbitals left. Setting **ncanorb** sets **icanorb** to a positive value.

9.5.20 Initial Guess Keywords

Table 9.30 lists the keywords related to the initial guess and the meaning of the values each keyword can take on. Most of the keyword values in Table 9.30 correspond to options described in Section 3.7 on page 53.

Table 9.30. Initial guess keywords

Keyword	Value	Description
atguess	*0*	No effect
	1	Generate a spherically averaged atomic wave function for use in the `default.atomig` file (see Section 10.2 on page 298). Sets the energy convergence to 1.0e-10, uses the GVB-DIIS convergence method (**iconv**=4) and the highest accuracy cutoffs in the SCF.
igonly	*0*	No effect
	1	Use initial guess or input wave function for any post-SCF calculations, skipping SCF step. No J, K, or Fock matrices are created, therefore properties that require any of these matrices cannot be calculated.

Table 9.30. Initial guess keywords (Continued)

Keyword	Value	Description
iguess	0	Generate initial guess by diagonalizing one-electron Hamiltonian
	1	Read initial guess from **guess** section from input file or from guess file specified in WAVEFNFILE line (**iguess**=1 automatically if input file contains non-empty **guess** section)
	10	Construct initial guess from orbitals that give best overlap with atomic orbitals in `default.atomig` (or other `.atomig` file listed in the input file) obtained by SCF calculations on atoms (note that if **guess** section exists, this is not the default choice)
	11	Construct initial guess from orbitals whose sum of densities best agrees with the sum of densities of the atomic orbitals in `default.atomig` (or the specified `.atomig` file)
	25	For a system that contains transition metal atoms, construct a high-quality initial guess using ligand field theory as described in Ref. 21.
	30	For a system that contains transition metal atoms, construct a high-quality initial guess using ligand field theory including d-d repulsion, as described in Ref. 21.
inewguess	***0***	Do not generate a new default initial guess if the original guess is bad.
	1	Generate a new initial guess if the supplied guess is bad. The quality of the guess is determined by testing the maximum absolute value of the DIIS error vector; if it is larger than **guess_thresh**; a new guess is generated with the default method.
guess_thresh	***1.0***	Threshold for determining if the original guess is bad, based on the maximum absolute value of the DIIS error vector.
ihamtyp	***0***	Construct Hamiltonian using standard orbitals
	2	Make highest two orbitals in initial guess an open-shell singlet pair (ROHF only)
	3	Input Hamiltonian in **ham** section (**ihamtyp**=3 by default if a non-empty **ham** section exists)
ioss	***0***	Use the default open-shell guess.
	1	Set up an "open-shell singlet" initial guess by mixing the LUMO into the HOMO, and set **isymm**=0 and **iuhf**=1. See text.
istate	***0***	Use the default state selection when there are degenerate states in transition-metal systems. Same as setting **istate**=1.
	n	Perform calculation on state number *n*. *n* is the index of the state in the output from `hfig` for degenerate states in transition-metal systems.

If you want to perform an "open-shell singlet" calculation using UDFT or UHF, set **ioss**=1. The alpha and beta HOMO are replaced with a mixture of the HOMO and LUMO, as follows:

$$\varphi^{\alpha}{}_{\mathrm{HOMO}} = (\varphi_{\mathrm{HOMO}} + \varphi_{\mathrm{LUMO}})/\sqrt{2}$$

$$\varphi^{\beta}{}_{\mathrm{HOMO}} = (\varphi_{\mathrm{HOMO}} - \varphi_{\mathrm{LUMO}})/\sqrt{2}$$

The orbitals are taken from a closed-shell starting guess. The LUMO remains the same. This option also sets **isymm**=0 and **iuhf**=1. Do not use this keyword for transition metals, for which you should use the **2spin** column in an atomic section to set up an antiferromagnetic guess.

Note: This starting guess does not correspond to the open-shell singlet state, but is a mixture of singlet closed-shell and triplet open-shell states. The final wave function in a UHF calculation will not necessarily correspond to what would be obtained in an ROHF calculation, and might be a mixture of a singlet and a triplet state. You should check the value of S^2 in the output to determine the extent of spin contamination. In UDFT calculations, exchange is handled differently, and all that can be concluded is that the final density represents the lowest state. This is more correctly described as a spin-polarized method rather than an open-shell singlet method; for UDFT it yields the correct dissociation behavior for a sigma bond.

In the transition metal initial guess section (`tmig`), Jaguar generates a set of states with all possible d orbital occupations. You can select one of these states by setting **istate** to the state index. This capability is useful when there are degenerate states, such as in highly symmetric transition metal complexes. When degenerate states are encountered, Jaguar prints a warning message that lists the states with a state index number, and by default continues the calculation with the first state. This calculation might not converge to the ground state, so you should run calculations for each state to determine which is the desired state. To run a calculation with another state, set **istate** to the state index number listed in the output.

9.5.21 Localization Keywords

For any Jaguar job, the final wave function can be localized after it is computed. Localization can also be used to provide localized orbitals for the LMP2 method; see Section 9.5.6 on page 206 for details. The keywords in Table 9.31 describe the available options for final wave function localization. See Section 3.7.5 on page 57 for a description of the localization methods and the GUI settings related to localization.

When the keyword **ixtrboy** described in Table 9.31 is set to 1, an additional procedure is added on to the Boys localization process. Boys orbitals may be unphysical for multiple bonds, since they create multiple "banana" bonds between pairs of atoms rather than forming sigma-like, pi-like, and related orbitals. The Boys orbitals for multiple bonds may therefore be diagonalized

Table 9.31. Keywords related to localization of orbitals

Keyword	Value	Description
locpostc	***0***	Do not localize core orbitals of final wave function
	1	Perform Boys localization on core orbitals of final wave function
	2	Perform Pipek-Mezey localization on core orbitals of final wave function, maximizing Mulliken atomic populations
	3	Perform Pipek-Mezey localization on core orbitals of final wave function, maximizing Mulliken basis function populations
	–1	Mix the core and valence orbitals before localization, then localize according to the **locpostv** setting
locpostv	***0***	Do not localize valence orbitals of final wave function
	1	Perform Boys localization on valence orbitals of final wave function
	2	Perform Pipek-Mezey localization on valence orbitals of final wave function, maximizing Mulliken atomic populations
	3	Perform Pipek-Mezey localization on valence orbitals of final wave function, maximizing Mulliken basis function populations
iordboy	0	Do not order orbitals at end of Boys localization
	1	Order orbitals by their one-electron energy at the end of Boys localization
ixtrboy	***0***	Do not try to diagonalize multiple-bond orbitals at the end of the Boys localization
	1	Try to diagonalize multiple-bond orbitals at the end of the Boys localization—see text in this subsection

using the one-electron Hamiltonians. The output for this procedure begins with a table of the Mulliken populations for each orbital on each atom, which reveals multiple bonds, as described in the following table. Every "bond pair space" made up of all orbitals with significant Mulliken populations on the same pair of atoms is diagonalized, and the output indicates the number of these bond pair spaces found and the ordering of the new orbitals by their one-electron Hamiltonian values. If you choose to print out Boys orbitals by setting the print keyword **ip107** to 2, it is these final orbitals which are printed.

9.5.22 File Format Conversion Keywords

You can call the program Babel [26] from Jaguar to generate files in any of a variety of formats, although the files produced by Babel contain only geometries, not calculation settings. The output can be generated at the end of each iteration in a geometry optimization or at the end of any job. To generate such an output file, you must set the format keyword for the chosen file type. The format keywords and file types supported are shown in Table 9.32.

If you want to generate an output file in a particular format only at the end of a job, you should use a keyword setting of the form **babel**=*outext*, where *outext* is one of the possible format keywords listed in Table 9.32. You can set **babel** more than once, using separate **babel**=*outext* assignments, if you want to generate several files.

To generate output files at the end of each iteration in an optimization, set the **babelg** keyword to the appropriate output extension string. Like the **babel** keyword, the **babelg** keyword can be set more than once to generate files in several formats.

As files are generated with Babel during Jaguar runs, they are immediately copied back to the relevant output directory. Files generated from jobs with **babel** keyword settings have names of the form *jobname*`.`*outext* (for instance, `h2o.spar`), where *jobname* is the usual job name and *outext* is the format keyword, which is used as the output extension. Files generated from geometry optimizations with **babelg** settings have names of the form *jobname#*`.`*outext*, where *#* is a four-digit number corresponding to the iteration number (for example, 0001 for the first geometry iteration), and all letters in the job name are converted to lower case by Babel. Note that you can use a **babelg** keyword setting to write structures generated during an optimization as the optimization proceeds.

For either **babel** or **babelg** keyword settings, you can use an optional extra extension for the file name by setting **babel** (or **babelg**) to a keyword in the form *outext*`.`*opt,* where *opt* is any extension you want to use. For instance, if you set **babel**=`gzmat.gau` in a Jaguar input file called `h2o.in`, the resulting job would create a Gaussian input file called `h2o.gzmat.gau`.

You can also convert file formats from the command line using the `jaguar babel` and `jagconvert` utilities. See Section 11.1.2 on page 318 for information on these utilities.

Table 9.32. Output format keywords and file types for babel file format conversions

Format Keyword	File Type
alc	Alchemy file
bs	Ball and Stick file
bgf	MSI BGF file
bmin	MacroModel file
box	DOCK 3.5 box file
caccrt	Cacao Cartesian file
cacint	Cacao Internal file

Table 9.32. Output format keywords and file types for babel file format conversions (Continued)

Format Keyword	File Type
cache	CAChe MolStruct file
c3d1	Chem3D Cartesian 1 file
c3d2	Chem3D Cartesian 2 file
cdct	ChemDraw Conn. Table file
diag	DIAGNOTICS file
dock	Dock Database file
wiz	Wizard file
contmp	Conjure Template file
cssr	CSD CSSR file
dpdb	Dock PDB file
feat	Feature file
fhz	Fenske-Hall ZMatrix file
gamin	Gamess Input file
gzmat	Gaussian Z-matrix file
gotmp	Gaussian Z-matrix tmplt file
gr96A	GROMOS96 (A) file
gr96N	GROMOS96 (nm) file
hin	Hyperchem HIN file
icon	Icon 8 file
idatm	IDATM file
sdf	MDL Isis SDF file
jagz	Jaguar Z-Matrix file
jagc	Jaguar Cartesian file
m3d	M3D file
macmol	Mac Molecule file
macmod	Macromodel file
micro	Micro World file
mm2in	MM2 Input file

Table 9.32. Output format keywords and file types for babel file format conversions (Continued)

Format Keyword	File Type
mm2out	MM2 Output file
mm3	MM3 file
mmads	MMADS file
mdl	MDL Molfile file
miv	MolInventor file
mopcrt	Mopac Cartesian file
mopint	Mopac Internal file
csr	MSI Quanta CSR file
pcmod	PC Model file
pdb	PDB file
psz	PS-GVB Z-Matrix file
psc	PS-GVB Cartesian file
report	Report file
smiles	SMILES file
spar	Spartan file
mol	Sybyl Mol file
mol2	Sybyl Mol2 file
maccs	MDL Maccs file
torlist	Torsion List file
tinker	Tinker XYZ file
unixyz	UniChem XYZ file
xyz	XYZ file
xed	XED file

9.5.23 Standard Output Keywords

The keywords listed in Table 9.33 are the standard print options. They are all set to 1 by default, which suppresses output of the information that the keywords select. The exception is **ip6**, which is set to 2 by default. Many of the print options can be turned on from the GUI, as described in Section 3.13 on page 82.

The keyword setting **ip6**=3 provides much more detailed timing information than the setting **ip6**=2. Similarly, the keyword setting **ip192**=3 provides more detailed output than **ip192**=2; the **ip192**=3 setting includes the Hessian.

The keyword setting **kesep**=1, which is normally part of a solvation calculation (see Table 9.22 on page 231), causes the virial ratio, –V/T, to be printed out at the end of each SCF calculation.

Table 9.33. Output keywords and their settings

Keyword	Value	Description
ip1	2	Gaussian function list for basis set
ip3	2	Gaussian function list for dealiasing functions
ip4	2	Number of dealiasing functions used
ip5	2	Memory, disk, and i/o information
ip6	2	Timing information (user CPU and user+system CPU)
	–2	Timing information (user cpu and wall clock)
ip7	2	Grid shell locations
ip8	2	Gaussian function list for derivatives of basis functions
ip11	2	Bond lengths and angles
	3	Same as setting **ip11**=2, but includes all internuclear distances (regardless of connectivity) and torsions
	4	Same as setting **ip11**=3, but includes all possible angles (regardless of atom connectivity)
	5	Same as setting **ip11**=4, but includes all possible torsions (regardless of atom connectivity)
ip12	2	Connectivity table
ip13	2	Eigenvectors and eigenvalues of overlap matrix
ip18	2	Overlap matrix
ip19	2	One-electron Hamiltonian
ip20	2	Additional output from RwR, including inverse condition numbers and dealiasing functions associated with small eigenvalues of RwR
ip23	2	Additional DFT grid information
ip24	2	*All* keyword settings, including internal ones
ip25	2	Multipole moments in atomic units (and Debye)
ip26	2	Geometries in bohr as well as Angstroms

Table 9.33. Output keywords and their settings (Continued)

Keyword	Value	Description
ip70	2	Extra geometry optimization details
ip170	2	Localized orbital locations and LMP2 pair energies for local LMP2 calculations (full local LMP2 energy correction is sum of pair energies)
ip173	2	Fock matrix in Boys-localized orbital space
ip192	2	Extra optimization-related information, such as the quadratic energy error
	3	Same as setting **ip192**=2, but includes more detailed information such as the Hessian
ip193	2	Numerical Hessian in freq output
ip194	2	Diagonal force constants in internal coordinates
	3	Same as setting **ip194**=2, but also includes off-diagonal force constants if they are larger than a factor (0.01 by default) times the geometric mean of the corresponding off-diagonal elements; the value of the factor can be set using the **opt194** keyword
	4	All diagonal and off-diagonal force constants are printed

9.5.24 File Output Keywords

The file output keywords are used to write files other than the usual log and output files. All but one of these keywords are set to 1 by default, meaning that the file is not created.

The file output keyword **ip151** controls whether or not a Jaguar restart file is written. It is the only file output keyword whose default value of 1 indicates that it is on. When **ip151** is set to 1, the file `restart.in` is created in the temp directory for the job at the end of the last completed Jaguar program, writing over any previously generated `restart.in` file for the job. The file `restart.in` contains the results from the run, including the new geometry if the run that produced it was a geometry optimization. This input file can therefore be used to restart the calculation. At the end of the job, the restart input file is copied to your local job directory (under the name *jobname*`.01.in`, unless that file already exists, otherwise *jobname*`.02.in` or *jobname*`.03.in`, and so on). To turn off **ip151**, you must set it to 0.

Setting **ip472**=2 writes structures at each step of a geometry optimization or an IRC scan to the Maestro-formatted output (`.mae`) file.

The other file output keywords control whether files for various other programs such as GAMESS are written out during a Jaguar job. The effect of setting each of these keywords to 2 is shown in Table 9.34. Many of these options can be turned on from the GUI, as described in Section 3.13.1 on page 85.

Table 9.34. Effect of setting output keywords for files to 2

Keyword[a]	Description of What Is Printed When ip*i* = 2
ip90	Molden orbitals file (`.molf` file)
ip160	GAUSSIAN 92 input file (`.gau` file) (see text for **ip160**=3, 4, or 5)
ip163	GAUSSIAN 92 basis set (`.gbs` file)
ip164	MQM basis set (`.bas` file)
ip165	SPARTAN 4.0 archive file; appears in temp directory as `spart.arc` (to write `.arc` file to local job directory instead, use **ip165**=3)
ip168	GAMESS input file (`.gamess` file)
ip172	RESP (Restrained Electrostatic Potential [204]) file (`.resp` file) Set to 3 to include grid weights.
ip175	XMol file (`.xyz` file) with geometries generated during optimization
ip177	AIMPAC (`.wfn` file) which works with RHF/ROHF but not UHF

a. See text in this subsection for information on **ip151** and on other options for **ip160**.

Additional settings are available for **ip160** and **ip165**. When **ip165**=3, the SPARTAN 4.0 archive file is written to the local job directory as *jobname*`.arc`. When **ip160**=3, an initial guess is included in the `.gau` file generated by the run. If SCF iterations are performed, the initial guess for the `.gau` file is obtained from the resulting wave function; otherwise, it is generated from the appropriate Jaguar initial guess routine. When **ip160**=5, the basis set is included explicitly in the `.gau` file, rather than just the basis set name. When **ip160**=4, the trial wave function and the basis set are included.

The format of the `.resp` file created with the **ip172** keyword is as follows. The first line contains the number of atoms and the number of grid points at which the electrostatic potential was evaluated. The cartesian coordinates of the atoms in bohrs are given next, one atom per line. Each subsequent line contains information for one grid point: the electrostatic potential value in hartrees, the coordinates of the grid point in bohrs, and, if **ip172**=3, the grid weights.

In addition to these keywords, there is a keyword for reporting the full paths to the input and output files as Maestro properties. The directory reported in the path is the subdirectory used for running the job. Setting **iflinks**=1 adds the names of the input, input Maestro, log, output, and restart files as the following properties:

```
s_j_Jaguar_input_file              s_j_Jaguar_output_file
s_j_Jaguar_mae_file                s_j_Jaguar_restart_input_file
s_j_Jaguar_log_file                s_j_Jaguar_restart_mae_file
```

These properties can be used in Maestro to keep track of the files used for a job. The default, **iflinks**=0, is to not add these properties to the Maestro output (restart) file.

9.5.25 Output Keywords for Each Iteration

The information in Table 9.35 concerns output which can be printed out every SCF iteration if the keyword is set to 2. The information is not printed if the keyword is set to 1. The option **ip152** is the only one whose default value of 1 indicates that it is on. When **ip152** is set to 1, the file `restart.in` is created in the temp directory for the job at the end of the last completed iteration (overwriting the `restart.in` file created from the previous iteration). This input file can then be used to restart the calculation. To turn off **ip152**, you must set it to 0.

Table 9.35. Effect of setting output keywords for each iteration to 2

Keyword[a]	Description of What Is Printed When ip*i* = 2
ip15	DIIS coefficients
ip17	Energy components
ip110	Density matrix (if Fock matrix updating was not performed during that iteration) or density difference matrix (if Fock matrix updating was done)
ip121	All ***J*** and ***K*** matrices, in atomic orbital space
ip122	Fock matrix in atomic orbital space (HF)
ip123	Fock matrix in canonical orbital space
ip188	Debug printing for automatic cutoff/convergence scheme
ip201	Total electronic density integrated on the DFT grid

a. See text in this subsection for information on **ip152**.

9.5.26 Orbital Output Keywords

Orbital information can be printed out as well. The orbital keywords determine what orbitals are printed in the output, at what stage they are printed, and the format.

The keyword **ipvirt** determines how many of the virtual orbitals are printed in the output file and in the restart (new input) file. Virtual orbitals are printed in order of increasing energy. The virtual orbitals are obtained by diagonalizing $H_0 + \sum f(2J - K)$, where *f* is the fractional occupation of each orbital (1 for a closed shell). If **ipvirt**=–1, all virtual orbitals are printed in the output and restart files; otherwise, **ipvirt** virtual orbitals are printed (if that many virtual orbitals exist). By default, **ipvirt**=10.

Several possible formats and levels of information can be requested for each other keyword determining the orbitals printed. The choice of keywords, which are listed in Table 9.36, determines the stage (or stages) at which orbitals are printed; the keyword values determine which orbitals are printed and the format of the printing. These settings can generally also be made from the GUI, as described in Section 3.13.2 on page 86.

Table 9.36. Keywords that specify when to print orbitals

Keyword	Prints Orbitals
ip100	For initial guess from before SCF (generally redundant with **ip105**)
ip101	In canonical orbital space (each SCF iteration)
ip102	At end of job
ip103	In atomic orbital space (each SCF iteration)
ip104	In atomic orbital space after SCF
ip105	For HF initial guess
ip107	After Boys or Pipek localization

Table 9.37 explains the possible values for the orbital output options, aside from 1, the default, which turns off printing. The variable **n** in the table can be either 0, 5, or 10. If it is 0, all occupied orbitals are printed. If **n** is 5, all occupied orbitals and **ipvirt** virtual orbitals are printed (or all virtuals if **ipvirt**=–1).

Table 9.37. Dependence of the format and type of orbital output on the value of **ip***x*

Value of ip*x*[a]	2 + n	3 + n	4 + n	5 + n	6 + n
Format	f5.2	f10.5	f19.15	f8.5	e15.6
Atom, basis function type shown	Y	Y	N	N	N
Orbital occupation indicated	Y	N	Y	Y	N
Coefficients printed	large	all	all	all	all
Form shown	list	table	list	list	table

a. The value of **n** determines which orbitals (e.g., occupied) are printed; **x** determines the stage at which orbitals are printed (see Table 9.36).

The options **ip105** ≥ 12 are not valid; use **ip100** instead. In canonical orbital space, the atom and function type labels are meaningless. If a keyword is set to 4, 5, 9, or 10, the orbitals can be used for input in the **guess** section or for GAUSSIAN (guess=cards).

When the orbital output is in table form, each function's coefficient for each orbital is shown, with the functions shown in numbered rows and the orbitals in numbered columns. When it is in list form, each orbital is listed in turn, with the function coefficients listed in order. When **ip*x*** = 2 + **n**, only coefficients larger than a particular value (generally .05) are listed, and the atom identifiers (for instance, h2) and function types (for instance, S for *s*, Z for p_z) are shown. When **ip*x*** = 4 + **n** or **ip*x*** = 5 + **n**, all coefficients are listed, in order but without numbering.

For examples of the output that shows up in the output file for a calculation of water with a 6-31G** basis set for various values of **ip104**, see the five examples given at the end of Section 3.13.2 on page 86. The five examples correspond to **ip104**=2, **ip104**=3, **ip104**=4, **ip104**=5, and **ip104**=6, in that order. Only the first two occupied orbitals are shown in each case, and not all functions are shown; those gaps are indicated by [...].

9.5.27 Grid and Dealiasing Function Keywords

The grid and dealiasing function keywords allow the user to select from among the various sets of grids and dealiasing functions available in the grid and dealiasing (`.grid` and `.daf`) input files, which are described in Section 10.3 on page 300 and Section 10.4 on page 305, and from the grids generated within Jaguar. These keywords are used to specify which grid or dealiasing sets correspond to particular descriptions; this correspondence is often indicated by keyword values depending on the order of sets in the grid and dealiasing input files.

For density functional theory calculations, the grid keywords **gdftmed**, **gdftfine**, **gdftgrad**, **gdftder2**, and **gdftcphf** select various predefined grids for the SCF (**gdftmed** and **gdftfine**), gradient, second derivative and CPHF calculations. The grids are indexed with negative numbers. The default values for these keywords are –10, –11, –12, –8, and –9. They can be assigned other values: for example, –13 corresponds to an ultrafine grid, and –14 to the largest DFT grid that can be defined in Jaguar, which has 125 radial shells and uses an angular offset of 30 (434 angular points per shell) with no pruning. To use such a grid throughout a geometry optimization, you would set the following keywords:

```
gdftmed=-14
gdftfine=-14
gdftgrad=-14
```

You can also define your own DFT grids using three keywords, which specify the number of radial shells, the number of angular points per shell, the pruning scheme, and the distribution of the radial shells. The keywords and their settings have the form:

ndfgrd*X*1=*nr*
ndfgrd*X*2=*na*
idfgrd*X*=*pqq*

where *X* is **m**, **f**, **g**, **u**, **d**, **p**, or **c**, signifying “medium,” “fine,” “gradient,” “ultrafine,” “second derivatives,” “main-CPHF”, and “post-CPHF,” and correspond to grids 10, 11, 12, 13, 8, 15, and 9; *nr* is the number of radial shells, *na* is the angular grid entry number from Table 10.1; *p* is a number denoting the radial shell distribution scheme; and *qq* is a two-digit number denoting the pruning scheme. The possible values for *p* are 1 (geometric distribution [231], the default for medium, fine, and gradient grids), 2 (Becke’s Gauss-Chebyshev distribution [232]), 3 (described in Ref [233]) and 4 (the Mura-Knowles distribution [234], the default for the ultrafine, second derivative, CPHF, and grid –14). The values of *qq* can be 00, 11, 22, or 33. 00 is the default for the medium grid, 11 is the default for the fine and gradient grids, and 33 is the default for the second derivative, CPHF, and ultrafine grids. 22 turns off pruning.

The value for **ndfgrd**X**2** is interpreted as an offset, to be added to the angular value for each radial shell that is determined from the pruning scheme. You can get more information about both pseudospectral and DFT grids for a job by setting **ip23**=2 in the input file.

Table 9.38 shows the types of grids that can be specified for portions of the calculation that do not involve density functional theory. Generally, these grid types are used for pseudospectral SCF iterations or for charge fitting.

The grid-related keywords and their allowed and default values are given in Table 9.39, where ***name*** corresponds to one of the grid types listed in Table 9.38. As an example, **gmedium**=2 indicates that the medium grid to be used is the second one listed in the `.grid` file, while **geldens**=–3 indicates that an electron density calculation should use a cubic grid.

You can read in your own set of grid points and weights by using **g*name***=–6 in the **gen** section and the `GPTSFILE` line (see Section 9.1 on page 195) at the top of the input file.

Table 9.38. Pseudospectral, charge-fitting, and electron density grid types

Name[a]	Description
coarse	Least expensive, least accurate level
medium	Used for most SCF iterations
fine	Sometimes used for a limited number of iterations
ufine	Ultrafine; most accurate level
grad	Used in gradient computation
lmp2	Grid used for LMP2 energy calculations
lmp2grad	Grid used for LMP2 gradient calculations
charge	Grid used for charge fitting
eldens	Used for electron density calculations

a. These names are used in the grid-related keywords described in Table 9.39.

Table 9.39. Keywords for specification of length scales for sorting of basis functions, grid usage, and dealiasing function usage. name is one of the grid types from Table 9.38.

Keyword	Value	Description	Default for
l*name*	1	Only one length scale used	**lcoarse**
	2	Basis functions are sorted into short- and long-range	**lmedium, lfine, lufine, lgrad**
g*name*	>0	Specifies which parameter set from `.grid` file should be used for grid (e.g., 2 for second)	**gcoarse** (1), **gmedium** (2), **gfine** (3), **gufine** (4), **ggrad** (4), **glmp2** (4), **glmp2grad** (2), **geldens** (4)
	–1	Use spherical charge fitting grid generated by Jaguar for grid ***gname***	**gcharge**
	–2	Use cubic charge-fitting grid generated by Jaguar for grid ***gname***	none
	–3	Use cubic electron density grid generated by Jaguar for grid ***gname***	none
	–6	Use grid and weights from file specified by `GPTSFILE` line in input file for grid ***gname***	none
d*name*	>0	Specifies which dealiasing function from the `.daf` file should be used	**dcoarse** (1), **dmedium** (2), **dfine** (3), **dufine** (4), **dgrad** (5)

9.5.28 Memory Use Keywords

Some of the memory use for Jaguar can be controlled through keywords. These keywords may be particularly useful if you are experiencing problems running jobs due to memory-related failures.

Memory-use keywords are listed in Table 9.40, along with their default values and a description of their uses. If you want to change some memory use but do not have a detailed knowledge of the code, do not change the variable **mxpr**.

Table 9.40. Keywords related to memory and disk use

Keyword	Default	Description
idynmem	256	Dynamic memory (in MB) to be allocated for LMP2 calculations. Suggested optimum value is machine memory – 256.
mxstrip	200	Information for matrix elements evaluated on basis functions stored in core in strips of **mxstrip***N words, rather than N^2 words at a time (where N is the number of basis functions).

Table 9.40. Keywords related to memory and disk use (Continued)

Keyword	Default	Description
nbuck	64	Grid blocks are split up into subblocks whose points are all on the same atom and in the same region of space, with at most **nbuck** points, where **nbuck** ≤ ngblok (ngblok is the maximum number of grid points per grid block, currently set to 128).
ndisk	5000	Atomic strips of $\boldsymbol{J}$ and $\boldsymbol{K}$ are kept in core rather than on disk if (# basis functions) x (# Hamiltonians) < **ndisk**. #Hamiltonians=1 for closed shell and 2 for open-shell.
mxpr	100	Pairs of dealiasing functions are organized so that each group's pairs have the same angular momentum values (e.g., a group with pairs with an s and a p function). The number of pairs in each group evaluated at the same time by subroutine novoro is restricted so that it is ≤ **mxpr**.
nbcmax	1000000	Maximum memory (in words) used by overlap and kinetic energy integral package, excluding final matrices themselves.
zmpmem	1.0	For LMP2 single-point and gradient code, maximum total size allowed for arrays holding partially transformed integrals on grid is 60 MB x **zmpmem**.

Finally, the **iq** keywords allow you to choose when to compute the full least-squares fitting matrix $\boldsymbol{Q}$ from the smaller matrix $\boldsymbol{S}[\boldsymbol{R}^{\dagger}\boldsymbol{wR}]^{-1}$ and whether to store it on disk. Names and default values (in bold italics) for these keywords are indicated in Table 9.41. If a grid is used only once per calculation, as the fine, ultrafine and gradient grids generally are, setting its **iq*name*** value to 0 saves disk space and costs no CPU time. Setting the **iq*name*** values for other grids to 0 adds some CPU cost, but saves some disk space.

Note: If you set **iqgrad**, you must set **iqufine** to the same value.

Table 9.41. Keywords to determine when to compute the full least-squares fitting matrix ***Q***

Keyword	Value	Description
iqcoarse	0	For coarse grid, compute $\boldsymbol{Q}$ on the fly in the program `scf`
	1	For coarse grid, compute $\boldsymbol{Q}$ in `rwr` and store on disk for later use
iqmedium	0	For medium grid, compute $\boldsymbol{Q}$ on the fly in the program `scf`
	1	For medium grid, compute $\boldsymbol{Q}$ in `rwr` and store on disk for later use
iqfine	***0***	For fine grid, compute $\boldsymbol{Q}$ on the fly in the program `scf`
	1	For fine grid, compute $\boldsymbol{Q}$ in `rwr` and store on disk for later use

Table 9.41. Keywords to determine when to compute the full least-squares fitting matrix ***Q***

Keyword	Value	Description
iqufine	0	For ultrafine grid, compute ***Q*** on the fly in the program `scf`
	1	For ultrafine grid, compute ***Q*** in `rwr` and store on disk for later use
iqgrad	0	For gradient grid, compute ***Q*** on the fly in the program `scf`
	1	For gradient grid, compute ***Q*** in `rwr` and store on disk for later use
iqlmp2	0	For LMP2 grid, compute ***Q*** on the fly in the program `scf`
	1	For LMP2 grid, compute ***Q*** in `rwr` and store on disk for later use
iqlmp2grad	0	For LMP2 gradient grid, compute ***Q*** on the fly in the program `scf`
	1	For LMP2 gradient grid, compute ***Q*** in `rwr` and store on disk for later use

9.5.29 Plotting Keywords

You can generate a plot file, using keywords in the **gen** section, that contains the values of the density, the spin density, the electrostatic potential, or orbital amplitudes. The data values are tabulated on a rectangular grid (the "box"), which is generated automatically and encompasses the van der Waals radii of all atoms in the molecule. The plot file can be used by Maestro and other programs to display surfaces for a particular value of the density, potential, or amplitude. The length units for the grid are set with the **iunit** keyword.

The possible values of the plotting keywords are given in Table 9.42. See Section 3.11 on page 75 for information on setting up plot data using the GUI.

Orbital amplitude data can only be generated for SCF wave functions: MP2 calculations do not generate natural orbitals that could be used for generating surfaces.

When the job is run, each type of output requested is written to a file whose name depends on *jobname*, the name for the job (for example, `h2o`), and the type of information being plotted. The file name stem is *jobname*`_density` for a density plot, *jobname*`_spin` for a spin density plot, and *jobname*`_potential` for a potential plot. Orbital plot information is written to separate files for each orbital. The orbital file name stems are of the form `jobname_`*spin*`_HOMO`*#* for occupied orbitals, and `jobname_`*spin*`_LUMO`*#* for unoccupied orbitals, where *spin* can be `alpha` or `beta`, and # is the orbital index number relative to the HOMO or LUMO (empty string instead of 0 for the HOMO or LUMO, –*n* for occupied orbitals, +*n* for unoccupied orbitals). For instance, the HOMO from the job `h2o` would be written to the file `h2o_alpha_HOMO.vis`.

Table 9.42. Keywords for generating plot data

Keyword	Value	Meaning
iplotden	***0***	Do not generate electron density data
	1	Generate electron density data
	2	Generate electron density data and electron density difference data between the final SCF density and the initial guess density.
iplotspn	***0***	Do not generate electron spin density data
	1	Generate electron spin density data
iplotesp	***0***	Do not generate electrostatic potential data
	1	Generate electrostatic potential data (also sets **epn**=1).
iorb1a	***0***	Do not generate any alpha orbital data
	>0	Index of first alpha orbital for which to generate data
	`homo-`*n*	Index of first alpha orbital for which to generate data, relative to highest occupied molecular orbital (HOMO). *n* can be any positive integer.
	`lumo+`*n*	Index of first alpha orbital for which to generate data, relative to lowest unoccupied molecular orbital (LUMO). *n* can be any positive integer.
iorb2a	>0	Index of last alpha orbital for which to generate data.
	`homo-`*n*	Index of last alpha orbital for which to generate data relative to highest occupied molecular orbital (HOMO). *n* can be any positive integer.
	`lumo+`*n*	Index of last alpha orbital for which to generate data relative to lowest unoccupied molecular orbital (LUMO). *n* can be any positive integer.
iorb1b	***0***	Do not generate any beta orbital data
	>0	Index of first beta orbital for which to generate data. Ignored for restricted wave functions.
	`homo-`*n*	Index of first beta orbital for which to generate data, relative to highest occupied molecular orbital (HOMO). Ignored for restricted wave functions. *n* can be any positive integer.
	`lumo+`*n*	Index of first beta orbital for which to generate data, relative to lowest unoccupied molecular orbital (LUMO). Ignored for restricted wave functions. *n* can be any positive integer.
iorb2b	>0	Index of last beta orbital for which to generate data.
	`homo-`*n*	Index of last beta orbital for which to generate data, relative to highest occupied molecular orbital (HOMO). *n* can be any positive integer.

Table 9.42. Keywords for generating plot data (Continued)

Keyword	Value	Meaning
	`lumo+`*n*	Index of last beta orbital for which to generate data, relative to lowest unoccupied molecular orbital (LUMO). *n* can be any positive integer.
iplotalie	***0***	Do not generate average local ionization energy data.
	1	Generate average local ionization energy data.
fukui_fn	***0***	Do not calculate Fukui functions.
	1	Calculate Fukui function for system with N+**fukui_delta** electrons.
	-1	Calculate Fukui function for system with N-**fukui_delta** electrons.
fukui_delta	***0.01***	Fraction of an electron to use in Fukui function calculations.
iplotnoncov	***0***	Do not calculate data for display of noncovalent interactions.
	1	Calculate the reduced density gradient, interaction strength, and density Laplacian for display of noncovalent interaction; identify bond critical points and include them as dummy atoms with zero-order bonds in the output structure file.
plotresnoncov	***20.0***	Number of points per unit length for noncovalent interaction plots. A higher density than for other plots is required for good visualization. The length units are defined by the **iunit** keyword. The default given here is in points/bohr.
ncicut_high	***0.1***	Maximum value of the density used to visualize the interactions. Values of the reduced gradient and Laplacian are not calculated at grid points where the density is greater than this value.
ncicut_low	***0.0***	Minimum value of the density used to visualize the interactions. Values of the reduced gradient and Laplacian are not calculated at grid points where the density is less than this value.
plotres	*2.5*	Number of points per unit length. The length units are defined by the **iunit** keyword. The default given here is in points/bohr.
xmaxadj	***0.0***	Amount to adjust the box boundary on the +*x*-axis. Can be positive or negative.
xminadj	***0.0***	Amount to adjust the box boundary on the -*x*-axis. Can be positive or negative.
xadj	***0.0***	Amount to adjust the *x* dimension of the box. Half the adjustment is added to each boundary. Can be positive or negative.
ymaxadj	***0.0***	Amount to adjust the box boundary on the +*y*-axis. Can be positive or negative.

Table 9.42. Keywords for generating plot data (Continued)

Keyword	Value	Meaning
yminadj	***0.0***	Amount to adjust the box boundary on the –*y*-axis. Can be positive or negative.
yadj	***0.0***	Amount to adjust the *y* dimension of the box. Half the adjustment is added to each boundary. Can be positive or negative.
zmaxadj	***0.0***	Amount to adjust the box boundary on the +*z*-axis. Can be positive or negative.
zminadj	***0.0***	Amount to adjust the box boundary on the -*z*-axis. Can be positive or negative.
zadj	***0.0***	Amount to adjust the *z* dimension of the box. Half the adjustment is added to each boundary. Can be positive or negative.

9.6 The lmp2 Section

The **lmp2** section has two uses. The first use is for specifying electron pairs that are to be correlated, while all other electron pairs are treated at the Hartree-Fock level. This is known as a “local” LMP2 calculation. The second use, in the case where all valence electron pairs are being correlated, is to specify delocalizations for particular pairs. These two uses are mutually exclusive, and are controlled by the setting of the **idelocv** keyword. A local LMP2 calculation is requested by setting **idelocv**=1, while a delocalized LMP2 calculation is requested by setting **idelocv**=2.

The **lmp2** section should contain a line for each atom pair describing atoms to be treated at the LMP2 level. Each line describing an LMP2 pair should begin with two atom numbers or labels, which specify the two atoms in the pair. A lone pair is indicated by specifying the same atom number or label twice. All bond pairs between two specified atoms are included. Pairs can be listed in any order. On each line, any atom numbers or names following the first two indicate atoms over which the LMP2 pair is to be delocalized.

The following example is for a local LMP2 calculation in which the bond pair between atoms 6 and 9, and the bond pair between atoms 9 and 10, are treated at the LMP2 level, while all other electron pairs are treated at the Hartree-Fock level.

```
&gen
mp2=3
idelocv=1
&
&lmp2
6   9
9  10
&
```

The following example is for a delocalized LMP2 calculation in which all valence electron pairs are treated by LMP2, and in addition, the bond pair between atoms C2 and C3 is delocalized over atoms C1 and C4, and the lone pair on N3 is delocalized over all four carbon atoms.

```
&gen
mp2=3
idelocv=2
&
&lmp2
C2 C3 C1 C4
N3 N3 C1 C2 C3 C4
&
```

A common situation in which extra delocalization is desirable is in the treatment of molecules containing aromatic rings. To automate the specification of the appropriate delocalizations, you may use the **ireson** keyword instead of an **lmp2** section (you do not need to set the **idelocv** keyword either, as it will be set internally). When you use the **ireson** keyword, the molecular structure is analyzed and aromatic rings detected, and the appropriate delocalizations are defined automatically. It is safe to use the **ireson** keyword even when there are no aromatic rings in the structure, but if you are going to compare energies for different molecules, be sure to use the same **ireson** setting. You can see whether any aromatic rings have been found, and what delocalizations will be used, by looking at the Jaguar output file, where you will see a table like the following:

```
 atom1    atom2         delocalization atoms
-------- -------- -----------------------------------
 C1       C2       C3     C18    C13    C6
 C2       C3       C1     C18    C13    C6
 C3       C18      C1     C2     C13    C6
 C18      C13      C1     C2     C3     C6
 C13      C6       C1     C2     C3     C18
 C6       C1       C2     C3     C18    C13
```

A similar table is printed when you specify the delocalizations in an lmp2 section.

For QST-guided transition-state searches with LMP2 wave functions, LMP2 delocalization will automatically be performed over neighboring atoms for any bonds present in one structure and not in another, unless the input file contains the **gen** section keyword setting **idelocv**=0.

For Jaguar to properly detect aromatic rings, it needs to determine the connectivity and bond orders between atoms to create an appropriate Lewis structure. When you run Jaguar from Maestro, the connectivity and bond orders that you see for the structure in the Workspace are used, because Maestro ensures that the Jaguar input file contains a `MAEFILE` directive pointing to the Maestro structure file for the job. If you do not provide a `MAEFILE` directive, then Jaguar tries to assign connectivity and create a Lewis structure. However, for reliable detection of aromatic rings, we recommend that you use Maestro to create the Jaguar input, because you can use Maestro’s Build tools to ensure that the connectivity and bond orders for the structure are appropriate. If you run from the command line, we recommend that you still use a Maestro file to define connectivity by including a `MAEFILE` directive in your Jaguar input file.

9.7 The relativity Section

In this section you can make settings for the use of relativistic Hamiltonians. These settings control details of the Hamiltonian and approximations to it. Currently only the scalar ZORA Hamiltonian is available.

For ZORA, you can choose to use the nuclear potential only in the ZORA integrals, or the nuclear and electronic Coulomb (Hartree) potentials. Using the latter is recommended, as it includes the bulk of the electronic screening of the nucleus in the relativistic integrals. The electronic Coulomb potential is evaluated with the atomic initial guesses, and is not changed during the calculations.

Only the one-center relativistic contributions to the ZORA Hamiltonian are evaluated. This is an approximation to the full ZORA method, but is a good approximation as the relativistic effects are well localized on the atoms. Its advantages are that it eliminates the gauge dependence of the ZORA method, and it allows geometry optimizations and frequency calculations to be performed at no extra cost.

Table 9.43. Keywords for the ZORA Hamiltonian

Keyword	Value	Meaning
potential	`local`	Evaluate ZORA integrals as one-center integrals, using one-center (atomic) nuclear and electronic Coulomb (Hartree) potentials. This is an approximation, but it removes the gauge dependence of the ZORA method.
	`local_nucl`	Evaluate ZORA integrals as one-center integrals, using only the one-center (atomic) nuclear potential.

9.8 The atomic Section

The **atomic** section allows you to specify data for different atoms in a molecule. The data can include basis sets for each atom, or atomic masses for isotope calculations. You can also use the **atomic** section to define groups of atoms called “fragments”, where each fragment can then be converted to dummy atoms or counterpoise atoms or used to define a part of the system for which you want to compute a numerical Hessian. Restart files may include **atomic** sections as well, in order to keep information about charge fitting or other properties calculated previously.

In addition, **atomic** sections can be used to supply information about transition-metal-containing systems that is used to generate high-quality initial guesses for these systems. See Section 7.1.2 on page 168 for more information on using **atomic** sections in this manner.

9.8.1 General Format of the atomic Section

After the `&atomic` line, the **atomic** section should list sets of atomic input values. Each of these sets is a free-format table. The first row of the table lists the keywords whose values are to be set for each atom. This row is the column heading row. Subsequent rows list the values for the keywords for each relevant atom. For instance, in the following **atomic** section:

```
&atomic
atom  mass   vdw2
H1    2.00   1.20
H2    2.00   1.20
atom  vdw2
O     1.55
&
```

the keywords are **atom** (the atom label or number), **mass** (the nuclear mass in amu), and **vdw2** (the van der Waals radii for PBF solvation), and the lines for the atoms H1 and H2 specify that these atoms have a nuclear mass of 2.00 amu (deuterium) and solvation van der Waals radii of 1.2 Å, while the line for atom O specifies a solvation van der Waals radius of 1.55 Å for this atom. It is not necessary to list information for atoms that are to be treated in the default manner. Keywords are case insensitive. Columns can be given in any order. All entries in a row should be separated by one or more spaces or tabs, but columns do not need to be aligned.

The **atom** column must be included in every set of atomic input values. The atom identifiers can be either atom labels (such as `H1` or `O` in the example above) or atom index numbers (such as 2 for the second atom listed in the **zmat** input). *Atom label input is case sensitive.*

If you do not want to set a value for a given atom, you can use a “?” or “–” to indicate that the default value should be used. You may leave the values blank for values at the end of the row. For instance, either of the sections below has the same effect as the first **atomic** section example listed above.

```
&atomic
atom  mass   vdw2
H1    2.00   1.20
H2    2.00   1.20
O     ?      1.55
&

&atomic
atom  vdw2   mass
H1    1.20   2.00
H2    1.20   2.00
O     1.55
&
```

Atoms may be described in more than one set of atomic input values, but the same keyword cannot be used more than once for the same atom. Thus, the following syntax is supported:

```
&atomic
atom basis
C1 6-31g*
atom formal
C1 1
&
```

but the following syntax is not supported:

```
&atomic
 atom basis
 C1 6-31g*
 atom basis
 C1 cc-pVTZ
 &
```

To print an **atomic** section in the restart file that contains information for all atoms, not just some, set the output keyword **ip29** to 2. If an **atomic** section exists or if **ip29**=2 in the input file, the **atomic** section is echoed in the output from the program `pre`.

9.8.2 Keywords That Specify Physical Properties

The keywords that specify physical properties of atoms are listed and defined in Table 9.44. Values for these keywords can appear in restart files.

The **formal** keyword is useful for solvation jobs (because the van der Waals radii are adjusted according to the chemical structure found by Jaguar) and for generating an improved initial guess for transition-metal-containing systems (along with the **multip** keyword). See Section 7.1.2 on page 168 for more information on using this improved initial guess method.

Table 9.44. Keywords for physical properties in the ***atomic*** *section*

Keyword	Description
isotope	Isotopic number (integer, e.g., 2 for deuterium); overridden by atom's **mass** setting if it exists
mass	Nuclear mass in amu
esp	Electrostatic potential fitted point charge (or request to fit charge to dummy atom; see text)
formal	Formal charge (integer value) on atom
multip	Spin multiplicity of atom (or fragment containing atom)
2spin	Number of unpaired alpha or beta electrons on atom; positive value for alpha spin, negative value for beta spin.
mulk	Mulliken population
vdw	van der Waals radii (in Å) for charge fitting
vdw2	van der Waals radii (in Å) for PBF solvation. Not applicable to SM6 or SM8 solvation.
cov	Covalent radius in Å (used to determine bonding and other properties)

The **esp** keyword can be used to freeze the charge on an atom to a particular value while fitting charges to other atoms, leave an atom out of charge fitting, or fit a charge to a dummy atom. If the **esp** column entry for an atom is set to a real number, the atomic charge for that atom is held fixed to that number during charge fitting. If the **esp** column entry for an atom is set to "n" or "no" (or 0), the atom is not included in charge fitting. If the **esp** column entry for a dummy atom is "y" or "yes," it is included in the charge fit.

Several warnings apply to the use of the **esp** column. First, the esp settings must not be inconsistent with the symmetry used for the rest of the job. Second, you should be careful not to overconstrain the charge fitting job. Third, if you are including any dummy atoms in the charge fitting, it may be advisable to perform the charge fitting in a separate job (based on the restart file) for which the charge fitting grid has been altered to include points around the dummy atoms by including a **grid** column in the **atomic** section, with "y" or "yes" entries for the dummy atoms, as described below.

The van der Waals surface used for charge fitting is constructed using DREIDING [104] van der Waals radii for hydrogen and for carbon through argon, and universal force field [101] van der Waals radii for all other elements. These radii are listed in Table 9.45, and can be changed using the **vdw** keyword.

The van der Waals radii for PBF solvation calculations are listed in Table 9.46, and can be changed using the **vdw2** keyword. The radii for the elements H, C, N, O, F, P, Cl, Br, and I can be adjusted by Jaguar in some functional groups. See Section 10.6 on page 311 for more information on how Jaguar uses these radii in solvation calculations.

The van der Waals and intrinsic Coulomb radii for SM6 calculations are listed in Table 9.47. These values cannot be changed.

The covalent radii used to determine which atoms are bonded are given in Table 9.48. Two atoms are considered to be bonded if the distance between them is less than **covfac** times the sum of their covalent radii, where **covfac** is a **gen** section keyword with a default value of 1.2. The radii can be changed using the **cov** keyword. See Section 9.5.2 on page 205 for more information on how Jaguar uses and presents covalent radii and bonding information.

9.8.3 Basis, Grid, Dealiasing Function, and Charge Usage for Individual Atoms

The **basis** keyword allows you to specify the basis sets used to treat particular atoms. The string provided to describe the basis set should be chosen from the first column of the tables in Section 3.2. Lowercase or uppercase letters can be used. Polarization and diffuse functions can be added by appending *, **, +, or ++ immediately after the basis name. The meaning of these symbols is also described in Section 3.2.

Three other keywords shown in Table 9.49 allow you to specify whether to include grid points, dealiasing functions, or nuclear charges for listed atoms. The values “n,” “no,” “none,” and “only” are not case sensitive. You can use the **atomic** section to specify counterpoise atoms, and that settings in the **atomic** section take precedence over Z-matrix counterpoise input. In the **atomic** section, counterpoise atoms are indicated by using an entry of “n” in the column entitled “charge” (see Table 9.49). Also, note that any other word or letter, such as the “Y” entries that may appear in restart files, indicates that the grid, dealiasing function, or charged particles for that atom are included (the usual default for the grid, daf, and charge keywords).

Table 9.45. Van der Waals radii (in Angstroms) used in calculation of electrostatic potential (ESP) fitted charges

1 H 1.597																	2 He 1.181
3 Li 1.226	4 Be 1.373											5 B 2.042	6 C 1.949	7 N 1.831	8 O 1.702	9 F 1.736	10 Ne 1.736
11 Na 2.308	12 Mg 2.308											13 Al 2.308	14 Si 2.217	15 P 2.147	16 S 2.070	17 Cl 1.958	18 Ar 1.958
19 K 1.906	20 Ca 1.700	21 Sc 1.647	22 Ti 1.587	23 V 1.572	24 Cr 1.511	25 Mn 1.480	26 Fe 1.456	27 Co 1.436	28 Ni 1.417	29 Cu 1.748	30 Zn 1.381	31 Ga 2.192	32 Ge 2.140	33 As 2.115	34 Se 2.103	35 Br 2.095	36 Kr 2.071
37 Rb 2.057	38 Sr 1.821	39 Y 1.673	40 Zr 1.562	41 Nb 1.583	42 Mo 1.526	43 Tc 1.499	44 Ru 1.481	45 Rh 1.464	46 Pd 1.450	47 Ag 1.574	48 Cd 1.424	49 In 2.232	50 Sn 2.196	51 Sb 2.210	52 Te 2.235	53 I 2.250	54 Xe 2.202
55 Cs 2.259	56 Ba 1.851	57 La 1.761	72 Hf 1.570	73 Ta 1.585	74 W 1.534	75 Re 1.477	76 Os 1.560	77 Ir 1.420	78 Pt 1.377	79 Au 1.647	80 Hg 1.353	81 Tl 2.174	82 Pb 2.148	83 Bi 2.185			

Table 9.46. Van der Waals Radii (in Angstroms) for PBF solvation calculations (Radii can be reassigned for atoms whose radius values below are shown in bold italics, since Jaguar generally adjusts radii for atoms in certain functional groups. See Section 3.8, Section 10.6, and the default.lewis data file for more information on van der Waals radius assignments for solvation calculations.)

1 H																	2 He
1.150																	1.181
3 Li	4 Be											5 B	6 C	7 N	8 O	9 F	10 Ne
1.226	1.373											2.042	**1.900**	**1.600**	**1.600**	**1.682**	1.621
11 Na	12 Mg											13 Al	14 Si	15 P	16 S	17 Cl	18 Ar
1.491	1.510											2.249	2.147	**2.074**	1.900	**1.974**	1.934
19 K	20 Ca	21 Sc	22 Ti	23 V	24 Cr	25 Mn	26 Fe	27 Co	28 Ni	29 Cu	30 Zn	31 Ga	32 Ge	33 As	34 Se	35 Br	36 Kr
1.906	1.700	1.647	1.587	1.572	1.511	1.480	1.456	1.436	1.417	1.748	1.381	2.192	2.140	2.115	2.103	**2.095**	2.071
37 Rb	38 Sr	39 Y	40 Zr	41 Nb	42 Mo	43 Tc	44 Ru	45 Rh	46 Pd	47 Ag	48 Cd	49 In	50 Sn	51 Sb	52 Te	53 I	54 Xe
2.057	1.821	1.673	1.562	1.583	1.526	1.499	1.481	1.464	1.450	1.574	1.424	2.232	2.196	2.210	2.235	**2.250**	2.202
55 Cs	56 Ba	57 La	72 Hf	73 Ta	74 W	75 Re	76 Os	77 Ir	78 Pt	79 Au	80 Hg	81 Tl	82 Pb	83 Bi			
2.259	1.851	1.761	1.570	1.585	1.534	1.477	1.560	1.420	1.377	1.647	1.353	2.174	2.148	2.185			

Table 9.47. Atomic radii (in angstroms) used for SM6 and SM8 aqueous solvation calculations. For elements where two values for the atomic radius is listed, the first value is the van der Waals radius that, along with the solvent radius, is used to compute the solvent accessible surface area (SASA), and the second value is the intrinsic Coulomb radius that is used to compute the polarization free energy. For all other elements, the van der Waals radius and intrinsic Coulomb radius are the same. Values for the actinides and lanthanides are 2.00.

1 H 1.20 1.02																	2 He 1.40
3 Li 1.82	4 Be 2.00											5 B 2.00	6 C 1.70 1.57	7 N 1.55 1.61	8 O 1.52	9 F 1.47	10 Ne 1.54
11 Na 2.27	12 Mg 1.73											13 Al 2.00	14 Si 2.10	15 P 1.80	16 S 1.80 2.12	17 Cl 1.75 2.02	18 Ar 1.88
19 K 2.75	20 Ca 2.00	21 Sc 2.00	22 Ti 2.00	23 V 2.00	24 Cr 2.00	25 Mn 2.00	26 Fe 2.00	27 Co 2.00	28 Ni 1.63	29 Cu 1.40	30 Zn 1.39	31 Ga 1.87	32 Ge 2.00	33 As 1.85	34 Se 1.90	35 Br 1.80 2.60	36 Kr 2.02
37 Rb 2.00	38 Sr 2.00	39 Y 2.00	40 Zr 2.00	41 Nb 2.00	42 Mo 2.00	43 Tc 2.00	44 Ru 2.00	45 Rh 2.00	46 Pd 1.63	47 Ag 1.72	48 Cd 1.58	49 In 2.93	50 Sn 2.17	51 Sb 2.00	52 Te 2.06	53 I 1.98	54 Xe 2.16
55 Cs 2.00	56 Ba 2.00	57 La 2.00	72 Hf 2.00	73 Ta 2.00	74 W 2.00	75 Re 2.00	76 Os 2.00	77 Ir 2.00	78 Pt 1.74	79 Au 1.66	80 Hg 1.55	81 Tl 1.96	82 Pb 2.02	83 Bi 2.00	84 Po 2.00	85 At 2.00	86 Rn 2.00

Table 9.48. Covalent Radii (in angstroms) used to evaluate levels of interaction (such as bonds) between atoms Generally, Jaguar considers two atoms bonded if the distance between them is less than covfac times the sum of their covalent radii, where covfac is keyword with a default value of 1.2. These radius settings can be altered by making cov settings in the atomic section of an input file, as described in Section 9.8.)

1 H																	2 He
0.32																	0.93
3 Li	4 Be											5	6 C	7 N	8 O	9 F	10 Ne
1.23	0.90											0.82	0.77	0.75	0.73	0.72	0.71
11 Na	12 Mg											13 Al	14 Si	15 P	16 S	17 Cl	18 Ar
1.54	1.36											1.18	1.11	1.06	1.02	0.99	0.98
19 K	20 Ca	21 Sc	22 Ti	23 V	24 Cr	25 Mn	26 Fe	27 Co	28 Ni	29 Cu	30 Zn	31 Ga	32 Ge	33 As	34 Se	35 Br	36 Kr
2.03	1.74	1.44	1.32	1.22	1.18	1.17	1.17	1.16	1.15	1.17	1.25	1.26	1.22	1.20	1.16	1.14	1.12
37 Rb	38 Sr	39 Y	40 Zr	41 Nb	42 Mo	43 Tc	44 Ru	45 Rh	46 Pd	47 Ag	48 Cd	49 In	50 Sn	51 Sb	52 Te	53 I	54 Xe
2.16	1.91	1.62	1.45	1.34	1.30	1.27	1.25	1.25	1.28	1.34	1.48	1.44	1.41	1.40	1.36	1.33	1.31
55 Cs	56 Ba	57 La	72 Hf	73 Ta	74 W	75 Re	76 Os	77 Ir	78 Pt	79 Au	80 Hg	81 Tl	82 Pb	83 Bi			
2.35	1.98	1.69	1.44	1.34	1.30	1.28	1.26	1.27	1.30	1.34	1.49	1.48	1.47	1.46			

*Table 9.49. Keywords for listing basis, grid, dealiasing function, and charge information for individual atoms in an **atomic** section*

Keyword	Value	Description
basis	n, no, or none	Use no basis functions on atom
	basis-name	Use basis functions from specified basis set on atom
grid	n, no, or none	Do not include any grid points on atom
	only	Include grid points on atom, but no basis functions, dealiasing functions, or nuclear charge
daf	n, no, or none	Do not include any dealiasing functions on atom
	only	Include dealiasing functions on atom, but no basis functions, grid points, or nuclear charge
charge	n, no, or none	Treat atom as a counterpoise atom—do not include nucleus or electrons for atom
	only	Include nuclear charge on atom, but no basis functions, grid points, or dealiasing functions

9.8.4 Defining Fragments

You can use the **frag** keyword in the **atomic** section to specify that all atoms with the same **frag** entry be treated in the same fragment. You can then request that all the atoms in one fragment be treated as dummy atoms or counterpoise atoms, or used as the only atoms for which numerical frequencies will be calculated (where Hessian elements for other atoms are zero).

The default **frag** value for each atom is 0, meaning it is not considered part of any fragment. To assign a group of atoms to the same fragment, in the **frag** column of the atomic section, enter the same value for each atom.

To treat all atoms in a fragment as counterpoise atoms, set **icpfrag**=*fragno* in the **gen** section of the input file, where *fragno* is the integer fragment label from the **frag** column of the **atomic** section. To treat them all as dummy atoms, make the keyword setting **idelfrag**=*fragno* in the **gen** section. To compute partial frequencies for a particular fragment, make the setting **freqfrag**=*fragno* in the **gen** section of a frequency input file.

One further use of fragments is for antiferromagnetic systems, for which standard transition metal initial guesses do not work. For an antiferromagnetic system containing two metal atoms that are not bonded, you can use a **2spin** column to set up the initial guess. When the metals are within bonding distance, or when there are more than two metals, you should assign the metal atoms to separate fragments using the **frag** column of the **atomic** section. Finally, add **formal** and **2spin** values in the **atomic** section.

9.9 The hess Section

If an input file has a non-empty **hess** section, the keyword **inhess** in the **gen** section is set to 2 automatically, and a Hessian is read in from the **hess** section. Since for a Hessian $\boldsymbol{H}$, $H_{ij} = H_{ji}$, only the elements with $j \leq i$ are read in, and the program symmetrizes the matrix itself later.

Since the Hessian has dimensions of 3N×3N, where N is the number of atoms (including dummy atoms), it may be large, so files listing all elements in each row by order of rows could be unwieldy and difficult for the user to read. Therefore, the Hessian is assumed to be presented in blocks composed of five columns each (with the last block possibly having fewer than five columns, if 3N is not a multiple of five). The format used for the **hess** section is the same as that used in GAUSSIAN files or BIOGRAF (`.hes`) files. All Hessian elements for dummy atoms should be set to 0 (as they are in Jaguar output).

Each set of elements from a block of five columns should be preceded by a line containing one or more arbitrary integer labels; for instance, column labels could be convenient for keeping track of the elements when looking at the **hess** section. All of the elements within a five-column block for which *j* (the column indicator) is less than or equal to *i* (the row indicator) are then read in, one row at a time. Each row of five or fewer matrix elements starts with an arbitrary integer label; this integer is not used in the program, but can be used to label the row, for example. When the relevant matrix elements from that entire five-column block have been read in, the next block is read in the same way, until all of the matrix elements for the lower triangle of the matrix have been entered.

For example, in the unlikely event that you wanted to enter this Hessian:

11	21	31	41	51	61	71	81
21	**22**	32	42	52	62	72	82
31	**32**	**33**	43	53	63	73	83
41	**42**	**43**	**44**	54	64	74	84
51	**52**	**53**	**54**	**55**	65	75	85
61	**62**	**63**	**64**	**65**	**66**	76	86
71	**72**	**73**	**74**	**75**	**76**	**77**	87
81	**82**	**83**	**84**	**85**	**86**	**87**	**88**

you would need to enter the elements from the bottom triangle of the Hessian (shown in bold above) in the following way:

```
&hess
j
i     11
```

```
i     21     22
i     31     32     33
i     41     42     43     44
i     51     52     53     54     55
i     61     62     63     64     65
i     71     72     73     74     75
i     81     82     83     84     85
j
i     66
i     76     77
i     86     87     88
&
```

where *i* and *j* indicate integer labels not actually used by the program.

9.10 The guess and guess_basis Sections

If an input file has a non-empty **guess** section, the keyword **iguess** in the **gen** section is set to 1, and an initial guess for the wave function is read from the **guess** section. If the label **basgss**, is given, the coefficients given in the **guess** section are interpreted as coefficients of functions from the basis set specified with this label. For instance,

```
&guess  basgss=6-31g**
```

If no **basgss** setting is given or if **basgss** is set to `non_standard`, the basis set for the guess is that specified by the **basis** keyword setting in the **gen** section. You should ensure that the initial guess given in the **guess** section is for the this basis set. Otherwise, a poor or meaningless guess is obtained and the calculation might not converge. Similarly, the ordering of the basis functions within the set being used must be the same as that used for the ordering of coefficients in the **guess** section.

This next line of the section should begin with a set of coefficients describing the contribution of each function in the basis set to the first molecular orbital, and continue on with similar coefficient sets for each molecular orbital. A single line, whose content is unimportant, should precede each molecular orbital's set of coefficients. If you like, you can use this line to label the molecular orbital for your own convenience.

If you choose to write the occupied orbitals, or occupied and virtual orbitals, from one run and use them in the **guess** section for another run, you must make sure to choose a proper format. In the Output tab of the Jaguar panel, you could select Occupied or All from the Orbitals option menu and All values (f19.15, list) or All values (f8.5, list) from the Format option menu for the original run, as described in Section 3.13.2 on page 86, and the resulting orbital output could be copied from the output file into the **guess** section of the input file for the next run. Similarly, you could set the relevant orbital output keyword to 4, 5, 9, or 10 in the **gen** section of the input file for the first run, as described in Section 9.5.26 on page 261, and use the resulting output file's orbital output in the **guess** section of the input file for the next run.

A sample **guess** section for water with an STO-3G basis set follows. The oxygen is atom 1, and for each molecular orbital, coefficients for the oxygen's 1s, 2s, $2p_x$, $2p_y$, and $2p_z$ orbitals are input. The 1s coefficient for the first hydrogen atom follows, followed by the 1s coefficient for the second hydrogen.

```
&guess  basgss=sto-3g
1: orbital energy = -.20251577D+02
   .99421641D+00  .25847131D-01  .31906711D-02   .88241647D-15
   .26760209D-02  -.55838749D-02 -.55838749D-02
2: orbital energy = -.12575489D+01
   -.23376569D+00 .84444935D+00  .94117884D-01   -.39742456D-17
   .78936818D-01  .15559441D+00  .15559441D+00
3: orbital energy = -.59385470D+00
   .30846088D-09  -.13714419D-08 -.39372414D+00 .21348436D-14
   .46944485D+00  .44922200D+00  -.44922200D+00
4: orbital energy = -.45973017D+00
   .10403593D+00  -.53816730D+00 .57914834D+00  -.40089482D-14
   .48573263D+00  .29510298D+00  .29510298D+00
5: orbital energy = -.39261707D+00
   .26538042D-15  -.27636653D-14 .26424743D-14  .10000000D+01
   .56164871D-15  .78183836D-15  .26536093D-14
&
```

The **guess_basis** section is only needed if the basis set for the guess in the **guess** section is non-standard: that is, it consists of multiple basis sets. This section lists each atom for which the basis is not defined by the **basis** keyword in the **gen** section, along with the basis set for the atom. An example is given below.

```
&guess_basis
27       cc-pvtz(-f)
28       cc-pvtz(-f)
33       cc-pvtz(-f)
34       cc-pvtz(-f)
&
```

9.11 The pointch Section

The **pointch** section describes the locations and magnitudes of a set of point charges. Up to 200,000 point charges may be used.

Each line of the **pointch** section should contain four real numbers, the first specifying the point charge in atomic units, and the next three specifying its (x, y, z) coordinates in the same units used for the geometry (angstroms by default, but bohrs if the **iunit** keyword in the **gen** section is set to 0 or 2; see Section 9.5.1 on page 204 for more information).

The sample **pointch** section below puts one point charge of charge +1 at location (0, 0, 0.2) and another of charge –1 at location (0, 0, 0.4).

```
&pointch
 1.0  0  0 -0.2
-1.0  0  0  0.4
&
```

Note that point charges should *not* contribute to the value of the net molecular charge, **molchg**, given in the **gen** section.

If you include a non-empty **pointch** section in the input file for a job, the output from the program `pre` includes a table of fixed charge information describing the point charges. This table appears in the output file immediately after the molecular geometry output.

9.12 The efields Section

If you would like to calculate wave functions or molecular properties in the presence of an electric field, you may use the **efields** section to describe this field. The x, y, and z components of the electric field should be specified, in atomic units, on the same line. The requested properties will then be calculated for the molecule in the presence of this field. The scf output will also include nuclear-electric field and electron-electric field terms.

The convention used in Jaguar for electric fields is to add a term of $\boldsymbol{E} \cdot \boldsymbol{r}$ to the no-field Fock matrix, where $\boldsymbol{E}$ is the electric field and $\boldsymbol{r}$ is the electron position. The contribution due to the interaction between the field and each nucleus of position $\boldsymbol{r}_i$ and charge q_i is $-q_i(\boldsymbol{E} \cdot \boldsymbol{r}_i)$.

The **efields** section can contain more than one line, describing several different fields. In that case, the calculations for each given field will be performed in turn. Up to 100 electric fields can be specified.

9.13 The egridpt Section

If you want to calculate the electrostatic potential and the electric field from the molecular wave function at a set of grid points, you can specify the grid points in this section. The points are given as the Cartesian coordinates *x*, *y*, and *z* in angstroms, space separated, one point per line. To request the calculation, add **ielec**=1 and **negrid**=–1 in the gen section.

9.14 The ham Section

By using the **ham** section and setting the **gen** section calculation keyword **ihamtyp** to 3, you can specify the exact coefficients used to calculate the electronic energy for open shell calculations. The electronic energy is given by the equation

$$E = \sum_i f_i h_{ii} + \sum_{ij} (a_{ij} J_{ij} + b_{ij} K_{ij})$$

where the sums are over orbitals [22]. The number of electron *pairs* per orbital in each orbital i is indicated by f_i, which can be listed in the **ham** section, and the one-electron Hamiltonian for that orbital is given by h_{ii}. The terms a_{ij} and b_{ij} are coefficients which can also be specified in the **ham** section, and the J_{ij} and K_{ij} terms are Coulomb and exchange terms for pairs of orbitals i and j. Orbitals which have the same a_{ij} and b_{ij} coefficients and number of electron pairs f_i are considered to be in the same shell.

The first line in the **ham** section should indicate the number of core orbitals for the molecule. Next, each shell is described in turn. The first line of each shell description should contain two numbers, the first an integer indicating the number of orbitals in that shell, and the second a real number indicating f_i, the number of electron pairs in each orbital of that shell. The next line should contain the a_{ij} terms for any orbital in the shell, where $j < i$ and j is not a core orbital. The last line describing the shell lists all b_{ij} terms for any orbital in the shell, where $j < i$ and j is not a core orbital.

9.15 The orbman Section

The **orbman** section allows you to reorder orbitals in the **guess** section of a restart file, or to form linear combinations of orbitals. The format of the **orbman** section is as follows:

```
&orbman
hfiglcmo    i, j, α    k, l, β   end
&
```

where *i, j, k,* and *l* are integers indicating the *i*th, *j*th, *k*th, and *l*th orbitals before mixing (i.e., χ_i, χ_j, χ_k, and χ_l), and α and β are angles (in degrees) indicating the degree of mixing. The command `hfiglcmo` mixes the orbitals to form orbitals χ_i^{new}, χ_j^{new}, χ_k^{new}, and χ_l^{new} according to the following equations:

$$\chi_i^{new} = \chi_i \cos\alpha + \chi_j \sin\alpha$$

$$\chi_j^{new} = \chi_j \cos\alpha - \chi_i \sin\alpha$$

$$\chi_k^{new} = \chi_k \cos\beta + \chi_l \sin\beta$$

$$\chi_l^{new} = \chi_l \cos\beta - \chi_k \sin\beta$$

Note that an angle of 90° permutes the two orbitals, reversing the sign of one.

Each combination operation is performed independently, and the operations are performed in the order they are listed in the **orbman** section. Each rotation involving a previously altered orbital uses the new, transformed orbital generated by the earlier operations. After all manipulations have been specified, the word "end" should be included.

For UHF wave functions, the syntax is modified slightly, and the alpha and beta spin-orbitals are designated by `hfiglcmoa` and `hfiglcmob`:

```
&orbman
hfiglcmoa    i, j, α    k, l, β    end
hfiglcmob    p, q, γ    r, s, δ   end
&
```

9.16 The cdft Section

The **cdft** section specifies the constraints used to constrain charge density to certain atoms in a DFT calculation. The syntax of a constraint involves specifying the net charge on a group of atoms, and a list of atom ranges with weights, in the format

net-charge
weight1 first-atom1 last-atom1
weight2 first-atom2 last-atom2
...

Each group of atoms is specified by a lower and an upper index, where the index is the atom index in the **zmat** section. All atoms in the range (inclusive) are included in the constraint region, and the weight specifies the relative amount of charge to be localized in the group. For example,

```
&cdft
-1.0
1.0 1 3
2.0 4 5
&
```

specifies that a net charge of –1 should be distributed over atoms 1 through 5, with 1/3 of the charge on atoms 1 through 3, and 2/3 on atoms 4 through 5. Atom groups can be overlapping.

You can add multiple blocks to this section that define a net charge and a set of atom groups to distribute the charge over.

Any atoms that are not in a constraint group will end up with the leftover charge, i.e. the difference between the charge on the entire structure and the sum of the net charges in each group.

9.17 The loc_ib Section

When running a B3LYP-LOC calculation on an ionized system, the **loc_ib** section is used to specify the bond from which an electron was removed from the neutral system, if it is removed from an orbital to which at least two atoms contribute. The section contains the atom number of the two bonded atoms on which the orbital is (largely) centered, and has the following syntax, where *atom1* and *atom2* are the atom numbers:

```
&loc_ib
atom1  atom2
&
```

For example, for the ethylene cation, CH2=CH2$^+$, the atoms should be the two carbons, because the orbitals contributing to the formation of the C=C bond are the most likely to be stripped of an electron on ionization. The orbitals contributing to the formation of the C-H bonds are too low-lying in energy to contribute to ionization to the lowest state of the ion.

9.18 The echo Section

The **echo** section contains only its own label, and requests a copy of the input file to be printed in the output file.

```
&echo &
```

9.19 The path Section

The **path** section allows you to specify the execution path, which determines the order of the Jaguar (or other) programs to be run. If no **path** section exists, Jaguar will use the default path resulting from the settings in other sections of the input file.

The items listed in a path can be either Jaguar programs, UNIX commands, or other programs accessible from the executable directory. If a program or command is not accessible from the executable directory, you must specify the full path for that program, with a “/” character at the beginning of the path.

Table 9.50 gives a brief description of each Jaguar program.

Table 9.50. Individual programs included in Jaguar

Program	Description
`jexec`	Driver program for all Jaguar executables (note: inclusion of jexec in path will cause recursive Jaguar calculations)
`pre`	Reads and checks input (including path, if any), performs symmetry analysis, and calculates terms dependent on geometry (e.g., nuclear repulsion energy)
`onee`	Calculates one-electron integrals and effective core potential (ECP) contribution to one-electron Hamiltonian, when relevant
`hfig`	Calculates Hartree-Fock initial guess
`probe`	Ensures orthogonalization
`grid`	Generates grids
`rwr`	Generates $\boldsymbol{Q}$ operators
`scf`	Performs self-consistent field calculation
`ch`	Evaluates electrostatic properties (multipole moments, electrostatic potential fitting, Mulliken populations)
`lmp2dip`	Calculates dipole moments for LMP2 wave functions
`cpolar`	Finds polarizabilities and hyperpolarizabilities using coupled perturbed HF method
`polar`	Finds polarizabilities and hyperpolarizabilities using finite field method
`elden`	Calculates electron density on set of grid points
`local`	Performs localization of orbitals
`lmp2`	Performs local second-order Møller-Plesset perturbation theory calculation
`cis`	Performs CI singles and TDDFT calculation
`der1a`, `der1b`	Calculate analytic one- and two-electron first derivatives
`lmp2der`, `lmp2gda`, `lmp2gdb`	Calculate analytic one- and two-electron first derivative terms for LMP2 wave functions
`nude`	Calculates numerical second derivatives of energy (as numerical derivatives of the analytical gradient)
`freq`	Calculates vibrational frequencies and related properties
`ira, irb`	Calculate dipole derivative terms needed for calculation of IR intensities
`geopt`	Performs geometry optimization

Table 9.50. Individual programs included in Jaguar (Continued)

Program	Description
`pbf`	Solves Poisson-Boltzmann equations for solvation calculation
`solv`	Performs solvation calculation with Poisson-Boltzmann solver
`sm6`	Performs solvation calculation with SM6 model
`sm8`	Performs solvation calculation with SM8 model
`sole`	Checks solvation energy convergence
`dsolv`	Computes solvation-related gradient terms for solvated geometry optimizations
`post`	Processes files, output, etc. at end of run
`timex`	Checks CPU time for entire run

The simplest form available for the **path** section is a list of the programs to be run, as in the following example:

```
&path pre hfig grid rwr &
```

It is not actually necessary to list `pre` in paths, since the `pre` program will always be run.

If you want to run additional programs after a standard Jaguar calculation, you can use the word `path` to indicate the default path, as below:

`&path path` *executable-list* `&`

More complicated paths involve looping over programs until the last Jaguar program in the loop indicates that convergence is reached. The first program in the section of the path to be looped over is preceded by a `loop` label, and the last is followed by a `goto` label, where each of these labels is followed by the same character string. Nested loops are also allowed. The following path illustrates a loop which will cause the programs `pre`, `onee`, `grid`, and `ig` to run once, the series of programs `rwr`, `scf`, `der1a`, `rwr`, `der1b`, and `geopt` to run until the convergence criteria for geometry optimization are satisfied, and the program `post` to run once.

```
&path pre onee grid hfig loopa1 rwr scf der1a rwr der1b geopt gotoa1 post &
```

If you put a `jump` label between a `loop` label and a `goto` label, where `jump` is followed by the same character string that follows `loop` and `goto` (`jumpa1` for the above path, for instance), the path will jump to the end of the loop after the `goto` label, and will exit the loop, when the `jump` label indicates that the convergence criterion for that program is reached.

Note that since loops only exit when convergence is reached, the program before a `goto` or `jump` label must have such a criterion. The three programs that can precede a `goto` or `jump` label are `scf` (when it is being used for solvation runs), `geopt`, and `nude`.

Sometimes you might want a path to include a command of more than one word—for instance, you might want to use the UNIX command `mv` *old-filename new-filename* to rename a file. In that case, you can enter the **path** section in such a way that each line contains a single command. To enter the path this way, you must include the word `line` after the `&path` (or `$path`) label at the beginning of the **path** section.

9.20 NBO Sections

To request a Natural Bond Orbital (NBO) analysis at the end of the Jaguar job, set `nbo = 1` in the **gen** section of the input file. This setting runs the NBO program with `NRT` and `PRINT=3` set. `PRINT=3` is the same as setting the NBO keywords `NLMO`, `DIPOLE`, and `BNDIDX`. If you need finer control over the behavior of the NBO program, you can use an **nbo** section instead of the `nbo` keyword. If the **nbo** section is empty, as it is here:

```
&nbo &
```

a basic NBO analysis is performed, without `NRT` or `PRINT=3`. Options for NBO calculations that are specified in the `$NBO` keylist can be included in the **nbo** section, in the format required by the NBO program. Likewise, if you want to specify options for the `$CORE`, `$CHOOSE`, or `$NRTSTR` keylists, you should include them in **core**, **choose**, and **nrtstr** sections in the Jaguar input file. The information in all these sections is copied unchanged into NBO keylists of the same name and passed to the NBO program. Jaguar's interface to NBO 6.0 does not support the `$DEL` keylist, which means that Natural Energy Decomposition Analysis (NEDA) is not supported. The `$DELH` keylist is also not supported.

If you sets **nboden**=1 and **mp2**=3 in the **gen** section, the LMP2 density is used in the NBO analysis. If **nboden**=0 or is not set at all, the HF density is used. NBO calculations require an SCF calculation to be performed first.

For more details on NBO input and output, see the *NBO 6.0 Manual*, or visit the NBO web site, http://nbo6.chem.wisc.edu/.

Other Jaguar Files

Jaguar needs certain types of files in order to run a job. An input file must be created, of course, but additional files specifying the basis functions, data for the initial guesses, dealiasing functions, grids, and cutoffs used during a run are generally necessary as well. Unless other files are specified in the input data, Jaguar uses the files `default.basis`, `default.atomig`, `default.daf`, `default.grid`, and `default.cutoff`, which are in the `data` directory. For many solvation calculations, Jaguar also uses the file `pbf_radii.ark`. All of these files are provided in the Schrödinger product distribution.

If you want to use other data files than those described above, you can put files whose names and formats match those described above in the directory that contains your input files. When you run a job, you can edit the input file and add `BASISFILE`, `ATOMIGFILE`, `DAFFILE`, `GRIDFILE`, `CUTOFFFILE`, or `LEWISFILE` lines with the names of the files you want to use. See Section 9.1 on page 195 for more details. You should avoid naming a `.cutoff` file `accurate.cutoff`, `quick.cutoff`, or `solvent.cutoff`, because the program assumes you are using an outdated file and resets the name to `default.cutoff`.

This chapter contains descriptions of the basis, atomic initial guess, dealiasing function, grid, cutoff, and Lewis files. Even if you do not plan on creating your own versions of these files, you might want to skim this chapter if you are curious about the methods used in Jaguar.

10.1 The Basis Set File

The basis sets available for use in Jaguar appear in the file `default.basis`, in the standard data directories. Portions of this file are shown in this section; you might want to refer to them as you read the description of the file. Basis sets at the top of the file do not contain effective core potentials, and will be described first. The basis sets with effective core potentials will be described later.

10.1.1 Basis Set Format

Each basis set description begins with a blank line. The next line (or lines) must begin with the word `BASIS`, followed by one space. That label is followed by one or more names of the basis set to be described: the name of the basis set as given in Table 3.1 on page 34 or Table 3.2 on page 36, and any other names which describe the same basis set (e.g., STO-3G and STO3G). The basis set names are separated by commas, and must include * and/or + characters, if those are allowed for that basis set. (** or ++ character strings are sufficient to describe the * and +

cases also, and the * characters can be listed either before or after the + characters.) The next notation in the line, “5D” or “6D,” sets the default number of functions for d shells when using that basis set, as described in Section 3.2 on page 33.

“Backup” basis set names, which are each preceded by the word `BACKUP`, may follow on the same line. If any sets are listed after the word `BACKUP`, it indicates that if an atom is not found in the current basis set, its basis function will be obtained from the list of backup basis sets. If there is more than one backup name listed, the basis function for the atom comes from the first backup set listed that contains that atom. Note that the numbers of d shells specified in the backup basis sets is ignored. Also, polarization or diffuse functions are chosen according to the basis set specified by the calculation; that is, *, **, +, or ++ options on backup basis sets are ignored if they do not agree with the options on the basis set chosen for the calculation.

The basis set description continues with a set of lines describing the basis functions on each atom. The information for each atom begins with a line containing the element symbol (e.g., He). The atomic symbol must not be preceded by any spaces or characters. The next line begins with the type of function (S, P, or D, for instance). If this label is SP, the corresponding set of data describes an s *and* a p function whose Gaussians have the same exponents. The next number in that line is the polarization/diffuse function parameter. If it is 1, it indicates a polarization function which is included in the basis set if the basis set name ends in an *, as described in Section 3.2 on page 33. If the number is 2, it indicates a ** basis set function; if –1, a + basis set function; if –2, a ++ basis set function. Otherwise, the number should be 0.

The rest of the numbers on that line determine the way that Jaguar will contract some of the functions, and the “range” of each function. The numbers before the dash (–) describe how many of the functions are included in that contraction. For example, if there were two such numbers, 2 and 1, the line would indicate that Jaguar would contract the first two Gaussians provided immediately below into one contracted function, and would treat the third Gaussian as an uncontracted function.

If you want to add or change a basis set to a `.basis` file, you should probably contract together all Gaussians whose exponents are greater than 0.3. The `default.basis` information generally follows this rule, although there are some exceptions (see the Li s and p function information in the sample file below for an example).

The numbers after the dash describe the range of each such function. There should be one such number for each contraction number before the dash. A zero indicates that the contracted function will be treated as a long-range function, while a 1, 2, 3, or 4 indicate various types of short-range functions. These assignments help determine the symmetrization of the Fock matrix components by the “side choosing” method described in Ref. 13. These range values are only used in pseudospectral calculations, so if your basis set will be used for non-pseudospectral calculations, use 0 as a place holder for each range value. Pseudospectral calculations require that grids and dealiasing functions exist for the basis set. These are defined in the `default.grid` and `default.daf` files, respectively; see below.

The Gaussians in the contraction are listed next, with the first number in each of these lines describing the exponent for the Gaussian, and the second its coefficient in the contraction. The Gaussians should be listed in decreasing size of exponent. If both s and p functions are being described, the second number on the line corresponds to the coefficient for that Gaussian in the s function's contraction, and the third number corresponds to the p function's contraction coefficient. The data for that atom ends with a line containing 4 * characters, with no spaces or other characters preceding them.

When all of the atoms for that basis set have been listed, ending with the obligatory **** line, the next basis set is listed, in the same manner described above.

The beginning of the `default.basis` file is shown below to illustrate most of these points.

```
BASIS STO-3G*,STO3G*,STO-3*,STO3* 5D
H
S    0       2  1   -   1  0
     3.42525091400000        0.154328967294599
    0.623913729800000        0.535328142281266
    0.168855404000000        0.444634542184440
****
He
S    0       3   -   2
     6.36242139400000        0.154328967291452
     1.15892299900000        0.535328142270350
    0.313649791500000        0.444634542175373
****
Li
S    0       3   -   4
     16.1195747500000        0.154328967293323
     2.93620066300000        0.535328142276839
    0.794650487000000        0.444634542180763
SP   0       1  2   -   1  0
    0.636289746900000       -9.996722918659862E-02    0.155916274998087
    0.147860053300000        0.399512826086407        0.607683718592546
    4.808867839999999E-02    0.700115468876179        0.391957393095192
****
```

10.1.2 Effective Core Potential Format

Basis sets containing effective core potentials (ECPs) are described in a slightly more complicated fashion. First, the string `ECP` must appear between the `5D` or `6D` label and the `BACKUP` label. This string indicates that the basis set description contains information about the effective core potential associated with the basis set.

As for the basis sets without effective core potentials, each atom in the set is described in turn. The description begins with the basis function, which is in the same format as those described above. After a line with two asterisks (**), the effective core potential is described.

The first line in the effective core potential description contains the element symbol (e.g., Na) and two numbers. The first number is the maximum angular momentum in the core, and the second gives the number of electrons replaced by the effective core potential. Next, the information for various angular projectors is listed. The first set of information contains the local components of the ECP and should begin with a line starting `D_AND_UP`, `F_AND_UP`, or `G_AND_UP`, which indicates that the maximum angular momentum to be described is 2, 3, or 4.

Following that line, the different terms for this angular projector are given. Each line describes a term of the form $Ce^{-\alpha r^2}r^{n-2}$, listing the parameters n, α, and C (from left to right) in a free format. Next, the angular projectors are listed in increasing order (S, P, D, etc.) in the same fashion.

A line with four * characters appears the end of the description of each atom's ECP. When all atoms have been described in turn, the next basis set is described.

The example below shows the beginning of the description in `default.basis` of the LAV2D and LAV2P basis sets. Note that these basis sets only differ in their choice of what basis set to use for atoms that are *not* described by the effective core potential.

```
BASIS LAV2D**,LANL1DZ** 5D ECP BACKUP D95V**
BASIS LAV2P** 5D ECP BACKUP 6-31G**
Na
S    0      1  1   -   1  0
    0.497200000000000       -0.275357400000000
    5.600000000000000E-02     1.09899690000000
S    0      1   -   0
    2.210000000000000E-02     1.00000000000000
P    0      1  1   -   1  0
    0.669700000000000       -6.838450000000000E-02
    6.360000000000000E-02     1.01405500000000
P    0      1   -   0
    2.040000000000000E-02     1.00000000000000
**
Na   2  10
D_AND_UP
1   175.55025900   -10.00000000
2    35.05167910   -47.49020240
2     7.90602700   -17.22830070
2     2.33657190    -6.06377820
2     0.77998670    -0.72993930
S-D
0   243.36058460     3.00000000
1    41.57647590    36.28476260
```

```
2    13.26491670    72.93048800
2     3.67971650    23.84011510
2     0.97642090     6.01238610
P-D
0  1257.26506820     5.00000000
1   189.62488100   117.44956830
2    54.52477590   423.39867040
2    13.74499550   109.32472970
2     3.68135790    31.37016560
2     0.94611060     7.12418130
****
```

10.1.3 Customizing Basis Sets

If you want to set up your own `.basis` file, you can do so, if you use the format described above. Generally, *you must also create an altered version of the* `.atomig` *file*, which is described in Section 10.2, although if you are just adding polarization functions to the basis set, and these functions are identified by the polarization/diffuse function parameter described earlier in this section, you can continue to use the usual `.atomig` file. Make sure your new `.basis` file contains the 6-31G basis set, because the initial guess program needs this basis set. If you alter the basis functions in the `default.basis` file only slightly, you can use the same names for the basis sets. If you change them a great deal, you should use a new name, so that Jaguar will not attempt to use grids or dealiasing functions that do not match the new basis set. If you change a basis set name to something Jaguar does not recognize, runs using that basis set will use all-analytic methods (see Section 3.7.1 on page 54 or the information on the input file **gen** section keyword **nops** in Section 9.5.19 on page 245).

To use the file in a Jaguar calculation, you must add a line in the form

`BASISFILE:` *filename*

to the input file for the job. The file name must not include a path, and the file must be in the same directory as your input file.

To make it easier to add basis sets to Jaguar, a script called `makejbasis` has been provided that converts basis sets in GAUSSIAN 94 format, as downloaded from the PNNL web site, into Jaguar format. The basis set download page of the PNNL web site is at

https://bse.pnl.gov/bse/portal

When you download the basis sets, you must save the data in text format, not HTML format.

The syntax of the `makejbasis` command is

`$SCHRODINGER/utilities/makejbasis -b` *input-filename* `-o` *output-filename* where *input-filename* is the name of the GAUSSIAN 94 format data file, and *output-filename* is the name of the Jaguar format basis set file. The script is a Perl script. If Perl is not installed in `/usr/bin`, you can run this script by prefacing the command with `perl`.

Because Jaguar currently cannot use g or higher basis functions, basis functions with angular momentum g or higher are removed from the basis set and a warning is displayed. If a basis set contains an ECP with h or higher potential (projectors with angular momentum g or higher), the *entire* basis set for that element is not converted, and a warning is displayed. The reason for discarding the entire basis set is that the ECP is not valid for molecular calculations if some projectors are removed from the ECP.

The script does not automatically distinguish polarization or diffuse functions from regular basis functions. If polarization or diffuse functions are included in the basis set, and you want to be able to select them by using '*' or '+', then you must edit the output from the script and add the appropriate data to mark the basis function as a polarization or a diffuse function as described on page 294. Otherwise Jaguar treats them as part of the standard basis set, as it does for cc-pVTZ, for example.

Note: Any basis sets you add will only be available for non-pseudospectral calculations, if you do not also add the associated grids and dealiasing functions.

10.2 The Initial Guess Data File

The file `default.atomig` contains the results of Hartree-Fock calculations on atoms for various basis sets. By default, the initial guess is constructed from wave functions in this file. When the basis set to be used for the calculation is STO-3G, 3-21G, 6-31G, MSV, TZP, TZP(F), LAV2P, LAV2D, LAV3P, LAV3D, LACVP, LACVD, LACV3P, CSDZ, or ERMLER2 (or any variant of these sets involving polarization or diffusion functions, such as 6-31G*), the initial guess is formed from wave functions obtained from atomic calculations in that basis set (ignoring polarization and diffusion functions). *Therefore, if you change the* `.basis` *file, you need to change the* `.atomig` *file correspondingly, and vice versa.*

For other basis sets, the wave functions used to construct the initial guess are obtained by projecting the appropriate atomic wave function in `default.atomig` onto the basis set actually being used for the molecular calculation. The 6-31G wave function is used whenever possible; when a 6-31G atomic wave function is not listed for a particular atom, the MSV wave function is used for that atom. For atoms beyond Xe in calculations using the LAV1S basis set, the LAV2P atomic results are used. It is important to add an initial guess if you add an ECP basis set, as projection onto another guess is likely to give bad results.

The overall structure of the initial guess data file is as follows. As in the `default.basis` file, the basis sets are listed in turn, and for each basis, the information for each atom is listed.

`BASIS` *basis-name-list*
atom-block
atom-block
`...`
atom-block

`BASIS` *basis-name-list*
atom-block
atom-block
`...`
atom-block
`...`

Each basis set section begins with a blank line The basis sets for this section are defined with on the `BASIS` line, where the names of all basis sets to which the following atomic sections apply are listed, separated by a space and/or comma. The basis set names are listed in Table 3.1 on page 34 and Table 3.2 on page 36.

Each atom block in the basis set section has the following structure:

atom-symbol comment
num-basis-functions num-ECP-core-electrons
orbital-index fractional-occupation orbital-energy
coefficient-list
orbital-index fractional-occupation orbital-energy
coefficient-list
`...`
`****`

The first line lists the atomic symbol for the atom, followed by information which is simply a comment and is not read in. The second line gives the number of basis functions for that atom and basis set, as listed in the `default.basis` file, followed by the number of electrons for that atom that are included in an effective core (0 for the basis sets listed in Table 3.1 on page 34).

Following these lines, the orbitals for the atom are specified. The first line in an orbital specification gives the orbital index (1 for the first orbital, 2 for the second, and so on), the fractional orbital occupation (1.0 if the orbital is fully occupied, 0.5 for a half-filled shell, and so on), and the orbital energy in hartrees. The orbitals should be listed in ascending order of orbital energy. The coefficients of each basis function for that orbital follow, in the order in which the basis functions are given in the `default.basis` file. The basis functions are taken to be normalized to 1. The coefficients must be given for the components of each shell: for example, there must be three coefficients for a p shell, in the order p_x, p_y, p_z. You can spread the coefficients over multiple lines. You can also include low-lying virtual orbitals with an occupation number of 0.0, such as the 2p orbital for Li or Be. The atom block is terminated by four stars.

You can create your own initial guess file using the format described above. To use it in a Jaguar calculation, you must add a line in the form

`ATOMIGFILE:` *filename*

to the input file for the job. The file name must not include a path, and the file must be in the same directory as your input file. The file name is arbitrary, but you might want to use the suffix `.atomig` to mark it as an initial guess file.

If you need to generate the atomic wave functions for your basis set to add to the default initial guess file, `default.atomig`, or to your own initial guess file, you must run a series of atomic calculations, one for each element covered by the basis set, using the `atguess=1` setting in the **gen** section. Set the basis keyword appropriately, but do not include any polarization or diffuse functions, as these are not occupied for the neutral atom. Do not set any other keywords. There is no need to set the multiplicity, because all valence orbitals will be equally (and usually fractionally) occupied. After the job runs, copy the *jobname*`.ig` file to the appropriate basis set entry in the `default.atomig` file or your own initial guess file.

10.3 The Dealiasing Function File

When Jaguar fits a function's grid point values to a basis set to find the applicable basis set coefficients for the function, it uses dealiasing functions to reduce errors. The dealiasing functions span the function space determined by the grid more completely than the basis functions, so a function on the grid can be better described using the dealiasing functions than by the basis functions alone. The basis set coefficients for the function can then be determined by using the overlap between the dealiasing functions and the basis set functions, which is determined analytically.

Some basis functions die off slowly and require long-range functions centered on each atom in the molecule, while others die off quickly over distance and can be described with short-range dealiasing functions centered on the nearby atoms. The latter type can employ different dealiasing functions, depending on the distance between the atom upon which the relevant basis function is centered and the atom upon which the short-range dealiasing functions are to be centered. If the atoms are the same, "home atom" dealiasing functions are used; otherwise, the distance between the two atoms determines whether the dealiasing functions used should be those for first-order or one of the other higher-order neighbors.[1] If the two atoms are further away than the farthest neighbor range specified, no dealiasing functions on one atom are used in calculating the contribution of a short-range basis function centered on the other atom.

The dealiasing functions themselves are simply Gaussian basis functions: s-type, p-type, and so on. Uncontracted dealiasing functions are formed by specifying the exponent of the

1. To see this connectivity information for a system, set **ip12** = 2 in the **gen** section.

Gaussian function. Contracted dealiasing functions are defined as linear combinations of the appropriate type of functions; the coefficients and exponents for the linear combination are the same as those used in the basis set for the contracted basis functions for the relevant function types (1s, 2s, $2p_x$, etc.). Thus, a dealiasing uncontracted function can be specified with the type (s, p, d, etc.) and the exponent desired for the Gaussian, while a contracted Gaussian function can be specified with the type and a reference to the desired set of contraction coefficients and exponents.

Section 10.3.1 below describes the file that determines the dealiasing functions for a calculation. Sets of dealiasing functions must be provided for each grid used in the calculation. Comments about a sample file refer to the sample `.daf` file in Section 10.3.2.

10.3.1 File Format and Description

The first line of a dealiasing function file contains a character string which includes the version number of Jaguar. This string should be “dafv” followed immediately by four digits giving the version number times 100. Lead zeros are added if necessary.

The second line has two integers. The first integer gives the number of dealiasing function sets provided for each atom type; each set is used for a particular grid during the calculation. The ordering of the sets used for each grid type is determined by the parameters named **dcoarse**, **dmedium**, and so on, which are specified in the **gen** section of the input file. By default, the coarse grid is listed first, then the medium, fine, ultrafine, and gradient grids, in that order.

The second number in the second line gives the number of ranges described in each of these dealiasing function sets. The ranges correspond to particular RwR blocks for the calculation. One of these ranges is the long range, basically covering the whole molecule; another is the home atom range, which actually only includes the relevant atom itself; and the rest are increasingly large neighbor ranges. The number of ranges should currently not exceed 10. The sample file’s second line indicates that for each basis set, five dealiasing function sets are specified for each atom, and that each of these sets contains dealiasing functions for a total of six ranges: the long-range functions, the functions for the home atom, and the functions for four other neighbor ranges.

The distances defining the neighbor ranges are set in the next line of real values, in units of bohr. Note, however, that generally only the third neighbor range is actually used. The first distance specifies that if the basis function whose coefficient is being evaluated is to be approximated by short-range dealiasing functions, then the dealiasing functions for first-order neighbors will be used for each atom within this distance of the atom upon which the basis function is centered (except for the basis function atom itself, for which the home atom dealiasing functions will be used). The second distance defines which atoms are considered second-order neighbors to each other, and so on. Since the number of neighbor ranges includes not only these ranges but also the long range over the entire molecule and the home atom range consisting of the relevant atom itself, the number of neighbor ranges actually specified in this line of the `.daf` file should be two less than the number of ranges listed in the previous line. Thus, in the sample file, the distances listed specify the neighbor ranges for first- through fourth-order neighbors.

The rest of the `.daf` file contains the dealiasing function sets for each atom type within each basis set. The data for each basis set should begin with a line listing the basis set name (as listed in Table 3.1 on page 34 and Table 3.2 on page 36), including the “*” characters indicating the polarization functions (e.g., 6-31G**). The first line for each atom type for that basis set should list three integers: the atomic number for that atom type, the number of uncontracted dealiasing functions about to be listed for each neighbor range in each set, and the corresponding number of contracted dealiasing functions. In the sample file, the first atom whose dealiasing functions are listed is hydrogen, since the atomic number listed is 1. The same line says that ten uncontracted functions and two contracted functions will be specified for each range in the five sets of dealiasing functions for hydrogen.

The second line for the same atom type should list real dealiasing exponents for each uncontracted function. The exponents specify what functions can be used. For instance, in the sample file, hydrogen’s s-type uncontracted basis function from the first exponent would be $N_1 e^{-.040634 r^2}$, while the p-type uncontracted basis function for the same exponent would be $N_2 r e^{-.040634 r^2}$. N_1 and N_2 are normalization constants.

Below those two lines, the dealiasing function sets for that atom type should be listed set by set. By default, the first set will be used for the coarse grid, the second for the medium grid, and so on, with the last set corresponding to the gradient. This ordering can be changed in the **gen** section of the input file. Each set should contain a line for each neighbor range; the long-range functions should be specified first, then the home atom functions, then the functions for each neighbor range, in increasing order. Within each line, there should be several integers, one for each uncontracted function, then one for each contracted function. These integers dictate how to construct the actual functions from the exponents (just given in the `.daf` file for uncontracted functions, and already established in the `.basis` file for contracted functions) and contraction coefficients for contracted functions (also established in the `.basis` file). If the value is 1, an s-type function will be constructed using the relevant exponent or exponents; if 2, a p-type function; if 4, a d-type function; if 8, an f-type function; and if 16, a g-type function. To construct more than one of these types of functions with the same exponent or exponents, the relevant numbers should be added together (for instance, 1 + 2 + 4 = 7 for s, p, and d).

The exponent or exponents for each of these functions are determined by the position of the entry in the row. The uncontracted functions are described first, in the same order as their exponents were listed earlier, and the contracted functions corresponding to the *contracted* functions found in the `.basis` file are described next, in the same order as in the basis set file.

Uncontracted functions in the basis set file should be ignored. Finally, the first derivatives of the basis set file contracted functions will be calculated, and the values listed for these “extra” functions correspond to the functions generated this way, in order of the function they were generated from and, within that order, of increasing complexity (s before p, etc.). For instance, if the basis set contained contracted functions for 1s, 2s, and 2p orbitals, the derivatives would be listed in the following order: a p-type function resulting from the derivative of the 1s function, a p-type function resulting from the derivative of the 2s function, an s-type function resulting from the first term of the derivative of the 2p function, and a d-type function resulting from the second term of the derivative of the 2p function.

The last six lines of the sample `.daf` file correspond to the gradient dealiasing function set for He (note that the atomic number specified for those five dealiasing function sets was 2). The first line of this set describes this set’s long-range dealiasing functions centered on the He atom, which will be used when coefficients for long-range basis functions are to be calculated, as explained above. The second value on this line, 3, dictates that uncontracted s-type and p-type (1 + 2 = 3) basis functions are to be constructed using the second exponent provided for this atom (0.145957). The second line of the set, which describes this set’s He-centered dealiasing functions to be used when calculating the coefficients for He-centered short-range basis functions (the home atom line of the set), has a value of 1 entered in the eleventh column, meaning that an s-type contracted function will be calculated using the exponents provided for the first contracted function for He in the basis set. Since this basis set only provides one contracted function for He, the 1s function, whose derivative is a p-type function, the last number entered on that line (2) dictates that a p-type function be constructed, using the contraction coefficients and exponents that correspond to that derivative function, as explained in the previous paragraph.

10.3.2 Sample File

The following sample `.daf` file lists the dealiasing set for H and He for a 6-31G** basis set. Blank lines may be added for readability, and data may be spread over multiple lines.

```
dafv0300
5  6              <-- number of sets/atom, number of rows/set
3.0 5.0 7.0 9.0  <-- neighbors cutoffs distances (neighbors = row# - 2)
BASIS 6-31G**

 1 10 2 (H)
0.040634 0.080953 0.161278 0.321306 0.640122 1.275283 2.540684 5.061679 10.084136
1.100000

 0  3  3  3  0  0  0  0  0   0   0  0
 0  0  7  0  0  5  0  0  0   2   1  0
 0  0  7  3  2  5  0  0  0   2   1  0
 0  0  3  0  2  1  0  0  0   2   1  0
 0  0  0  0  0  0  0  0  0   0   0  0
 0  0  0  0  0  0  0  0  0   0   0  0

 0  3  3  3  0  0  0  0  0   0   0  0
 0  0  7  0  0  5  0  0  0   2   1  0
 0  0  7  3  2  5  0  0  0   2   1  0
 0  0  3  0  2  1  0  0  0   2   1  0
 0  0  0  0  0  0  0  0  0   0   0  0
 0  0  0  0  0  0  0  0  0   0   0  0

 0  3  3  3  0  0  0  0  0   0   0  0
 0  3  7  0  0  5  0  0  0   2   1  0
 0  3  7  3  2  5  0  0  0   2   1  0
 0  0  3  0  2  1  0  0  0   2   1  0
 0  0  0  0  0  0  0  0  0   0   0  0
 0  0  0  0  0  0  0  0  0   0   0  0

 0  3  3  3  0  0  0  0  0   0   0  0
 0  3  7  0  0  5  0  0  0   2   1  0
 0  3  7  3  2  5  0  0  0   2   1  0
 0  0  3  0  2  1  0  0  0   2   1  0
 0  0  0  0  0  0  0  0  0   0   0  0
 0  0  0  0  0  0  0  0  0   0   0  0

 0  3  3  3  0  0  0  0  0   0   0  0
 0  3  7  0  0  5  0  0  0   2   1  2
 0  3  7  3  2  5  0  0  0   2   1  2
 0  0  3  0  2  1  0  0  0   2   1  2
 0  0  0  0  0  0  0  0  0   0   0  0
 0  0  0  0  0  0  0  0  0   0   0  0

 2 10 2 (He)
0.071497 0.145957 0.297964 0.608279 1.241774 2.535023 5.175131 10.564786 21.567514
1.100000


 0  3  3  3  0  0  0  0  0   0   0  0
 0  0  7  0  0  5  0  0  0   2   1  0
 0  0  7  3  2  5  0  0  0   2   1  0
 0  0  3  0  2  1  0  0  0   2   1  0
 0  0  0  0  0  0  0  0  0   0   0  0
 0  0  0  0  0  0  0  0  0   0   0  0

 0  3  3  3  0  0  0  0  0   0   0  0
 0  0  7  0  0  5  0  0  0   2   1  0
 0  0  7  3  2  5  0  0  0   2   1  0
 0  0  3  0  2  1  0  0  0   2   1  0
 0  0  0  0  0  0  0  0  0   0   0  0
 0  0  0  0  0  0  0  0  0   0   0  0
```

```
0  3  3  3  0  0  0  0  0   0   0  0
0  3  7  0  0  5  0  0  0   2   1  0
0  3  7  3  2  5  0  0  0   2   1  0
0  0  3  0  2  1  0  0  0   2   1  0
0  0  0  0  0  0  0  0  0   0   0  0
0  0  0  0  0  0  0  0  0   0   0  0

0  3  3  3  0  0  0  0  0   0   0  0
0  3  7  0  0  5  0  0  0   2   1  0
0  3  7  3  2  5  0  0  0   2   1  0
0  0  3  0  2  1  0  0  0   2   1  0
0  0  0  0  0  0  0  0  0   0   0  0
0  0  0  0  0  0  0  0  0   0   0  0

0  3  3  3  0  0  0  0  0   0   0  0
0  3  7  0  0  5  0  0  0   2   1  2
0  3  7  3  2  5  0  0  0   2   1  2
0  0  3  0  2  1  0  0  0   2   1  2
0  0  0  0  0  0  0  0  0   0   0  0
0  0  0  0  0  0  0  0  0   0   0  0
```

10.4 The Grid File

The grid input file (`.grid` file) determines the grids used during the calculation. Each grid type, for example, “coarse” or “ultrafine,” is constructed from grids assigned to each atom in the molecule. For any basis set for which the pseudospectral method is used, the grid file must contain grids for each grid type used, where each of these grid types in turn requires atomic grids for each element in the molecule. Grids can be assigned to grid types in the input file using the **gen** section keywords **gcoarse**, **gmedium**, and so on.

10.4.1 File Format and Description

The first line of a `.grid` file contains a character string which includes the version number of Jaguar. This string should be `gridv` followed immediately by four digits giving the version number times 100. Leading zeroes are added if necessary.

The next line should consist of an integer which gives the number of grid types described in the file. For instance, this number would be six if the grids specified were of the types coarse, medium, fine, ultrafine, eldens (for electron density calculations), and gradient. By default, Jaguar uses the coarse grid for electron density calculations and the ultrafine grid for gradient calculations, and the “extreme” grid is included for testing purposes, so the number of grid types in the file `default.grid` is actually five. Jaguar uses the grids upon each atom in the molecule provided by the `.grid` file to generate molecular grids.

All grids for each basis set are then listed in turn. The basis set is identified with a `BASIS` line and containing its name, and is followed by a blank line.

Each molecular grid description starts with two comment lines, usually a blank line followed by a descriptive line. The next line contains an integer flag which determines which points from the atomic grids for the atoms in a molecule are included in the molecular grid. Jaguar generates a boundary plane between the two atoms and perpendicular to the vector between them, disposing of any points from one atom that are on the other atom's side of the boundary plane. The integer flag determines the location of this plane: if the flag is 0, the plane is located so that the ratio of the distances of the atoms to the plane is the same as the ratio of their covalent radii, while if it is –1, the boundary plane is set where the grid point density from each atom, on the vector between the atoms, is equal. The grid point density is determined as a spline fit of the density for each shell, where each shell's density is determined as the number of points for that shell divided by the shell volume, which is the volume between the spheres whose radii are the average of the current and previous shell radii, and the current and following shell radii.

After the flag for the grid, information for each atomic grid is provided. The first line of each atomic grid section contains two integers, one providing the atomic number for that atom and the other giving the number of shells to be described. Currently, this second number should be 30 or less. The next line contains that number of entries defining the radial shell spacing, listing the radius of each shell in bohr. Grid points for that shell will be placed at that radius, in a pattern determined by the integers given in the third line. This last line of integers represents the density of the angular grid for each shell. The values are explained below.

The `default.grid` file for Jaguar version 9.0 begins as follows:

```
gridv0220
5 24

BASIS 6-31G

coarse grid
-1

1 6
0.23021 0.71955 1.74518 2.82595 3.94135 6.40743
1 3 7 7 3 1

2 7
0.20699 0.45860 0.97184 1.61794 2.40119 3.26487 5.20964
1 3 7 9 7 3 1
3 7
0.59584 1.69094 3.39571 5.30494 7.49262 11.30338 16.61803
1 3 7 9 7 3 1
```

Blank lines have been added between atomic grids for readability. Data may be spread over multiple lines.

As explained above, the beginning of the `default.grid` file indicates that five grid types are listed for each atom (corresponding to the coarse, medium, fine, ultrafine, and gradient grids. All coarse grids for 6-31G (with or without the polarization functions indicated by the **) will set the boundary plane between atoms (described earlier) at the point where the grid point densities are the same for the two atoms, because of the "–1" flag. Next, seven shells apiece are specified for H (atomic number 1), He (atomic number 2), and Li (atomic number 3). The actual `default.grid` file continues with a list of coarse atomic grids for the other atoms in the basis set, followed by the medium, fine, and ultrafine atomic grids in the same format, before proceeding to define the grids for another basis set in the same manner.

The possible values of the numbers on the angular grid line are listed in Table 10.1, along with the corresponding number of points per angular shell and the degree of the highest spherical harmonic which the grid integrates exactly, when relevant. The full references are provided in a section beginning on page 375.

Table 10.1. Number of points per angular shell and degree of the highest spherical harmonic exactly integrated by grids specified by various entries on the angular grid line

Entry	Points	Degree	Reference for Grid
1	6	3	Un 3-1 (Stroud), p.294 [205]
2	8	3	Un 3-2 (Stroud), p.294 [205]
3	12	3	U3 3-1 (McLaren), p.296 [205]
4	14	5	Un 5-2 (Albrecht & Collatz), p.294 [205]
5	18	5	Un 5-1 (Albrecht & Collatz), p.294 [205]
6	18	5	Un 5-1 (Albrecht & Collatz), p.294 [205]
7	24	5	Un 5-4 (Stroud), p.295 [205]
8	26	7	Un 7-1 (Albrecht & Collatz), p.295 [205]
9	38	9	9.1 (Lebedev) [206]
10	38	9	9.1 (Lebedev) [206]
11	42	9	9.2 (Lebedev) [206]
12	44	9	9.3 (Lebedev) [206]
13	44	9	9.4 (Lebedev) [206]
14	50	11	U3 11-1 (McLaren), p.301 [205]; 11.1 (Lebedev) [206]

Table 10.1. Number of points per angular shell and degree of the highest spherical harmonic exactly integrated by grids specified by various entries on the angular grid line (Continued)

Entry	Points	Degree	Reference for Grid
15	54	11	11.2 (Lebedev) [206]
16	56	11	U3 11-2 (Stroud), p.301 [205]
17	60	11	11.3 (Lebedev) [206]
18	60	11	11.3 (Lebedev) [206]
19	78	13	13.2 (Lebedev) [206]
20	78	13	13.3 (Lebedev) [206]
21	86	15	15.1 (Lebedev) [206]
22	90	15	15.2 (Lebedev) [206]
23	90	15	15.2 (Lebedev) [206]
24	110	17	17.1 (Lebedev) [206]
25	116	17	17.2 (Lebedev) [206]
26	146	19	19 (Lebedev) [207]
27	146	19	19 (Lebedev) [207]
28	194	23	23 (Lebedev) [207]
29	302	29	29 (Lebedev) [208]
30	434	35	Lebedev [209]

10.5 The Cutoff File

The cutoff file specifies parameters to be used for the various iterations of an SCF calculation. The file to be used is determined by the `CUTOFFFILE` entry in the input file, as described in Section 9.1 on page 195. If the input file has no such entry, Jaguar uses the file `default.cutoff` from the data directory. If the `CUTOFFFILE` entry is `accurate.cutoff`, `solvent.cutoff`, or `quick.cutoff`, the program interprets the setting as `default.cutoff`.

The first line of a cutoff file contains a character string that includes the version number of Jaguar. This should be `cutv` followed by four digits giving the version number times 100. Leading zeroes are added if necessary. A comment on the same line can follow the version string.

The next five lines each have five numbers. Each line describes a particular level of accuracy to be used for the calculation. The first line provides the information necessary to run a calculation with all ultrafine pseudospectral grids and with "tight" cutoffs, and corresponds to an accuracy level setting of Ultrafine from the GUI, as described in Section 3.7.1 on page 54, or to the keyword setting **iacc** = 1 in the **gen** section of the input file, as described in Section 9.5.19 on page 245. The second line gives the parameters for the accurate level (**iacc** = 2), while the third line provides information for the quick level (**iacc** = 3). The last two lines are filled with zeroes, since they are required, but are not yet used.

In each of these rows, the columns describe which cutoff sets are used for various SCF iterations. The cutoff sets themselves are provided later in the file, and dictate the level of analytic corrections, the grid, and the non-default values of the **gen** section cutoff keywords (**cut1**, for example). The cutoff sets are described in more detail below. The columns reflect a scheme in which calculations are broken down into preliminary and final sets of iterations. The iterations from the beginning of the first SCF calculation in a run are considered to be part of the preliminary set, while the iterations from the end of the first SCF calculation, or from any subsequent set of SCF iterations, are considered to be part of the final set. For instance, for a solvation calculation, the SCF iterations for the analysis of the converged gas phase wave function are preliminary iterations followed by final iterations, while the SCF iterations for all subsequent SCF calculations (those including the solvent effects) are final iterations. Jaguar determines how many iterations are preliminary and how many are final for the initial SCF calculation.

The number in the first column in each of the five accuracy level lines dictates the cutoff set used for the first iteration in the preliminary sequence: if the number is a 1, the first cutoff set listed in the file is used; if it is a 5, the fifth is used, and so on. The number in the second column provides the cutoff set used for updates during the preliminary sequence of iterations. The third and fourth columns describe the cutoff sets used for the first and updating iterations in the final sequence, respectively. Finally, the last column dictates the cutoff sets used for non-SCF calculations, as for gradient calculations.

The first six lines of the `default.cutoff` file, which illustrate these points, are:

```
cutv0300
1 1 1 1 7   max. accuracy (prelim,prelim update,final,final update,gradient)
3 5 1 4 7   accurate
5 6 2 6 8   quick/solvent
0 0 0 0 0
0 0 0 0 0
```

The rest of the `.cutoff` file consists of the cutoff sets. Each set is specified by one line with four integers, sometimes followed by lines containing explicit cutoff keyword values, and ending with a blank line. The four integers represent the variables *jcor* and *kcor* (described below), the grid number, and the number of cutoff values to follow immediately below. The grid number should be 1 for the coarse grid, 2 for the medium grid, 3 for the fine grid, and 4 for the ultrafine grid, 5 for the charge grid, 6 for the gradient grid, 7 for the electron density cubic grid, 8 for the DFT medium grid, or 10 for the DFT gradient grid, where these grids are specified by the keywords **gcoarse**, **gmedium**, **gfine**, **gufine**, **gcharge**, **ggrad**, **geldens**, **gdftmed**, and **gdftgrad**. Section 9.5.27 on page 263 contains more information on these keywords.

The next lines specify each cutoff by number (e.g., 22 for the variable **cut22**) and value. Thus, the cutoff set:

```
5 2 4 3       set 3
  21 1.0e-3
  22 3.0
  24 1.0e-2
```

means that *jcor* is 5, *kcor* is 2, the ultrafine grid is used, and that three cutoff values which differ from the defaults follow. The next three lines set the cutoff values **cut21**, **cut22**, and **cut24**. If you need more information on cutoffs, contact Schrödinger.

The variables *jcor* and *kcor* determine what analytic corrections are calculated for a particular SCF iteration. The meanings of their possible values are shown in Table 10.2. The variables a, b, and c in the table refer to distinct atoms.

To perform an all-analytic calculation, you can set the keyword **nops** in the **gen** section of the input file to 1. All-analytic calculations use the cutoff keyword values in the `.cutoff` file, but ignore the *jcor*, *kcor*, and pseudospectral grid information.

Table 10.2. Determination of calculations of analytic corrections for SCF iterations

Variable	Value	Description[a]
jcor	0	No Coulomb terms calculated analytically
	1	Atomic analytic corrections of the form <aa|aa> calculated for ***J***
	3	Analytic corrections of the form <aa|aa> and <aa|bb> calculated for ***J***
	4	Analytic corrections of the form <aa|aa>, <aa|ab>, <aa|bb>, and <aa|bc> calculated for ***J***
	5	Analytic corrections of the form <aa|aa>, <aa|ab>, <aa|bb>, <ab|ab>, and <aa|bc> calculated for ***J*** (diatomic + <aa|bc>)
kcor	0	No exchange terms calculated analytically
	1	Atomic analytic corrections of the form <aa|aa> calculated for ***K***
	2	Diatomic analytic corrections of the form <aa|aa>, <aa|ab>, <aa|bb>, and <ab|ab> calculated for ***K***

a. a, b, and c refer to distinct atoms.

10.6 The PBF Radii File

The PBF radii file, `pbf_radii.ark`, determines how van der Waals radii for calculations using the Jaguar PBF solvation module are set according to chemical functional groups. Functional groups are defined in terms of SMARTS patterns (http://www.daylight.com/dayhtml/doc/theory/theory.smarts.html). The file sets radii and solvent accessibility parameters for chosen elements in each SMARTS pattern, using Ark syntax. It is included in the standard data directory, and supersedes the old `default.lewis` file.

By default (**pbf_radii**=1), the program obtains a Lewis structure from the Maestro file for the molecule or system, scans the PBF radii file for radius information for each atom and sets radii for relevant atoms, then sets any radii not determined by the PBF radii file according to the **atomic** section or to the standard default value. Settings for radii not included in the PBF radii file are described in Section 9.8 on page 273, and are listed in Table 9.46 on page 278. If you do not want the atomic radii that determine the dielectric continuum boundary to change according to the chemical environment of the atom (the first shell correction factor), set the solvation keyword **isurf** to 0 in the **gen** section. Otherwise, Jaguar alters radii by using the `pbf_radii.ark` file from the data directory, unless you specify your own file in a `PBFRADIIFILE` line in the input file, as described in Section 9.1 on page 195. If you want Jaguar to determine the Lewis structure from the input coordinates, set **pbf_radii**=2.

The radius settings in the file `pbf_radii.ark` (and `default.lewis`) were optimized for HF and LMP2 solvation calculations in water with Jaguar's solvation module that included the default correction terms for the cavity and surface area. The molecules used for radius optimization were the molecules containing carbon, hydrogen, oxygen, nitrogen, and sulfur from Reference 210. All calculations used a 6-31G** basis set. Geometries were obtained from gas phase optimizations at the HF and LMP2 levels. For both the geometry optimizations and the solvation energy calculations, the LMP2 treatment was restricted to heteroatom pairs.

10.6.1 Summary of Ark Syntax

The PBF radii file is formatted according to the Ark syntax, which is described in brief in Section B.1 of the *Desmond User Manual*. The format is based on a *keyword*=*value* notation. Keywords can be composed of alphanumeric characters and underscores, and must not begin with a numerical character. Values can be arbitrary strings (including integer, fixed-point, and floating-point numbers), tables, and lists. A table is an ordered set of *keyword*=*value* pairs enclosed in braces, in the following form:

{*key1* = *value1* *key2* = *value2* . . . *keyN* = *valueN*}

Lists are sequences of values of the form

[*value1* *value2* *value3* . . . *valueN*]

They can be used to assign values for an array, for example. Note that pairs of keywords and values in tables and values in lists are separated by spaces or new line characters, not commas. As values can be tables or lists, nested tables and lists are allowed. Empty strings are ignored. The # sign is used to start comments, which are also ignored.

10.6.2 The PBF Radii File Structure

The PBF radii file has the following syntax:

```
calculation_type = [
'smarts_pattern_1' {element1 = rad1 element2 = rad2 ... element1_dgsa =
dgsa_value1 ...}
['smarts_pattern_2a' 'smarts_pattern_2b'] {element1 = rad1 element2 = rad2
... element1_dgsa = dgsa_value1 ...}
...
]
```

The keyword *calculation_type* can take the values `hf`, `lmp2`, or `dft`. A single file can contain several calculation type specifications. The value of calculation type is a list of an even number of values. The odd-numbered values in this list specify SMARTS patterns, and the even-numbered values are tables which are used for specification of atom parameters. These atom specification tables consist of two types of pairs: *element* = *rad* and *element_dgsa* = *dgsa_value*. The former setting is used for assigning van der Waals radii on the atoms matched by the corresponding SMARTS pattern; *element* is an element symbol and *rad* is a number defining the van der Waals radius in angstroms. Setting *element* = *rad* assigns the van der Waals radius of value *rad* angstrom on all the atoms of this element matched by the corresponding pattern. The other type of setting, *element*`_dgsa` = *dgsa_value*, works similarly to the van der Waals radius specification, except the keyword *element*`_dgsa` is the dGsa parameter (ΔG of solvent accessibility) for the particular element and *dgsa_value* is its value in kcal/mol. For example, the table of van der Waals radii settings and dGsa settings may look like

```
{O = 1.30 N = 1.45 H = 1.15 N_dgsa = -1.40}
```

where the matched oxygen, nitrogen and hydrogen atoms will be assigned the van der Waals radii of 1.30, 1.45, and 1.15 angstroms, respectively. In addition, the matched nitrogen atom will be assigned the dGsa value of −1.40 kcal/mol. The unmatched atoms or atoms for which no custom values are specified in the table are assigned the default values, or, if matched by some other pattern in the SMARTS file, the values specified in the table corresponding to that pattern. If a certain atom is matched by two or more patterns, the values of the later pattern overwrite those of the earlier pattern. Thus, the more general patterns should be included earlier in the file, and the more specific patterns later in the file.

There are two ways to specify SMARTS patterns. The most typical way is a simple single-quoted SMARTS pattern, such as `'[CX3]=[OX1]'` for the carbonyl group. The second way of matching atoms is by two SMARTS patterns, `['`*smarts_pattern_1*`' '`*smarts_pattern_2*`']`, in which the second pattern targets the atoms matched by the first. This syntax is useful for very specific matching. Consider, for example, matching the sp3-hybridized oxygen in the carboxylic group. This can be accomplished by the following list of two patterns:

```
['[CX3](=O)[OX2H1][H]' '[OX2]']
```

The first pattern, `[CX3](=O)[OX2H1][H]`, matches the COOH group, and the second, `[OX2]`, matches the doubly-bonded oxygen within the COOH group. Note that for this example it is not sufficient to write the first pattern as `[CX3](=O)[OX2H1]` because the matched group will be COO, not COOH, so that the second pattern, `[OX2]`, will be unable to find the double-bonded oxygen in COO.

The examples below summarize distinctions between one and two SMARTS patterns, as used for assigning radii in the SMARTS file:

```
# Assign van der Waals radius of 1.30 A to sp3 oxygen in the COOH group
['[CX3](=O)[OX2H1][H]' '[OX2]'] {O = 1.30}

# Assign van der Waals radius of 1.30 A to sp2 oxygen in the COOH group
['[CX3](=O)[OX2H1][H]' '[OX1]'] {O = 1.30}

# Assign van der Waals radius of 1.30 A to both oxygens in the COOH
group
'[CX3](=O)[OX2H1]' {O = 1.30}
```

Running Jobs

Running, monitoring and controlling jobs is done by the Schrödinger job control facility. This facility has both a graphical user interface in the Maestro Monitor panel and a command-line interface in the `jobcontrol` command. The job control facility handles scratch directory creation and cleanup, and ensures that each job has a unique scratch directory. Output files are copied to the working directory while the job is running. Detailed information on Job Control can be found in the *Job Control Guide*. Some of this information is repeated here. Basic information on running jobs is given in Section 1.2 on page 2 and Section 1.3 on page 4.

If you intend to run jobs on various hosts, you must provide information on the hosts to the job control facility through a file named `schrodinger.hosts`. How to provide this information is described in Section 6.1 of the *Installation Guide*.

In addition to using the job control facility, you can use the `jaguar` command to perform a number of job submission tasks. The `jaguar` command is described in the following section, and creating batch scripts to submit multiple Jaguar jobs is described in the subsequent section.

11.1 The jaguar Command

You can use the `jaguar` command to perform the following tasks, among others:

- Run a job on any machine at your site
- List the machines on which Jaguar is installed
- Run special jobs such as Babel file conversion or NBO calculations.

If Jaguar is installed on more than one machine at your site, you can use the `jaguar` command on one machine to run Jaguar jobs on another machine, even if you are not logged in to the second machine. This section describes in some detail how and when to use the `jaguar` command.

The syntax of the `jaguar` command is

`jaguar [`*command*`] [`*options*`]`

where *command* is any of the commands listed in Table 11.1. If the `jaguar` command is not in your path, you must precede it with `$SCHRODINGER`, i.e. `$SCHRODINGER/jaguar` (This is not necessary on Windows). The options may be given in any order, and may precede any options specific to the command. See Section 1.2 on page 2 for general information on running from the command line.

Table 11.1. Commands for the `jaguar` *command*

Command	Description
`run` [*script*`.py`|*script*`.bat`] [*options*] *input-files*	Start the Jaguar jobs whose input files are listed, using the specified script if given, and the specified run time options.
`pka` [*options*] *jobname*	Start a Jaguar pK_a calculation.
`babel` [*babel-options*]	Perform a file format conversion using Babel.
`nbo`	Run an NBO calculation.
`results` *options*	Summarize results from the output file using the options specified. See Section 6.1 on page 125 for more information on this command.
`help`	Display a command syntax summary including a list of valid commands.

The *input-files* argument to the `jaguar` command is a list of names of Jaguar input files. The name must include the `.in` extension. Jaguar removes the `.in` extension to form the job name. For example, the command

```
jaguar run h2o.in
```

runs a Jaguar job with the job name `h2o` and the input file `h2o.in`.

The `jaguar jobs` command is an interface to the `jobcontrol` command with Jaguar selected as the program, which executes the following command:

```
jobcontrol -list program=jaguar
```

In addition to running the commands listed in Table 11.1, you can use the `jaguar` command with the standard Job Control options listed in Table 2.3 of the *Job Control Guide* to obtain information about hosts available. For example, to determine which machines are available for running Jaguar jobs, enter the command

```
jaguar -HOSTS
```

The hosts listed are those in the `schrodinger.hosts` file that are being used by the `jaguar` command. If you find that the list of hosts is incomplete, you may need to edit the `schrodinger.hosts` file indicated on the first line of the command output. See Section 6.1 of the *Installation Guide* for a description of the `schrodinger.hosts` file.

11.1.1 Running a Jaguar Job From the Command Line

The `jaguar run` command lets you run a Jaguar job using the Jaguar input file you specify and any of the `jaguar run` command options shown in Table 11.2 and described below. You can also use the standard Job Control options—see Section 2.3 of the *Job Control Guide* for more information. Note that running MPI or OpenMP parallel jobs is not supported on Windows.

Table 11.2. Options for the `jaguar run` *command*

Option	Effect	Default Behavior
`-WAIT`	Wait for the Jaguar job to finish before returning control to the shell.	Return control to the shell immediately.
`-keepscr`	Keep temporary files and temp directory for job at end of job.	Temporary files are cleaned out of temp directory and temporary directory is removed at end of job
`-OMPI` *nprocs*	Use *nprocs* MPI processes for a parallel job.	Run a serial job
`-TPP` *nthreads*	Use *nthreads* OpenMP threads per process. These threads can only run on the same host. This option can be combined with `-OMPI` to mean that each MPI process should use *nthreads* OpenMP threads on the same host.	Do not use threads.
`-PARALLEL` *npr*	Use *npr* processors, and distribute them between subjobs, OpenMPI processes and OpenMP threads to optimize resource use.	Run a serial job.
`-FORCE`	Force the scratch directory to be overwritten if it exists.	Abort the job if a scratch directory named for the job already exists
`-t`	Write time stamps to the log file after each executable has run	Write time stamps to the log file at the start and the end of a job

To run a Jaguar job, you first need a Jaguar input file. The file should be named in the form *jobname*`.in`.You can create an input file using the GUI (see Section 2.3 on page 11 for more information). If you create or edit an input file using a text editor, make sure its format agrees with that described in Chapter 9.

You can run a single Jaguar job from the command line with the command

```
jaguar run jobname.in
```

where *jobname* is the stem of your input file name, *jobname*`.in`. With this command, the job runs on the machine upon which you have submitted the command.

To run a Jaguar job on another machine, use a command in this form:

```
jaguar run -HOST hostname jobname.in
```

where your input file is named *jobname*`.in` and *hostname* is one of the hosts in the file `schrodinger.hosts`. For instance, if you were logged into a machine named `alpha` and wanted to run a job with the input file `ch4.in` on a machine named `beta`, you would enter

```
jaguar run -HOST beta ch4.in
```

The `jaguar run` command has several other command line options. For example,

```
jaguar run -NICE -SAVE jobname.in
```

causes executables to be run with a lower CPU scheduling priority (see the man page on `nice`) and leaves all temporary files generated during the job in the temporary directory.

To submit a series of independent jobs, you can replace *jobname*`.in` with a list of input file names. If you do not specify a host, or specify a single host, the jobs run sequentially. If you specify multiple hosts with the `-HOST` option, the jobs are distributed over the hosts specified. When a host finishes running a job, it starts the next job, until there are no more jobs to be run. The list of hosts must be separated by spaces and enclosed in quotes. For hosts that have more than one processor, you can append the number of jobs to run on that host to the host name, separated by a colon, as in the following example.

```
jaguar run -HOST beta:2 ch4.in nh3.in
```

11.1.2 Converting File Formats

Jaguar uses the Babel program [26] to convert between many of the file formats used in computational chemistry. Babel can read over 40 kinds of input and output file types, and writes both cartesian and Z-matrix geometry specifications. Babel is used in the GUI to read and write files that are not in Jaguar or Maestro format. You can also request Jaguar to write out files during a job run using the **babel** or **babelg** keywords (see Section 9.5.22 on page 254 for more information).

To convert file formats from the command line, you can use the `jaguar babel` command. The syntax of the command is:

```
jaguar babel [-v] -iinput-format input-file [-h|-d] [range]
   -ooutput-format [output-file] [-split]
```

The `-i` and `-o` arguments are required to set the input and output formats, respectively. The output format keywords are listed in Table 9.32 on page 255; the input format keywords are listed in Table 11.3.

Note that the format keywords are not used for file extensions, as they are when you use the **babel** and **babelg** keywords in a Jaguar input file. The input and output file names given in the `jaguar babel` command are used as they are. If you omit the output file name, or if you give `CON` as the output file name, the output is written to standard output.

You can add hydrogen atoms to a structure when you do a conversion using the `-h` option, and you can delete hydrogen atoms from a structure, using the `-d` option.

Babel can convert multi-structure files to other multi-structure files or to a set of single structure files. You must supply both an input file name and an output file name if you are converting a multi-structure file.

You can select the structures to convert by specifying the *range* input argument. A valid ranges is in the form "*number1-number2*", or the word `all` to select all structures. The quotes are required. For Jaguar output files, the last structure is converted if no range is given; otherwise, the first structure is converted by default.

To generate a set of single structure files, use the `-split` keyword. The names of these files have a four-digit index number inserted before the file extension. For example, to write individual Jaguar input files (Cartesian) for the 5th through 10th intermediate structures in a Jaguar geometry optimization run, type the command

```
jaguar babel -ijagout job.out "5-10" -ojagc iter.in -split
```

The files `iter0001.in`, `iter0002.in`, ... `iter0006.in` are written by Babel.

Babel cannot read Maestro-formatted files. For conversions between Schrödinger file formats that are not recognized by Babel, there is a file conversion utility, `jagconvert`. This utility reads and writes Jaguar input (`.in`) and output (`.out`) files and Maestro (`.mae`) files. The utility is located in `$SCHRODINGER/utilities`. The command syntax is as follows:

`jagconvert [`*intype*`]` *infile outtype outfile*

where *intype* is one of `-ijag`, `-ijin`, `-ijout`, `-imae`, or `-imultimae`, and *outtype* is one of `-ojin`, `-omae`, or `-omultimae`. The input file is assumed to be a Jaguar input file if no input type is explicitly given. MacroModel files are read in using `-imae`. If you convert a file that contains multiple structures, only the first structure in the file is converted to the new format.

Table 11.3. Input format keywords and file types for babel file format conversions

Format Keyword	File Type
alc	Alchemy file
prep	AMBER PREP file
bs	Ball and Stick file
bgf	MSI BGF file
car	Biosym .CAR file
boog	Boogie file
caccrt	Cacao Cartesian file
cadpac	Cambridge CADPAC file
charmm	CHARMm file
c3d1	Chem3D Cartesian 1 file
c3d2	Chem3D Cartesian 2 file
cssr	CSD CSSR file
fdat	CSD FDAT file
gstat	CSD GSTAT file
dock	Dock Database file
dpdb	Dock PDB file
feat	Feature file
fract	Free Form Fractional file
gamout	GAMESS Output file
gzmat	Gaussian Z-Matrix file
gauout	Gaussian 92 Output file
g94	Gaussian 94 Output file
gr96A	GROMOS96 (A) file
gr96N	GROMOS96 (nm) file
hin	Hyperchem HIN file
sdf	MDL Isis SDF file
jagin	Jaguar Input file
jagout	Jaguar Output file
m3d	M3D file

Table 11.3. Input format keywords and file types for babel file format conversions (Continued)

Format Keyword	File Type
macmol	Mac Molecule file
macmod	Macromodel file
micro	Micro World file
mm2in	MM2 Input file
mm2out	MM2 Output file
mm3	MM3 file
mmads	MMADS file
mdl	MDL MOLfile file
molen	MOLIN file
mopcrt	Mopac Cartesian file
mopint	Mopac Internal file
mopout	Mopac Output file
pcmod	PC Model file
pdb	PDB file
psin	PS-GVB Input file
psout	PS-GVB Output file
msf	Quanta MSF file
schakal	Schakal file
shelx	ShelX file
smiles	SMILES file
spar	Spartan file
semi	Spartan Semi-Empirical file
spmm	Spartan Molecular Mechanics file
mol	Sybyl Mol file
mol2	Sybyl Mol2 file
wiz	Conjure file
unixyz	UniChem XYZ file
xyz	XYZ file
xed	XED file

11.2 Running Multiple Jobs with Batch Scripts

If you need to run series of Jaguar jobs frequently, you can create batch scripts that define the jobs and run them using the `jaguar run` command. For instance, you might want to study the dissociation of a bond by evaluating the molecule's energy at various appropriate bond lengths; scan a potential energy surface; or perform a Hartree-Fock-level geometry optimization and then evaluate the energy of the new structure using LMP2 or DFT techniques.

To run such a job you need a batch input file (whose name should end in `.bat`) and at least one input file. The input files can provide structures (in either Maestro or Jaguar format) or can provide templates for running the calculations. The batch input file has instructions for creating a Jaguar input file or modifying a template input file for each Jaguar job. These modifications can include changes to particular bond lengths and angles of the structure, changes in the wave function or job type (such as changing an HF geometry optimization input file to a DFT single-point energy calculation input file), changes in the files or directories used for jobs, and virtually all other settings made in input files. One batch input file can be used to request several different input files, either from one template input file or from several different templates. The `jaguar run` command then generates the input files and runs the corresponding jobs, either consecutively if only one host is specified, or by distributing the jobs over the specified hosts.

11.2.1 Batch Input File Format

Batch input files can include directives, job specifications, UNIX commands, and comments. Lines that contain comments must begin with a `#` symbol, and lines that contain Unix commands must begin with a `%` symbol. Blank lines can also be used in the batch script, and are ignored.

The available directives are summarized in Table 11.4. The directives apply to *all* jobs described below them, unless a later line of the same type replaces them. Any `OUTDIR`, `TEMP`, `EXEC`, or `FLAGS` directive replaces any earlier setting made by the same directive, and any of these settings can be reset to their default values with the value `NONE` (for instance, `FLAGS=NONE`). An `OPTIONS =` directive clears all previously set options and creates a new options list. An `OPTIONS +` directive adds new options to the options list or redefines options already in the options list.The syntax for the options set by `OPTIONS` directives is described later and summarized in Table 11.5.

The syntax for job specifications is as follows:

template-name [*new-name* [*options*]]

Table 11.4. Batch input file directives

Directive	Action
`OUTDIR = `*directory*	Set the directory to which output from jobs will be written. The default is the job submission directory. This directory is created automatically if it does not exist.
`SCRATCH = `*directory*	Set the scratch directory. Equivalent to specifying the `JAGUAR_SCRATCH` environment variable.
`TMPDIR = `*directory*	Set the scratch directory root. Equivalent of `tmpdir` setting in `schrodinger.hosts`.
`WORKDIR `*directory* [*files*]	Create the specified directory and use it as the working directory for input and output. Copy the specified files into the directory. The default is the job submission directory.
`FLAGS = `*options*	Specify `jaguar run` command line options
`OPTIONS {=|+} `*options*	Set options to apply to subsequent jobs. Options can be specified over multiple lines by using = on the first line and + on subsequent lines. Options are listed in Table 11.5.
`OUTFILES {=|+} `*files*	Copy the specified files from `WORKDIR` to `OUTDIR` at the end of the job. The file list can be spread over multiple lines by using = on the first line and + on subsequent lines. If set to `_ALL_`, then output files from all subjobs are copied; if set to `_LAST_`, only the output files from the last subjob are copied. The default is `_ALL_`. The files that are copied become the output files of the batch job.
`STRUCTOUT = `*file*	Specify the structure output (Maestro) file of the batch job. This is a file in `WORKDIR` that is copied to `OUTDIR` at the end of the job, and that will be monitored and incorporated by Maestro.
`IGNORE_ERRORS`	Continue to the next job if a job step fails. The default is to stop execution of the batch script and exit.
`PURGE_JOBDB`	Purge the job record for each job after it finishes.
`EXIT`	Exit from the batch script.

Each job specification defines a single Jaguar job. For each job, the following steps are taken:

1. The template file *template-name*`.in` is read.

   This file is read from the current working directory.

2. Any options that are defined are applied to the contents of the template file.

   Options that are given on the job specification line override options that are specified with an `OPTIONS` directive. Option syntax is given below.

3. A new input file, *new-name*`.in` is created.

   The new file is written to the directory specified by a `WORKDIR` directive or, if no `WORKDIR` directive has been given, to the current working directory. If *new-name* is not specified, *new-name* is set to *template-name*. If the file *new-name*`.in` already exists, it is overwritten, unless you use the `-r` option described later in this section.

4. The Jaguar job is run using `jaguar run` with this new file as input.

   The command line options for the Jaguar job are specified by the `FLAGS` directive. Temporary files generated during the job are written to the subdirectory *new-name* in the scratch directory, and output files are written to the directory listed on the `OUTDIR` line, if given, or from the current working directory.

The template job name can either be the stem of an existing input file or the string `$JOB`. If the string `$JOB` is used, the batch script is run multiple times, substituting for `$JOB` the job names that are provided as arguments to the `jaguar run` command. For example, for the job specification

```
h2o    h2o_dft   dftname=b3lyp
```

the file `h2o.in` is read, the keyword setting `dftname=b3lyp` is added to the **gen** section of the input, and the new input is written to the file `h2o_dft.in`. The same effect is achieved with the job specification

```
$JOB   $JOB_dft   dftname=b3lyp
```

and running `jaguar run` with the file name `h2o.in` as an argument.

If no options are specified, the Jaguar job is run using the template file as input. For example, if you had a set of input files *jobname1*`.in`, *jobname2*`.in`, *jobname3*`.in`, you could use the following batch input file to run Jaguar for each input file in order:

jobname1
jobname2
jobname3

Options for each Jaguar job can be set in preceding `OPTIONS` directives or by an options list appearing in the job specification. An options list appearing in the job specification applies only to that job. Options specified in an `OPTIONS` directive apply to all subsequent jobs, unless superseded by a later `OPTIONS =` directive or by the options list for the job.

These job options can specify any of the following items for the relevant jobs:

- Keyword settings in the **gen** section of the Jaguar input file
- Paths and names of data files, such as the basis set file or the grid file

- Sections to remove from the template input file: for example, the **guess** section if you are changing basis sets
- Substitution of a specified number or string for one already in the template input file

The format for each of these options and an example of each kind are shown in Table 11.5. Option assignments must not have spaces around the `=` or `==` operators. Host names cannot be included in any of the paths described in the table. You should avoid using any of the characters `"$!\<>?` in a substitution pattern.

These options and the other line types in a batch input file are illustrated in the sample files in the next subsection. Directions on how to submit a batch job follow in the final subsection.

Table 11.5. Definition of options that are applied to a template file to generate an input file

Change	**Format**	**Examples**

set keywords	*keyword*=*new-value,* or *keyword*=NONE to remove a setting	`basis=lav3p**`
		`dftname=b3lyp`
		`igeopt=NONE`

specify a data file path and name	*filetype*=*name*, or *filetype*=NONE to return to default choice for that file type	`BASISFILE=my.bas`
		`ATOMIGFILE=NONE`
		`DAFFILE=NONE`
		`GRIDFILE=NONE`
		`CUTOFFFILE=NONE`
		`GPTSFILE=NONE`
		`WAVEFNFILE=NONE`

remove a section	`RMSECTION=`*section-name*	`RMSECTION=guess`
clear the **gen** section except for the **multip** and **molchg** settings	`RESETGEN`	
insert a file[a] at the top of the input	`ADDTOP=`*filename*	`ADDTOP=guess.txt`
append a file[a] to the input	`ADDEND=`*filename*	`ADDEND=guess.txt`

substitute a value for a variable	*old-pattern*==*new-pattern*. Do not use any of the characters `"*$!\<>?` in either pattern	`bond==1.5`
		`torang==170.0`

a. Filename must not end in `.in`, as this is taken as an input file.

11.2.2 Running Batch Jobs

You can start Jaguar batch jobs from Maestro or from the command line. Maestro automatically creates a simple batch job when you specify multiple structures as input to any Jaguar task. See Section 2.10 on page 26 for more information on using Maestro to run batch jobs.

The syntax of the `jaguar` command to use is:

`jaguar run [`*command-options*`]` *batchfile*`.bat [`*input-files*`]`

If the batch script *batchfile*`.bat` uses `$JOB` in job specifications, you must supply the list of files to substitute in *input-files*. The suffix `.bat` is required. The current directory is searched first for the batch file. If a batch file is not found, the batch scripts directory in the installation, `$SCHRODINGER/jaguar-v`*version*`/scripts`, is searched. The scripts in this directory are described in Table 2.1 on page 28.

The Job Control options `-HOST`, `-USER`, and `-WAIT` can be used in the `jaguar run` command. For distributed batch jobs you can specify a list of hosts with the `-HOST` option. The host names in the list must be separated by spaces, and if there is more than one host, the list must be enclosed in quotes. If a host has more than one processor, you can run multiple subjobs (individual calculations) on that host either by repeating the host name or by appending a colon and the number of subjobs to the host name, e.g. `cluster:32`.

There are also some special command options for use with batch scripts, which are summarized in Table 11.6.

The `-r` option is a restart option, which prevents `jaguar run` from overwriting existing Jaguar input and output files and from running the job steps that created them.

Table 11.6. The `jaguar run` *command line options for batch scripts*

Option	Description
`-c`	Create input files, but don't run the batch job.
`-nolocal`	Do not run in the local directory.
`-r`	Restart option. Skip execution of steps that are completed, i.e., steps that have input files and completed output files. The default action is to generate Jaguar input files from template files even if they overwrite previously existing files, and run the corresponding job step.

You can use both Jaguar input files (*jobname*`.in`) and Maestro files (*jobname*`.mae`) as input, and you can specify both on the same command line. Maestro files are used as a source of structures only, and a Jaguar input file is created for each structure in the Maestro file, using the commands in the batch script. In this case, there is no template to be modified. You can create the batch script and the Maestro input file from the Jaguar panels in Maestro and submit the batch job for execution, or save the files to disk and run them later from the command line or from Maestro.

If you run remote batch jobs, by default the input and output directories must be on a disk system that is available to both the submission and the execution host, such as a cross-mounted disk or an NFS file system. This behavior corresponds to the use of the `-LOCAL` option. If you want to remove this restriction, run the batch job with the `-nolocal` option. The individual jobs in the batch script are run in their own local directories, which may be on the remote host.

The `jaguar run` command also accepts Python scripts as batch scripts (see Section 11.3 on page 329). Apart from the Python scripts in the distribution, you can only run `jaguar` with a Python script from the command line.

11.2.3 Batch Input File Examples

Batch scripts can be used in multiple ways. In this section, two examples are provided for the following scenarios:

1. Pipelined scripts, in which the output from one job provides the input to a subsequent job. The pKa batch script is an example. Here we provide an example that performs an optimization at a medium level of theory followed by single-point calculations at a higher level of theory.
2. A script that uses the input files in one directory as templates and writes all new files into the launch directory. This kind of script is useful for running jobs with different options on the same structure files.

11.2.3.1 Pipelined Jobs

Suppose you have ten different molecules and you want to optimize the geometry of each one at the B3LYP/6-31G* level of theory, and then do two single-point energy calculations on the optimized geometry, one using B3LYP/6-311+G* and the other using LMP2/6-311+G*. The batch file for this process, given below, would read in each molecular geometry from an existing input file, make the necessary keyword changes, and perform the calculations:

```
# B3LYP/6-31G* geometry optimization
$JOB  $JOB_dft_opt  igeopt=1  basis=6-31g* dftname=b3lyp

# remove igeopt setting for the following single-
# point calculations and change basis set to 6-311+G*
OPTIONS = igeopt=NONE  basis=6-311+g*

# run B3LYP single-point calculation
$JOB_dft_opt.01  $JOB_dft_sp
```

```
# change level of theory to LMP2 and run single-point calculation
$JOB_dft_opt.01  $JOB_lmp2_sp dftname=NONE  mp2=3
```

11.2.3.2 Running Jobs from Input in a Specified Directory

The script below runs jobs for a set of input files in a specified directory using two different sets of options, and places the output in subdirectories of the launch directory.

```
# Define protocol #1

OPTIONS= basis=midi dftname=x3lyp icfit=1

# Create the job directory and cd into it.  All job files will be
# written to this directory if WORKDIR and OUTDIR are set to the same
# directory name.

WORKDIR= protocol1
OUTDIR= protocol1

# Get the input files, append "-proto1" to each, add the OPTIONS
# keywords, and run them through Jaguar.

$JOB  $JOB-proto1

# Now go back to the original launch directory, create the directory
# for protocol2, and repeat the calculations using protocol #2

OPTIONS= basis=cc-pvtz dftname=x3lyp icfit=1

WORKDIR= protocol2
OUTDIR= protocol2

$JOB  $JOB-proto2
```

To run this script for the files in the subdirectory `structures`, you would use the command:

```
jaguar run template.bat structures/*.in
```

The input files are copied from the `structures` to the directory specified by `WORKDIR`, modified with keywords and renamed as specified in the batch script, and then submitted to Jaguar. The output file, the log file and the restart file are written to the directory specified by `OUTDIR`.

11.3 Running Complex Jobs with Python Scripts

The Python package provided with Schrödinger software provides Python language extensions to analyze and manipulate structures, including reading and writing a variety of file formats; interfaces to Maestro, including support for creating customized user interfaces; and functions to interface with the Schrödinger Job Control facility. For more information, see the *Python Module Overview*, which you can open from the Help menu in Maestro.

The Jaguar software distribution includes several Python scripts, which are described in the sections below. Some of these scripts also have Maestro interfaces for setting up the job.

11.3.1 counterpoise.py

The script `counterpoise.py` can be used to calculate a counterpoise-corrected binding energy of a complex consisting of two non-covalently bound molecules. You can run this script from Maestro, as described in Section 2.4.7 on page 18, or you can run this script from the command line using the following syntax:

```
jaguar run counterpoise.py filename
```

where *filename* can be either a Jaguar input file or a Maestro structure file. A job directory called *filename*`_counterpoise` is created in the current working directory, and the job files for all calculations are written to this directory. When the job finishes, a file named *filename*`_counterpoise.out` is written to the current working directory. This file contains the counterpoise-corrected binding energy and the counterpoise correction energy.

If the basis set and level of theory are not specified in the input file, then each job runs using the same default settings as for Jaguar jobs launched from Maestro: the default level of theory is DFT with the B3LYP functional, the default basis set is 6-31G**, and no geometry optimization is performed. You can specify keyword settings in the input file if it is a Jaguar input file, but you can also specify settings on the command line. For example, if you want to optimize the geometries of the fragments and of the complex using the X3LYP functional, then you would use the following command:

```
jaguar run counterpoise.py filename -keyword igeopt=1
        -keyword dftname=x3lyp
```

We have run binding energy calculations on intermolecular complexes which are held together by a variety of interactions, including hydrogen bonds, π-π interactions, π-cation interactions, halogen bonding, chalcogen bonding, dispersion, and strong electrostatic (ionic) interactions. The most accurate and computationally practical protocol that we have found is M06-2X/cc-pVTZ(-f)+, which usually has a binding energy error of less than 1 kcal/mol when compared to basis set extrapolated CCSD(T) values. One exception is the case of π-cation complexes in which the cation is a metal. Here, no DFT method that we have tried is able to capture the very strong polarizing effect of the metal cation on the π electrons, and the binding energy error is about 5 kcal/mol. For interaction energies of π-cation complexes we recommend using the B3LYP-MM method which contains semiempirical a posteriori corrections that capture effects of such interactions—see Section 3.3.5 on page 46 for more detail. However, when the cation is polyatomic, like ammonium, the error drops to about 1 kcal/mol. LMP2 is comparable, and in fact it is no better than M06-2X when treating the dispersively bound complexes in Hobza's S22 database [244], but it is much more expensive than M06-2X.

11.3.2 hydrogen_bond.py

The script `hydrogen_bond.py` can be used to calculate the total binding energy of a hydrogen-bonded complex of two molecules. See Section 5.1 on page 115 for details on the computational protocol. You can use this script from Maestro, as described in Section 5.1 on page 115, or from the command line. The command syntax is as follows:

`jaguar run hydrogen_bond.py [`*options*`]` *filenames*

where *filenames* are one or more Jaguar input files or Maestro structure files (only one file type, not a mixture of both). For a description of the options, run the command with the `-h` option.

For each complex, a job directory called *name*`_hydrogen_bond` is created in the current working directory, where *name* is the stem of the structure file name. The job files for that complex are written to this directory. A file called *name*`_hydrogen_bond.out` is written to the current working directory when the job finishes. The content of this file should be self-explanatory:

```
-----------------------------------------------------------------------
Hydrogen bond energy calculation for h2o-h2co.in
Energy units are kcal/mol

BSSE-corrected cc-pVTZ(-f) binding energy:  -3.98
  (cc-pVTZ(-f) BSSE correction:   1.08)
BSSE-corrected cc-pVQZ(-g) binding energy:  -4.37
  (cc-pVQZ(-g) BSSE correction:   0.41)

              Extrapolated binding energy:  -4.77
-----------------------------------------------------------------------
```

The binding energy calculated with the two basis sets is printed, along with the corresponding BSSE corrections. As the size of the basis set increases, the size of the BSSE is expected to decrease, and the binding energy is expected to increase in magnitude. The binding energy extrapolated to the basis set limit is printed last. For especially weakly-bound systems, it occasionally happens that the energy obtained with the cc-pVTZ(-f) basis set is slightly lower than that obtained with the larger cc-pVQZ(-g) basis set[1]. In this situation the usual two-point basis set extrapolation formula cannot be used, so the energy is instead corrected using a simple linear correction of the LMP2/cc-pVQZ(-g) energies to the CCSD(T) reference energies. The output file clearly indicates when this has happened. The binding energy will still be quite accurate to within 0.5 kcal/mol when compared to the CCSD(T) energy.

11.3.3 h_abstraction.py

The script `h_abstraction.py` estimates the energy of homolytic breaking of a bond to a hydrogen atom in the following two processes:

(a) R-H = R· + H·

(b) (R·)-H = R + H·

The second is only relevant for sp and sp^2 carbon atoms, such as those in a benzene ring.

The script does not calculate the energy of abstraction by a probe, for which the process is

R-H + Probe· = R· + Probe-H

but this can be evaluated simply by performing a calculation for the probe first, then subtracting the resulting bond energy from those of the target molecules. Often, only the relative bond energies are required, to determine which hydrogens can be most easily abstracted by a given probe. The script should be useful for studying reactivity with respect to radical oxidation, among other things.

The syntax is as follows:

`jaguar run h_abstraction.py -atom` *atoms options maestro-file*

The structure input is a Maestro file containing the structures for which hydrogen abstraction energies are needed. For a description of the options, run the command with the `-h` option.

The standard job control keywords are accepted, such as `-TPP`, `-OMPI`, `-HOST`. The `-atoms` option defines the list of hydrogen atoms to abstract and calculate bond energies for. The list of atoms is applied to all structures in the input file, so you must make sure that these atom numbers make sense—for example, if you build the structures on a common scaffold by adding functional groups, and the atom numbers all apply to the common scaffold. If the structures do not allow a common atom set to be used, then you should run them separately.

1. This may be due to the fact that the BSSE corrections are calculated only for the Hartree-Fock part of the wave function rather than for the full LMP2 wave function (which would be more expensive). The error in this approximation amounts to a few tenths of a kcal/mol.

The following example command passes two settings for the energy calculation:

```
jaguar run h_abstraction.py -atoms=1,4,12,19
    -spk maxit=155 -spk isymm=0 mystructure.mae
```

The text output is contained in the file *filestem*`_h_abstraction.out`, where *filestem* is the structure file name without the extension. An example of the output follows:

```
Hydrogen dissociation energy calculation

Input structure file: habs_test3.mae
Job options: -atoms=4,5,7

Hydrogen atom energy: -0.5 au

                     Dissociation energy
Structure    Atom        (kcal/mol)
---------  --------  -------------------
     1        H4           109.54
     1        H5            99.87
     1        H7           108.59

Done.
```

The default computational protocol is as follows. The geometry optimizations on the hydrogenated and dehydrogenated forms are performed with B3LYP/LACVP*. The single-point calculations on the hydrogenated and dehydrogenated forms use M06-2X/cc-pVTZ(-f). Both calculations use the pseudospectral method. All calculations are performed in the gas phase. The protocol can be modified by setting keywords.

The energy of the hydrogen radical is not computed explicitly but is taken to be -0.5 hartrees. The actual energy of the hydrogen radical is unimportant for relative comparisons of the strengths of the bonds, because it has the same value in every process, and thus shifts all the dissociation energies by a constant.

11.3.4 distributed_scan.py

This script splits up a 1D or 2D scan job into multiple jobs that can be distributed over multiple processors, and is used automatically by Maestro for such a job. The script accepts a single input file that defines the scan job. The processors are specified using the `-HOST` job control option (see page 10 of the *Job Control Guide*). The results are collated into a single output when all jobs have finished.

11.3.5 fukui.py

This script can be used to calculate Fukui functions. You can use this script from Maestro, as described in Section 5.2 on page 117, or from the command line. The command syntax is:

`jaguar run fukui.py [`*options*`]` *filename*

where *filename* is the Maestro structure file that contains the molecules for which you want to calculate Fukui functions. For a description of the options, run the command with the `-h` option.

The Fukui functions are returned as a set of five `.vis` files for each structure in the input Maestro file, named as follows:

jobname`_N+1_density.vis`
jobname`_N-1_density.vis`
jobname`_density.vis`
jobname`_f+.vis`
jobname`_f-.vis`

If the input file contains multiple structures, *jobname* is replaced with *jobname_entry-name*. The file *jobname*`_fukui.out` contains information on the input file and command options used. The output files from each of the Jaguar jobs run by the script are copied back to the subdirectory *jobname*`_fukui`.

To display the results in Maestro, import the output structure file, *jobname*`-fukui.mae` into Maestro. When you do, ensure that the Import associated data files option in the Import panel is selected, otherwise the `.vis` files are not imported and will have to be imported manually.

11.3.6 vcd.py

The `vcd.py` script automates the calculations needed to produce high quality VCD spectra. This includes conformational searching of the input structure (which requires a MacroModel license), quantum mechanical refinement of the conformers (using Jaguar), elimination of redundant conformers (again using MacroModel), calculation of the VCD spectra for the remaining conformers (using Jaguar), and computation of Boltzmann weighting factors for the conformers (using Jaguar). This information can then be used to generate a VCD spectrum using the Spectrum Plot panel (see Section 3.12).

The command syntax is as follows:

`jaguar run vcd.py [`*options*`]` *filename*

where *filename* is a Maestro structure file. If the file contains multiple structures, all of the structures are assumed to be conformers of the molecule of interest. If the file contains a single structure is used as input, then a conformational search is run on it by default, although this may be suppressed with the appropriate option. For a description of the options, run the command with the `-h` option.

11.3.7 canonical.py

The script `canonical.py` guarantees that calculations on two or more isomers I_1, I_2, ... will use the same number of canonical orbitals, which makes the energies of these isomers comparable. First, the script precomputes the number of canonical orbitals NCO_I_*k* for each provided isomer I_*k*, terminating after the one electron integrals step. Then, the number of the canonical orbitals NCO to be used for all the calculations is determined by taking the minimum of NCO_I_1, NCO_I_2, etc. Finally, the original calculations are rerun with the identical number of canonical orbitals, NCO. The input can contain multiple sets of isomers as well as unique structure, and the appropriate number of canonical orbitals is determined for each set.

The command syntax is:

```
jaguar run canonical.py [options] filename
```

where *filename* is a Maestro structure file containing two or more isomeric structures. The options, accepted in the form `-k` *keyword=value* define the details of the Jaguar calculations. For example, the options `-k basis=6-31G -k dftname=m06-2x -k igeopt=1` perform a geometry optimization using the basis set 6-31G and the functional M06-2X. The value can also be a comma-separated list, and jobs are set up for each possible combination of values. For example,

```
jaguar run canonical.py -k basis=STO-3G,6-31G -k dftname=pbe,b3lyp
    input.mae
```

runs four jobs with the four possible combinations of basis set and functional. Calculations on isomers with the same basis set are run with the same number of canonical orbitals.

11.3.8 jreactions.py

The script `jreactions.py` calculates the energy of a reaction, and optionally the enthalpy and free energy. The reactants and products are specified on the command line.

The command syntax is:

```
jaguar run jreactions.py [options] -react reactant1
    [-react reactant2 ...] -prod product1 [-prod product2 ... ]
```

where *reactantN* and *productN* are Jaguar input file names. For a description of the options, run the command with the `-h` option.

11.4 Running Parallel Jobs

There are three different ways to use multiple processors to make Jaguar jobs run faster: distribution of multiple subjobs over processors, parallelization with MPI, and threading with OpenMP. Each has advantages and disadvantages, which are discussed below. Jaguar provides automatic selection of a fairly efficient way of parallelizing the execution of a job. This information is presented first, followed by the discussion of each particular way of using multiple procesors. Here, the term "processor" is used to cover any hardware unit that can run a process, such as a single core on a CPU unit.

11.4.1 Automatic Selection of Distribution and Parallel Options

While it is possible to set options for the three forms of parallel execution, it is usually best to allow the Jaguar startup script to determine the mix of job distribution, threads, and MPI processes from the command line by using the `-PARALLEL` *nproc* option. This is the recommended way of running jobs over multiple processors. The number of subjobs, MPI processes, and threads is set automatically. For example,

```
jaguar run filename.in -PARALLEL 12 -HOST queue
```

requests 12 processors, and the parallelization for this job, which has a single input file, would be equivalent to

```
jaguar run filename.in -OMPI 2 -TPP 6 -HOST queue
```

See the following sections for the syntax of the MPI processes and threads.

When you use `-PARALLEL` with multiple input files or structures, all subjobs are run on a single host. The value of *N* is ignored if you use the *hostname*:*N* syntax, and all hosts except the first are ignored if you use the `'host1 host2 ...'` syntax to specify the host.

This mechanism can also be used with Python batch scripts, e.g.

```
jaguar run script.py filename.in -PARALLEL 12 -HOST queue
```

The subjobs are distributed as well, and the number of subjobs to run concurrently is determined along with the number of threads and MPI processes. The exceptions are the `counterpoise.py` and `hydrogen_bond.py` scripts, which are still run sequentially. Generally, distribution of subjobs over available processors is done first, as this is the most efficient.

11.4.2 Distribution of Jobs

This method distributes separate Jaguar jobs ("subjobs"), each defined by an input file, over multiple processors. It is the most efficient method, and applies to all calculation types. The disadvantage is that it requires multiple licenses, one per subjob.

This method should be used whenever the number of subjobs to be run is equal to or greater than the number of processors available. The command syntax is one of the following:

```
jaguar run input-files -HOST hostname:njobs
jaguar run script.py input-files -HOST hostname:njobs
jaguar run script.bat input-files -HOST hostname:njobs
```

where *njobs* is the number of subjobs that you want to run simultaneously. This is by default the same as the number of processors that you use, because each subjob takes the resources of a single processor. The value of *njobs* does not have to be equal to the number of input files. If it is larger, then all the subjobs are run simultaneously. If it is less, then *njobs* subjobs are launched at once, and as soon as one finishes, the next subjob is run, until all subjobs finish.

This form of parallel execution is used when you launch a job from Maestro and you select multiple entries in the Project Table, and you choose a value greater than 1 in the CPUs text box in the Job Settings dialog box. It is also used when you select multiple CPUs and you run a rigid or a relaxed coordinate scan.

11.4.3 Message Passing Interface (MPI)

In this method, the work of a single job is distributed over multiple processors, by making use of a library that passes information between processors. The memory used by MPI can be distributed over multiple hosts. For more information on MPI, visit

http://www.mcs.anl.gov/research/projects/mpi

The advantages are that it can use multiple hosts, and it consumes only one license no matter how many processors are used. The disadvantages are that it requires extra scratch disk space and memory, it is somewhat less efficient than OpenMP (see the next section), and not all calculation algorithms can use MPI. Use of MPI is supported on Linux and Mac platforms, but not on Windows.

If you have a single job to run on a very large molecule, or an expensive calculation to run (like vibrational frequencies), MPI is useful in speeding up the calculation.

You do not need to install anything in order to run parallel MPI calculations with Jaguar. Jaguar is bundled with an MPI implementation called Open MPI (http://www.open-mpi.org).

The command syntax to use MPI is one of the following:

```
jaguar run filename.in -OMPI ncores
jaguar run script.py input-files -OMPI ncores
jaguar run script.bat input-files -OMPI ncores
```

where *ncores* is the number of processors (equal to the number of MPI processes) to use.

The Jaguar output file records whether your job is using MPI, and on which hosts the MPI processes are running, e.g.,

```
  Running   4 MPI processes

  Master MPI process running on bob-0-103.local
   Slave MPI process   1 running on bob-0-47.local
   Slave MPI process   2 running on bob-0-19.local
   Slave MPI process   3 running on bob-0-104.local
```

11.4.4 OpenMP threads

In this method, parts of the code are executed in parallel, using threads. See http://www.openmp.org for details.

The advantages are that it takes only one license no matter how many threads are used, it is somewhat more efficient than MPI, and more calculation types are parallelized with OpenMP than with MPI. The disadvantages are that it requires extra memory, and all threads spawned from a single process must run on the same host.

Like MPI, OpenMP is useful in speeding up expensive calculations for single jobs. It is usually a bit more efficient than MPI for the same number of processors, and more algorithms in Jaguar are parallelized with OpenMP than with MPI. Use of OpenMP is supported on Linux and Mac platforms, but not on Windows.

The command syntax to use OpenMP threads is one of the following:

```
jaguar run filename.in -TPP nthreads
jaguar run script.py input-files -TPP nthreads
jaguar run script.bat input-files -TPP nthreads
```

where *nthreads* is the maximum number of OpenMP threads to create for the multithreaded parts of the calculation. In practice you should not specify more threads than there are processor cores on the host, or you risk overloading the host. You can also set the number of threads in the Job Settings dialog box when starting a job from Maestro.

Because a team of OpenMP threads must run on the same host, you should be careful when using OpenMP on a cluster, so that you do not request more threads than there are cores on any one compute node.

When OpenMP is being used, the output file indicates the number of threads:

```
  Using up to   4 threads per process
```

11.4.5 Considerations for Running Parallel Jobs

Most Jaguar calculations can use at least one of MPI or OpenMP. Jobs that cannot be run with MPI include:

- LMP2 jobs other than closed-shell single-point calculations
- LMP2 jobs with more processors than LMP2 orbitals
- Jobs using more processors than there are atoms in the input structure
- CIS/TDDFT jobs
- Calculation of NMR and polarizabilities
- PBF solvation jobs

Jobs that cannot be run with OpenMP include:

- LMP2 jobs of any kind
- PBF solvation jobs
- HF/DFT ESP charge fitting

Properties that are quick to calculate and may not benefit from parallel execution include Mulliken analysis, Fukui indices, stockholder charges, thermochemical properties, multipole moments. Properties that are usually quick to calculate, but may benefit from parallel execution for large jobs, are NBO, and those properties that are evaluated on a grid for visualization: ESP, density, ALIE, NCI.

You can combine OpenMP and MPI options, in which case each MPI process spawns *nthreads* threads. This is one way to make an OpenMP job use multiple hosts. For example, if you wanted to use 16 cores for a Jaguar job on 4 nodes of a cluster in which each node had only a single quad-core processor, you could run the job with the following syntax:

`jaguar run` *filename*`.in -OMPI 4 -TPP 4 -HOST` *queue*

As long as the queue is configured to allocate one MPI process to a node ("round robin"), this command would run one MPI process on each node, and each of those processes would spawn 4 OpenMP threads.

A summary of the methods and properties that have been parallelized is given in Table 11.7. Note that you can run some calculation types with parallel options even though the calculations themselves do not run in parallel.

If you have many processor cores available and many large jobs to run through the same protocol, you can combine all three forms of parallelism. The most important thing is to get as many subjobs launched as possible since this is 100% efficient. For this purpose, it is recommended that you use the -PARALLEL option where possible. However, if you want to manage the parallel execution yourself, you can do so. For example, if you have *N* molecules (and at

Table 11.7. Parallelization of methods and properties

Methods/Properties (pseudospectral only)	**Parallelized with MPI**	**-OMPI job runs**	**Parallelized with OpenMP**	**-TPP job runs**
HF/DFT energies (and rigid scans)	Y	Y	Y	Y
HF/DFT gradients (geopts, TS, IRC, relaxed scans)	Y	Y	Y	Y
Closed-shell LMP2 energies (without solvation, charge-fitting, or multipole moments)	Y	Y	N	Y
ROLMP2	N	N	N	Y
Closed-shell LMP2 gradients	N	N	N	Y
CIS/TDDFT	N	N	Y	Y
Vibrational frequencies (der2b, der2ndb, cphfa)	Y	Y	Y	Y
HF/DFT ESP charge fitting	Y	Y	N	Y
DFT grid generation	Y	Y	Y	Y
NMR	N	Y	Y	Y
Polarizabilities	N	N	Y	Y
PBF solvation	N	Y	N	Y
SM6/SM8 solvation	N	Y	N	Y
non-pseudospectral calculations	N	Y	Y	Y

least *N* Jaguar licenses) and at least 2**N* cores available, then you could use the following command:

`jaguar run` *input-files* `-TPP 2 -HOST` *hostname*`:`*N*

This command runs all *N* subjobs at once, each using 2 threads. You could also use `-OMPI 2` instead of `-TPP 2`.

If you don't have enough processors to run all the subjobs at once but still want each subjob to run in parallel, you can limit the number of subjobs by setting *N* to the maximum number of subjobs to run at a time. In general, you can run a set of jobs from Jaguar input files or Maestro files with the syntax

`jaguar run` *input-files* `-TPP` *K* `-OMPI` *M* `-HOST` *hostname*`:`*N*

and the maximum number of processors used is $K \times M \times N$.

The workflow scripts (such as pK_a, fukui, VCD) all make use of parallel algorithms, and so benefit significantly from the use of multiple processors as long as you do not choose an unparallelized level of theory like ROLMP2. As PBF solvation is not parallelized, a pK_a job on a large molecule will lose some efficiency if using MPI; however, the subjobs in this workflow can be distributed. Workflow scripts typically are used on more than one structure, so these scripts also benefit from subjob distribution. The exceptions are the `counterpoise.py` and `hydrogen_bond.py` scripts, which are still run sequentially.

MPI and OpenMP are resource-intensive, and you can overload a computer by requesting too many MPI processes or OpenMP threads. A parallel job using 2 MPI processes or OpenMP threads will use almost twice as much memory as a single-processor job, at least for those portions of the job that can be run in parallel. A 2-processor MPI job will use almost twice as much scratch disk space as a single-processor job. If too many Jaguar MPI processes are run on the same host, it can slow the job down appreciably due to too many processes competing for use of the scratch disk.

Also keep in mind that MPI and OpenMP are not 100% efficient, nor is their scaling linear. Thus a 4-processor job does not run 4 times faster than a 1-processor job, and a 32-processor job does not run twice as fast as a 16-processor job. The efficiency depends on the computer hardware, on the kind of calculation you are doing, on the level of theory, and on what portion of the program code has been enabled to use MPI or OpenMP.

Considering the above advantages and limitations of the available parallel methods, we recommend distributing as many subjobs simultaneously as you can. If you still have more processors available, run each subjob with 4 OpenMP threads.

You can set the `-OMPI` or `-TPP` options from Maestro as well as the maximum number of subjobs to run simultaneously. In the Job Settings dialog box, select the MPI processes option. Three text boxes are displayed, for the MPI processes, threads, and maximum simultaneous subjobs. The total number of processors in use at any time is the product of the three values in these text boxes. The job is launched with `-OMPI` *P* `-TPP` *T* `-HOST` *hostname*:*J*, where *P* is the number of MPI processes, *T* is the number of threads, and *J* is the number of subjobs.

If instead you select CPUs and enter a value into the CPUs text box, this is interpreted as the number of processors to use. If you select just one entry, then the Jaguar startup script determines whether to use OpenMP or MPI or a mixture of the two, based on the number of CPUs that you specify for the job.

Troubleshooting

Naturally, we hope that you will never need to use this chapter. However, if you have problems using Jaguar, you may find useful advice here. If you don't, feel free to contact us as described on page 372.

For problems with settings, you might find the information you need in the online help. You can obtain help from any panel by clicking its Help button. The Help panel opens with the appropriate topic selected.

12.1 Problems Getting Started

If you are having problems starting Maestro or submitting jobs, read this section.

Your local system manager should have already installed Jaguar. If the command

```
$SCHRODINGER/maestro &
```

does not work because the `maestro` command does not exist or if you get an error message regarding installation, contact this person.

The exact wording of error messages you get when trying to run Jaguar might differ from the error messages described here, depending on your hardware and X implementation. Remember that your X server is either your workstation or the machine that acts as the server for your X terminal, the display host is the workstation or terminal at which you are sitting, and you are trying to start Jaguar as an X client on some machine not necessarily serving as your X server.

Some of the issues addressed here are standard X windows or UNIX issues, and consulting your X and UNIX documentation may help. Also, you may be able to avoid repeatedly entering commands described in this section by including them in your `.login`, `.cshrc`, or other startup files in your home directory.

If you can start Maestro but you have problems submitting jobs, skip to Section 12.1.3 on page 343 and Section 12.1.4 on page 344.

12.1.1 The SCHRODINGER Environment Variable

Before running Jaguar on any particular Linux machine, you must set the environment variable `SCHRODINGER` to point to the installation directory on that machine. This is the directory containing Jaguar version 9.0, which is in a subdirectory called `jaguar-v`*xxxxx*, where *xxxxx* is the five-digit version number.

To check whether the `SCHRODINGER` environment variable is set, enter the command

```
echo $SCHRODINGER
```

If the output from this command is a directory containing Jaguar, you can skip the rest of this subsection.

If you determine that the `SCHRODINGER` environment variable has not been defined, you must set it. If you don't know where the installation directory is, ask the person who installed Jaguar. Then, depending on your shell, enter one of the following commands:

csh/tcsh:	`setenv SCHRODINGER` *installation-directory*
bash/ksh:	`export SCHRODINGER=`*installation-directory*

You should also set the `SCHRODINGER` environment variable in your shell startup file (in the `.cshrc` file in your home directory if you are running C shell, for instance) by adding the `setenv` or `export` command to the file, so that it is defined for any shell that is used, whether interactively or in a batch job.

12.1.2 Problems Starting Maestro

If you have problems when you try to start Maestro, they are likely to involve permissions needed to do things over a network. Most of these problems never arise if the machines you are using are within a local network. If you are using only local hosts and still have these problems, you might ask your system manager for advice in addition to following the instructions given here.

If you get the message:

```
Error: Can't Open display
```

you are probably trying to start Maestro from a machine that is not acting as your X server, and this machine does not know what your display is. Before starting Maestro, you can specify the display with the following command, substituting the name of your X server or terminal for *displayhost*.

csh/tcsh:	`setenv DISPLAY` *displayhost*`:0.0`
sh/bash/ksh:	`export DISPLAY=`*displayhost*`:0.0`

The error message

```
Xlib:  connection to "displayhost:0.0" refused by server
Xlib:  Client is not authorized to connect to Server
Error: Can't Open display
```

usually means one of two things. First, if you are not the person who initially logged on to the X server, you cannot bring up any type of X window on the display. In this case you should log out and log in as yourself. Second, if your X server and the host from which you start Maestro (the "launch host") are not the same machine, the X server might not recognize the right of that host to display. To correct this problem, type the following in a window on your X server:

```
xauth nextract - displayhost:0.0 | rsh ihost xauth nmerge -
```

Here, *ihost* should be replaced by the name of the launch host. Also, the "remote shell" command should be used for `rsh`; usually `/bin/ucb/rsh` serves this purpose, but `rsh` gives `/usr/bin/remsh` on some machines. If the restricted shell `rsh` precedes the remote shell version in your path, you must use the full path name. If the `xauth` command listed above results in the message `xauth: Command not found.`, your path probably does not include `/usr/bin/X11`, and you should include `/usr/bin/X11` in your path. You could also substitute `/usr/bin/X11/xauth` for `xauth` in the command and try again. If the `xauth` command yields `Permission denied`, the `rsh` command was not allowed, and you should read the paragraphs on `rsh` and `rcp` commands in Section 12.1.4 on page 344.

If you have problems running the `xauth` command described in the above paragraph, an alternative is to simply type:

```
xhost +ihost
```

on your X server. This command allows anyone (including yourself) logged onto *ihost* to run X programs on *displayhost*. Since this command is a potential security risk, it is not recommended as a permanent solution.

12.1.3 Problems Related to Your Temporary Directory

When you run a Jaguar job, Jaguar generates various files it needs during the calculation within a temporary directory (often within a directory called `/scr`, `/tmp`, or something similar). At the end of the job, the program deletes most files in this directory by default, copying back only the output file and any other files you requested. If you get an error related to temporary directory space when you try to run Jaguar, the program is probably having trouble getting access to the temporary directory space it needs to run.

If you are using Maestro to run jobs, you can tell what temporary space Jaguar will try to use by looking at the Scratch directory setting in the Start dialog box. The program actually makes a subdirectory named after the job within this directory and writes files there. For instance, if a person with the user name `erwin` has a Scratch directory listing of `/scr` for a job called `h2o`, Jaguar attempts to create the directory `/scr/erwin/h2o` and write files there during the job.

If your job gives error messages related to the temporary directory, you should check to make sure that the temporary directory listed in the Start dialog box exists and that you have write permission within that directory. For example, if the output

```
Error creating or cd-ing to temp directory:
   /scr/erwin/h2o
```

appeared in the `h2o.log` file for `erwin`'s job, it could be because `/scr/erwin` did not exist or because `erwin` did not have permission to make the subdirectory `h2o` within it. If you are running parallel or distributed jobs, you might not have permission to create a directory on one of the hosts.

You might need someone to create the appropriate temporary directory or change permissions on it from the root account. Use the command `ls -l` to get information on ownership of your temporary directory or the directory above it. If you need to be able to create a subdirectory within a directory owned by root or another account that does not belong to you and for which you do not have write permission, contact your system administrator for help.

12.1.4 Problems Running Jaguar Calculations on Other Hosts

Running jobs on remote hosts requires that you set up a `schrodinger.hosts` file, and have communications properly set up between the hosts. Information on these tasks can be found in Chapter 6 of the *Installation Guide*.

If you get an error that refers to problems writing or changing to a temp directory for the job, you should make sure that you have permission to write to the directory specified in the Scratch directory option menu in the Start dialog box, and that you have permission to create that directory if it does not already exist.

One further complication can result if you have distinct user names on the local and execution hosts. In this case, you will get login errors. To handle these distinct hosts, you must use a personal `schrodinger.hosts` file. Each host line in the file should include your user name on that host in the following format:

`user:` *username*

where *username* is replaced by your user name. See Section 6.1 of the *Installation Guide* for details on how to construct your own `schrodinger.hosts` file.

12.2 Other Problems

Some other problems you may encounter are detailed below, along with solutions or explanations.

- *The molecular structure for the calculation is not what you expected it to be.* If you read in a Jaguar input file, the geometry is obtained from that file, unless you edit the geometry after reading the file. Also, if you symmetrize the geometry or set symmetry on for the calculation, as described in Section 2.7.2 on page 23, Jaguar may make small changes to the molecular geometry. If these changes are a problem, you should avoid symmetrizing the geometry and possibly turn the symmetry option off as well.

- *The calculation is not what you expected it to be.* If you read in a Jaguar input file some of the settings in the file take precedence over settings previously made in the GUI. See Section 2.5 on page 20 for more details. Also, certain settings affect other settings automatically—for instance, if you choose to calculate polarizabilities, the energy convergence criterion can be reset to 1.0 x 10^{-6}.

- *The SCF calculation does not converge properly, or frequencies or other properties look wrong.* If the geometry entered is of poor quality, the calculation may not converge properly, which may also lead to inaccurate calculation of molecular properties. If you are performing a geometry optimization, check to see whether the geometry changes are reasonable; if you are performing a single-point calculation, make sure the structure entered is appropriate. You might want to minimize the structure with a molecular mechanics program first. See Chapter 7 for more tips. If the structure is reasonable, convergence problems should not occur, and we would appreciate it if you would contact technical support (see page 372), and provide the input, output, and log files for the job with a brief explanation. To get converged results in the meantime, you can try using level shifting or setting the accuracy level to Ultrafine, both of which are described in Section 3.7.4 on page 56 and Section 3.7.1 on page 54. The calculation will be slower, but convergence may be better for problem cases.

- *The settings available in the* Job Settings *dialog box are not what you expected them to be.* Many of the options are determined by the hosts file used for the job. This file is the `schrodinger.hosts` file found in the directory from which Maestro was started, if it exists; otherwise, it is the `schrodinger.hosts` file in your home directory, if that file exists; and if neither of those two files exists, the default `schrodinger.hosts` file for the submission host is used. If you are using a different hosts file from what you expect, or if you are working with a new version of Jaguar and a new hosts file has been installed on your computer, you should examine the hosts file for the job and make sure it is in the correct form and that the settings are appropriate. See the *Job Control Guide* for a description of the `schrodinger.hosts` file.

- *The job fails with a memory-related error ("Memory fault," "out of memory," or "no memory available for array," for example) or the log file indicates "Killed" for the job.* Your job may have failed because the machine was too heavily loaded, in which case rerunning the job when the load is lower could solve the problem. Otherwise, you might want to try an appropriate setting from Section 9.5.28 on page 265 to avoid a problem for a large job, or you (and/or your system manager) might want to investigate increasing the maximum virtual size or the "soft" limit allowed for memory on your machine. Contact us, as described on page 372, if you would like any tips for setting memory use for your machine.

The pK_a Prediction Module

Schrödinger's pK_a prediction module represents the first attempt to use ab initio quantum-chemical methods to reliably predict pK_a values in aqueous media [236]. The module uses a combination of correlated ab initio quantum chemistry, a self-consistent reaction field (SCRF) continuum treatment of solvation, and empirical corrections to repair deficiencies in both the ab initio and continuum solvation models. This combination leads to high accuracy for a wide range of organic compounds, in conjunction with tractable computational requirements.

The user interface has been designed to avoid the necessity of running the many individual jobs required to assemble the various components of the calculation. Schrödinger has optimized each of the components for the best tradeoffs of accuracy versus efficiency. The empirical correction terms, which have been developed for ionizable groups relevant to the chemical and pharmaceutical industries, are specifically designed to work with the basis sets, electron correlation levels, and solvation model of the ab initio methodology. The transferability of the corrections has been tested by examining a sizeable set of test molecules.

Several features of the method distinguish it from purely empirical, fragment-based approaches, which are complementary to the present product. First, we expect that the use of ab initio quantum chemistry rather than fragment table lookups and interpolation will lead to a substantially wider range of applicability, as well as significantly higher precision when the compound in question is not a direct entry in the empirical table. Second, our methods allow for a reasonable treatment of conformational effects, which are in general entirely missing from fragment-based methods. Optimal use of the methodology in this fashion is accomplished by performing solution phase conformational searches with MacroModel. Third, the method can handle multiple protonation states in a systematic fashion.

This chapter is divided into four sections. First, the basic theory of pK_a calculations is explained, including a discussion of the empirical correction approach. Then, a discussion of key issues in using the program in complex situations (conformational flexibility, multiple protonation states) is given. Thirdly, results from our internal suite of test cases are presented. Finally, a practical tutorial describing how to set up, run, and interpret jobs is presented.

13.1 Theory of pK_a Calculation

13.1.1 Ab initio Quantum Chemical Calculation of pK_a Values

The calculation of the pK_a of a molecule in aqueous solution can be represented as a thermodynamic cycle:

$$\begin{tikzcd}
BH^+_{(g)} \arrow[r, "A"] \arrow[d, "B"] & B_{(g)} \arrow[d, "C"] & + & H^+_{(g)} \arrow[d] \\
BH^+_{(aq)} \arrow[r, "D"] & B_{(aq)} & + & H^+_{(aq)}
\end{tikzcd}$$

The strategy in our pK_a module is to calculate parts *A, B*, and *C* of the above cycle, whereupon the actual pK_a, which is related to *D* by

$$pK_a = \frac{1}{2.3RT}D$$

can be obtained by summing the free energy changes for these three components and the experimental value of –259.5 kcal/mol for the solvation free energy change of a proton.

Segment A is the gas phase reaction:

$$BH^+{}_{(g)} \xrightarrow{A} B_{(g)} + H^+{}_{(g)}$$

The gas phase free energy difference between the protonated and deprotonated states can be computed via the usual relations

$$\begin{aligned} A &= \Delta H - T\Delta S \\ &= E_{B(g)} - E_{BH+(g)} + 5/2RT - T\Delta S \end{aligned}$$

Evaluation of this expression requires the following quantum chemical calculations:

1. *Geometry optimization* of the protonated and deprotonated species. Quantum chemical methods generally carry out a conjugate gradient optimization and hence cannot search for multiple minima. We assume here that there is only a single, well-defined conformational minimum and that a good initial guess, obtained, for example, from molecular mechanics or semiempirical quantum chemistry, is available. Density functional theory, particularly the hybrid methods that include Hartree-Fock exchange, have been shown to provide good quality geometries; we utilize B3LYP/6-31G* geometry optimization.

2. *Accurate single-point energies* at each optimized geometry must be evaluated. These single-point calculations are carried out at a significantly higher level of theory than the geometry optimization, but since only one energy is required, the overall cost of this step is less than that for geometry optimization. In recent publications, and in our own extensive unpublished work, the B3LYP method with large basis sets has been shown to yield excellent gas phase energetics for deprotonation reactions, with errors typically in the 1-3 kcal/mol range. We use the cc-pVTZ(+) basis set of Dunning and coworkers in the present methodology. The cc-pVTZ(+) basis set represents a mixed basis set where cc-pVTZ+ is used for atoms involved in the deprotonation reaction, while cc-pVTZ covers the rest. The residual errors in the DFT calculations appear to be relatively constant for a given functional group as the substituents are altered, and hence can be largely removed by the empirical corrections.
3. *The solvation free energy* of the protonated and deprotonated species must be computed. We have chosen to do this using the gas phase geometries, an approximation that we have tested and shown to be sufficient for the present purposes (some of the errors induced are compensated by the empirical parameterization).

As we have discussed extensively in several publications, empirical optimization of parameters is absolutely necessary to obtain accurate solvation free energies from SCRF calculation, no matter what the level of electron correlation. Continuum solvation methods do not rigorously treat effects at the dielectric boundary, which therefore must be adjusted to fit experiment.

For neutral species, we have optimized parameters (both dielectric radii and surface tension terms) by fitting to experimental gas to water solvation free energy data for small molecules. Agreement to within a few tenths of a kcal/mole can be obtained for most functional groups. However, parameterization of the model for ionic species in this fashion cannot lead to high levels of accuracy, because there are large error bars on the experimental data (typically 5-10 kcal/mole). An error of 5 kcal/mol in the solvation free energy that was not systematic would lead to huge errors in pK_a calculations. This is because in determining the pK_a there is a cancellation of two very large terms: the gas phase deprotonation energy (which favors the protonated state) and the solvation free energy (which favors the deprotonated state). Errors in either term therefore can be a small percentage of the total energy but lead to very large errors in the resulting calculated pK_a.

To overcome this problem, the parameters for ions are fitted *directly to experimental pK_a data.* If the gas phase quantum chemistry and neutral solvation are reliably computed, then the solvation free energy of the ionic species becomes the remaining unknown quantity. Since pK_a measurements are carried out to quite high precision (in contrast to direct measurements of ionic solvation), fitting to this data does not lead to the large uncertainties that would be associated with the ionic solvation data. Additionally, there is an exceptionally large database of known pK_a values for a wide range of chemical functional groups.

In general, the dielectric radii of ions (particularly negative ions) are expected to be smaller than that for the corresponding neutral species, due to the phenomenon of electrostriction. In our fitting procedure, the ionic radii are adjusted to yield the smoothest and most consistent results for the members of the training set for each functional group. For anions, special radii are assigned to the principal location of the negative charge; for cations, radii are assigned to hydrogens on the proton acceptor and to the proton acceptor itself. Functional groups for which radii have been developed are listed in Table 13.1 on page 354. For novel functional groups with divergent electronic properties, reparameterization of the model to a subset of experimental data is advisable, as the results are rather sensitive to these quantities. However, the current model is able to robustly handle substituent and conformational effects once a functional group is parameterized.

In our work on neutral solvation, we have found that it is necessary to supplement parameterization of dielectric radii with surface area terms to correct for first shell hydrogen bonding. A purely electrostatic model is incapable, by itself, of properly describing such interactions for all molecules. For ions, these terms are expected to be even larger and more important, as the magnitude of the first shell hydrogen bonding interactions are 3-5 times larger than in neutral species. However, what we have done in the present model is to incorporate these corrections into our overall empirical fitting scheme, described below. In this fashion, all of the errors associated with the various components of the method are subsumed into a small number of parameters characteristic of the functional group in question.

13.1.2 Empirical Corrections

The results of the above calculation can be assembled to yield a raw pK_a value. Because of the intrinsic errors involved in each step, it is necessary to apply an empirical correction scheme to the raw data to yield good agreement with experiment. The validity of this scheme can be assessed only by comparison with experimental data. For the most important functional groups, we have examined a large and diverse set of molecules (including those containing polyfunctional groups and conformational flexibility) to evaluate the robustness of the methodology. For the molecules considered below, it appears to be quite satisfactory. For example, for protonation of nitrogens in heterocycles, an average prediction accuracy of 0.19 is obtained over 16 molecules whose pK_a values range from 0.65 to 9.17.

Our empirical corrections take the simple linear form:

$$\mathrm{p}K_\mathrm{a} = a\ \mathrm{p}K_\mathrm{a}(\mathrm{raw}) + b \qquad (1)$$

That is, we assume that the correction terms obey a linear free-energy relationship. The *b* term is similar to our previously employed surface tension corrections for solvation of neutral species. The linear term takes into account the significant variation in charge on the ionizable group as a function of substituents. Consider, for example, carboxylic acids. The charge on the oxygens in the CO_2^- moiety varies by as much as 0.45 eu when electron-withdrawing substituents (such as in oxalic acid) are replaced by electron donating substituents (such as in propionic acid). This change in charge alters the hydrogen-bonding first-shell correction term as well as the solvation free energy computed by the SCRF calculation. Since changes in the raw pK_a are well correlated with these charge shifts, linearly scaling the correction term to the raw pK_a is capable of capturing this effect.

While corrections to the solvation model are the dominant terms in our empirical corrections, there are also intrinsic errors in the gas phase DFT calculations, which are implicitly incorporated into the correction scheme. The assumption is that these errors are systematic for a given functional group. This means that the DFT calculations are required only to reproduce the relative energetic changes produced by modification of substituents, a less demanding task than absolute pK_a prediction. As the accuracy goal (0.5 pK_a units) is beyond the capabilities of the raw DFT calculations, empirical corrections are necessary.

13.2 Predicting pK_a Values in Complex Systems

The algorithm described in Section 13.1 on page 348 can be straightforwardly applied in the simplest cases, which are characterized as follows:

1. There is only one relevant ionizable group in the molecule.
2. There is a single relevant conformation of the molecule, and this conformation is valid for both the protonated and deprotonated form.

An example of this situation is acetic acid. However, it is also possible to use the module in more complex situations. In the following sections, we explain how this is accomplished.

13.2.1 Conformational Flexibility

First, consider the case in which assumption (1) above holds, but the protonated and deprotonated states can each exist in multiple conformations, which might be energetically competitive. There are several possible ways in which the conformational problem can be addressed, both of which are available in the current release.

1. Perform calculations on one protonated and one deprotonated conformation, which are assumed to dominate the phase space due to being lowest in energy in their class. This is a reasonable assumption for many problems. Note that *the lowest-energy conformation can be different for the protonated state and the deprotonated state.* In many cases there are obvious electrostatic reasons why a conformational change on protonation or deprotonation would occur. The program can accept a different conformation for each species.

The selection of the appropriate conformation can be nontrivial. Our recommendation is to do a solution phase conformational search in MacroModel, using the OPLS_2005 force field and the GB/SA continuum solvent model. This is a very fast procedure and gives a reasonable ordering of conformational free energies in solution. This procedure has been automated and can be used from Maestro. Alternatively, you can either construct the conformation by hand or use a gas-phase conformational search. Preliminary results indicate that there are situations where a solution-phase conformational search is necessary to obtain accurate results.

2. Perform quantum chemical calculations for multiple conformations, generated from a MacroModel solution phase conformational search, and use all of this information to compute the pK_a. Two ways of doing this are

   a. Pick the conformer that has the lowest solution-phase free energy for each protonation state and compute the pK_a from this value. This method is analogous to (1) above but allows for imprecision in the conformational search protocol. It also takes more CPU time.

   b. Carry out a statistical mechanical average over conformations to determine the average pK_a. The assumption made if this option is chosen is that the midpoint of the pK_a titration curve is achieved when the total population of the deprotonated state, summing over all deprotonated conformations, is equal to the total population of the protonated state, also summing over all conformations. This approach should be more accurate than that described in (a), although how important statistical effects are in practice remains to be ascertained.

13.2.2 Equivalent Sites

Some molecules have two or more equivalent sites for protonation or deprotonation. Examples include ethanediamine, the analogous dicarboxylic acid, or the molecule melamine, which has three equivalent sites. In this situation, a statistical correction factor arises from increased entropy of the appropriate species. Jaguar automatically recognizes these equivalent sites and includes the appropriate correction of $\pm\log_{10}(N)$, where N is the number of equivalent sites.

You can turn off the entropic correction by setting `ipkasymm=0` in the **gen** section of the input file. By default it is set to 1, to include the entropic correction. To print the entropic correction, set `ipkaverbose=1` in the **gen** section; by default it is set to 0 and the correction is not printed separately.

13.2.3 Multiple Protonation Sites

Many molecules have several sites, which can have different pK_a values. Consider a case with two distinct possible protonation sites for which we want to calculate the pK_a of site 1. Then the following situations are possible:

1. The two pK_a values are well separated and the pK_a of site 1 is higher than that of site 2. In this case, site 2 will be deprotonated when site 1 is being titrated in an experiment. The pK_a calculation for site 1 is run with site 2 in the deprotonated state.
2. The two pK_a values are well separated and the pK_a of site 2 is higher than that of site 1. In this case, site 2 will be protonated when site 1 is being titrated in an experiment. The pK_a calculation for site 1 is run with site 2 in the protonated state.
3. The two pK_a values are unknown, or the pK_a values are close together. In this case, there are a total of four protonation states to run: both sites protonated, one site protonated (two cases), and no sites protonated. If one obtains data for these four cases, the titration curve can be assembled and one can make comparison with experiment.

When the ionizable groups are close together in the molecule, the calculated pK_a may not be as accurate because the two groups could interact in ways that the existing parameterization cannot handle. For case (3), you must run two separate pK_a jobs, each of which handles two of the four protonation states, and build the titration curve by hand.

13.3 Training Set Results

Table 13.1 presents a summary of the results for the functional groups for which parameters are available, including the root mean square deviation (RMSD) from experiment. The functional groups are classified as acids and bases: for bases, it is the pK_a of the conjugate acid that is computed. Unless otherwise noted, the values are for aqueous solution.

The largest sets of test cases examined have been for carboxylic acids and nitrogen bases in heterocyclic rings. The latter cases have minimal conformational flexibility and hence should be easier to handle, and this is indeed reflected in the remarkably low average error. Molecules that have intramolecular hydrogen bonds have been excluded from the training set (except for o-nitrophenols), because their pK_a values are strongly affected by the hydrogen bond. Because of this, pK_a predictions for such molecules are likely to have large errors.

The carboxylic acids include some examples with polyfunctional groups and significant flexibility. We have not carried out an exhaustive analysis of the conformational energetics for these cases; hence much of the deviation from experiment that we report may be due to this. Nevertheless, the errors are quite respectable. Alpha-keto and beta-keto carboxylic acids and other conjugated carboxylic acids are parametrized separately.

Table 13.1. Parametrized functional groups, with deviation from experiment.

Functional Group	RMSD	Max
ACIDS		
1,2,3-triazole	0.9	1.5
1,2,4-triazole	0.8	2.0
4-hydroxycoumarin	0.2	0.3
4-pyridone as acid	0.1	0.1
4-pyrimidone as acid	0.5	0.9
5-2H-isoxazolone	0.9	1.4
6-member aromatic enol with neighboring C(C=O)R	1.1	1.9
6-member aromatic enol with neighboring C=O	0.2	0.3
6-member aromatic enol with neighboring C=O and amide	0.2	0.5
8-hydroxyquinoline	0.3	0.8
alcohol	0.2	0.4
Alk-(C=S)-NH-Ar as acid	0.3	0.7
alpha-keto carboxylic acid	0.5	1.3
amide as acid	0.9	1.4
Ar-(C=S)-NH-Ar as acid	0.2	0.5
aromatic-N-OH acid	0.6	1.5
arsinic acid	0.1	0.4
arsonic acid	0.2	0.4
barbituric acid	0.5	0.9
benzimidazole as acid	0.6	1.2
beta-keto carboxylic acid	0.7	1.4
boronic acid	0.5	1.2
carboxylic acid	0.7	1.6
carboxylic acid, conjugated	0.6	1.5
carboxylic acid, non-conjugated	0.8	1.6
cyclopentadiene CH acid	1.4	2.5
delta lactam	0.5	1.0

Table 13.1. Parametrized functional groups, with deviation from experiment.

Functional Group	RMSD	Max
diketo/mono sulfone CH acid	0.3	0.7
enol	0.5	1.1
heterocylic CH acid	0.7	1.3
hydrazone	0.6	1.2
hydroxamic acid	0.4	0.7
imidazole as acid	0.6	1.1
imide	0.7	1.4
mono keto CH acid	0.6	1.6
multi-keto/-sulfone (except diketo) CH acid	0.8	1.5
NH-P=O acid	0.7	1.0
nitro-CH acid	0.7	1.6
OH attached to heterocycle	0.6	1.1
OH attached to heterocycle (with one N)	0.8	1.5
organophosphoric acid	0.2	0.5
organophosphoric diacid	0.4	0.9
organosulfuric acid	0.5	1.0
oxime	0.9	2.4
peroxide	0.4	0.7
phenol, excluding o-nitro- and o-nitrosophenols	0.7	2.1
o-nitro or o-nitroso phenol	0.7	1.4
phosphonic acid	0.4	1.1
phosphonic diacid	0.3	0.6
pyrazole as acid	0.4	0.7
R-(C=S)-NH-Alk as acid	0.3	0.6
seleninic acid	0.2	0.7
SH attached to heterocycle	0.6	1.1
SO2-NH-CO acid	0.9	2.6
sulfinic acid	0.3	0.9

Table 13.1. Parametrized functional groups, with deviation from experiment.

Functional Group	RMSD	Max
sulfonamide	0.9	2.3
sulfonamide, unsubstituted	0.5	1.3
sulfonic acid	1.2	2.1
tetrazole	0.6	1.2
thiol	0.3	0.7
thiophenol	0.2	0.4
thiourea derivatives	0.2	0.2
tropolone	0.5	0.9
urea derivatives	0.8	1.5
ACIDS in DMSO		
CH acid	1.6	3.4
NH acid	1.3	2.7
BASES		
4-pyridone as base	0.5	0.9
4-pyrimidone as base	0.4	0.7
amide as base	0.6	0.9
amidine	0.7	1.7
fully N-substituted amidine	0.6	1.3
partly N-substituted amidine	0.7	1.2
unsubstituted amidine	0.6	1.6
benzimidazole as base	0.7	2.0
benzodiazepine	0.5	0.7
enamine[a]	0.6	0.9
ether	0.3	0.5
guanidine	0.3	1.1
heterocycle[b]	0.8	2.9
heterocycle custom 1[c]	0.6	1.8
hydroxylamine	0.3	0.6

Table 13.1. Parametrized functional groups, with deviation from experiment.

Functional Group	RMSD	Max
imidazole as base	0.6	2.2
imine	0.9	1.7
indole C-3 protonation	0.4	1.0
N-oxide	0.2	0.5
aromatic N-oxide	0.1	0.1
NR2 attached to heterocycle	0.1	0.5
NR2 attached to heterocycle (meta only)	0.5	0.7
oxazole	0.2	0.7
piperazine	0.5	1.4
primary amine	0.4	1.2
primary aniline	0.1	0.7
protonation of O=Het	0.6	1.3
protonation of O=Het=R	0.6	1.0
pyrazine	0.3	0.6
pyrazole as base	0.5	1.0
pyrazoline	0.3	0.5
pyridazine	0.3	0.4
pyridine	0.5	1.3
pyrimidine	0.2	0.6
pyrrole C-2 protonation	0.3	0.5
quinoline	0.4	0.8
R2N-NH2 hydrazine[d]	0.3	0.5
R2N-NR2 hydrazine	0.3	0.4
R2S=N-C(=O) ylide	0.3	0.8
RN=Het	0.6	0.7
secondary amine	0.5	1.9
secondary aniline	0.3	0.8
tertiary amine	0.7	1.6

Table 13.1. Parametrized functional groups, with deviation from experiment.

Functional Group	RMSD	Max
tertiary aniline	0.4	0.9
thiazole/benzothiazole	0.6	1.6
thioamide as base	0.2	0.3
thioether	2.6	3.7
ZWITTERIONS		
amino acid as acid[e f]	0.2	1.0
amino acid as base[e f]	0.7	2.7
Protonation of [R3N/R3P/R2S](+)-...-ArO(-) zwitterion	0.8	2.0
Protonation of [R3N/R3P/R2S](+)-...-COO(-) zwitterion[g]	0.4	2.5

a. for C-1 protonation
b. any 5- or 6-membered aromatic ring containing N
c. same as heterocycle except the 5-membered rings have two carbons not attached to the pK_a atom
d. for protonation at the substituted N
e. Jaguar optimization done in solution
f. forces conformational search off (even if specified)
g. uses distance corrections

13.3.1 Jaguar pK_a Shell Model

A functional group that is not identified by Jaguar as one of the functional groups listed in Table 13.1 is treated differently. Because there is no parameterization available for that specific group, Jaguar treats the group more generally, and comes up with the best possible parameterization for it, using the training data from Table 13.1. This is the essence of the "shell model" which identifies each functional group as a series of matched nested patterns of increasing specificity. The least specific shell is the generic shell. It is matched by any functional group. The most specific shells are those listed in Table 13.1. Typically, the more specific the shell the more accurate its parameterization (the smaller the RMSD of its linear fit). The RMSDs and maximum deviations for these less specific shells are given in Table 13.2. The Jaguar pK_a output file prints all the shells that match along with the pK_a values predicted by the corresponding models. The list of results is arranged in the order of increasing RMSD. This means that the top shell's result are the most trustworthy. Maestro only incorporates the pK_a coming from the shell with the smallest RMSD (the top shell in the output file).

The shell model guarantees that there is a pK_a parameterization model associated with any functional group. Thus, it provides an essentially 100% coverage of chemical space. The shell

Table 13.2. Upper level shells for which pKa parameters are available.

Functional Group	RMSD	Max
Aliphatic CH acid in water	2.6	5.9
aromatic OH w/ neighboring C=O	0.7	1.3
Protonation of R^+-...-O^- zwitterion	2.0	3.8
Protonation of generic anion to zwitterion R^--...-RH^+	0.7	2.7
Protonation of generic atom	2.3	8.0
Protonation of generic zwitterion R^+-...-R^-	2.0	3.8
Protonation of sp^2-like aliphatic carbon	2.6	6.0
Protonation of sp^2-like aliphatic nitrogen	1.8	6.3
Protonation of sp^2-like aliphatic nitrogen to zwitterion R^--...-$NR2H^+$	0.7	2.7
Protonation of sp^2-like aromatic nitrogen	1.0	2.7
Protonation of sp^2-like oxygen	2.6	6.4
Protonation of sp^2-like sulfur	2.2	4.4
Protonation of sp^3-like aliphatic nitrogen	1.7	6.4

model is activated by the input file setting `ipkashell=1`, which is the default. In order to disable the shell model and use an earlier Jaguar pK_a parameterization, set `ipkashell=0`.

In some cases it might be advantageous to develop your own parameterization for functional groups of interest—see Section 13.5 on page 368.

13.3.2 Zwitterions

Parameterization of zwitterion functional groups is available for the following groups, which are also listed in Table 13.1:

- Protonation of $[R3N/R3P/R2S]^+$-...-COO^- zwitterion (carboxylic acids)
- Protonation of $[R3N/R3P/R2S]^+$-...-ArO^- zwitterion (phenols)
- amino acids as acid
- amino acids as base

A zwitterion intercharge distance-dependent term was found to be necessary for the protonation of $[R_3N/R_3P/R_2S]^+$-...-COO^- zwitterions. The term is of the form Ar^{-2}, where r is the zwitterion intercharge distance. Where the charge can be spread out over multiple atoms, the smallest of the distances is used. The results are fairly insensitive to the exponent. The value of A used is -14.427466.

Amino acid zwitterion species undergo proton transfer to yield the neutral species when optimized in the gas phase, and so solution-phase optimizations are done instead. The conformational search was also found to degrade the results, so it is turned off.

Generalized zwitterion shells (as part of the Jaguar shell model) are also available. The groups covered in this model are listed in Table 13.2.

13.4 Running pK_a Calculations

pK_a calculations consist of a series of calculations on the protonated form and on the deprotonated form of the target molecule, followed by an empirical correction. The calculations are performed using a Jaguar batch script. You need only supply an input file with the acidic or basic atom marked, and use the batch script to run the calculation. You can set up pK_a calculations in Maestro, or prepare the input files in a text editor and submit the jobs from the command line.

The pK_a module requires a special license in addition to the regular Jaguar license. When you request a Jaguar license, you should explicitly indicate in your license request that you want to run pK_a calculations.

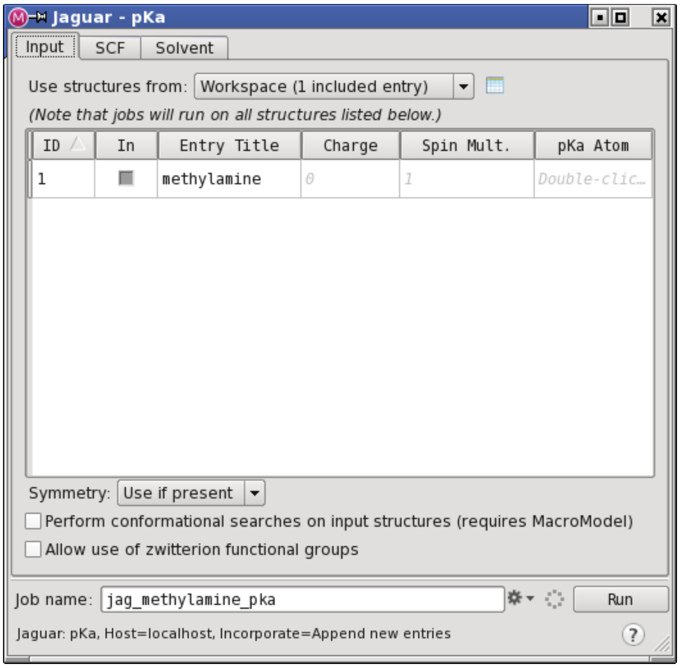
13.4.1 Running pK_a Calculations from Maestro

To set up and submit pK_a calculations from Maestro, select the entries in the Project Table or Entry List for which you want to calculate the pK_a, then choose Applications → Jaguar → pKa or Tasks → Quantum Mechanics → pKa in the main window. The Jaguar panel opens with the Input tab displayed.

For each structure, the atom or atoms whose pK_a you want to calculate can be specified in the pKa Atom cell in the Input tab. Click in this cell, then pick the atoms in the Workspace, or enter the atom names separated by commas. If the structure is not already in the Workspace, it is placed in the Workspace (as the only structure). If you pick more than one atom, calculations will be run for each atom that you pick, and thus you can calculate the pK_a for each site in a single molecule that has multiple protonation or deprotonation sites

A pK_a atom should be the acidic hydrogen atom in an acid, or the basic atom in a base. You can use the conjugate acid or base as input, but you should chose the hydrogen for a conjugate acid such as an ammonium ion or the heavy atom for the conjugate base such as a carboxylate.

Note: Molecules in which the pK_a atom is involved an intramolecular hydrogen bond are likely to give erroneous results, so you should not run calculations on these molecules. You should consider using Epik for these systems.



Use structures from: Workspace (1 included entry)

(Note that jobs will run on all structures listed below.)

ID	In	Entry Title	Charge	Spin Mult.	pKa Atom
1	■	methylamine	0	1	Double-clic...

Symmetry: Use if present

☐ Perform conformational searches on input structures (requires MacroModel)

☐ Allow use of zwitterion functional groups

Job name: jag_methylamine_pka

Run

Jaguar: pKa, Host=localhost, Incorporate=Append new entries

Figure 13.1. The Input tab for pK_a calculations.

If you want to perform a conformational search on the protonated and the deprotonated species with MacroModel before the pK_a calculation is performed, select Perform conformational searches on input structures. The lowest-energy conformer is used for each species (method 1 above). This option requires a MacroModel license. For amino acid zwitterions, the conformational search is skipped, even if this option is selected (see Section 13.3.2 on page 359).

You can also disable the specific zwitterion parametrization by deselecting Allow use of zwitterion functional groups.

The other tabs available for pK_a calculations are the SCF tab, in which you can make settings to control the SCF convergence, and the Solvent tab, in which you can choose the solvent. If you want to calculate the pK_a for a CH acid or an NH acid in DMSO, for which parameters are available, you must choose DMSO as the solvent; otherwise the solvent should be water. These are the only solvent choices for pK_a calculations.

When you have finished setting up the job, click Run to start the job or click the Settings button to make job settings in the Job Settings dialog box, then click Run to submit the job. See Section 2.9 on page 25 for information on job settings. You can distribute a job for multiple molecules across multiple processors.

The pK_a values for each atom are added to the structure in the Maestro output file as values of an atom property (`r_j_pKa_water`). The output structure is the structure obtained from a geometry optimization of the input structure. The pK_a value for the first pK_a atom is also added to this structure as an entry property. When a pK_a job is incorporated, the property is listed in the Project Table, and the pK_a atoms are labeled with the pK_a values.

13.4.2 Jaguar Input Files for pK_a Calculations

You can prepare a Jaguar input file for a pK_a calculation using a text editor. The input file must contain a molecular geometry and a labeled pK_a atom. The pK_a atom is either an acidic hydrogen in acids or conjugate acids, or a non-hydrogen atom to be protonated in bases or conjugate bases.

The pK_a atom can be marked by setting the **gen** section keyword **ipkat** to either the atom's name, or to the atom's order number in the **zmat** section.

Here are two equivalent input file examples for formic acid:

```
&zmat
  C1   1.0590559100      0.0794463600      0.3608319800
  O2   0.8609619100      1.1054614700     -0.2390046100
  O3   2.2130316700     -0.6129886300      0.3489813100
  H1   2.8258867600     -0.1221771000     -0.2269021000
  H2   0.3281776900     -0.4358328800      1.0011835800
&
&gen
ipkat=H1
&

&zmat
  C1   1.0590559100      0.0794463600      0.3608319800
  O2   0.8609619100      1.1054614700     -0.2390046100
  O3   2.2130316700     -0.6129886300      0.3489813100
  H1   2.8258867600     -0.1221771000     -0.2269021000
  H2   0.3281776900     -0.4358328800      1.0011835800
&
&gen
ipkat=4
&
```

When the pK_a job is run, Jaguar checks that the functional group containing the designated pK_a atom has correction parameters available. If it does not, the job fails immediately, and an error message explaining the problem is printed to *jobname*`.out`.

If you want to calculate a raw pK_a value using the Jaguar methods but without correction factors, you can set `ipkaraw=1` in the **gen** section of the input file. You might want to do this when developing your own correction factors (see Section 13.5 on page 368).

You can generate multiple conformers for the protonated and deprotonated species, and calculate a macroscopic pK_a value making use of these conformers (method 2 in Section 13.2.1). To generate multiple conformers, set **ipka_max_conf** to the maximum number of conformers, and include the `-csrch` option when you run `jaguar pka`. You can also set an energy window (in kcal/mol) for filtering out conformers, with the **ipka_erg_window** keyword. Conformers are discarded if their energy relative to the lowest energy conformer is higher than the specified value. The default is 12.0 kcal/mol. If you want higher accuracy in the search, you can set **ipka_csrch_acc** to 1, to use a maximum of 3000 steps instead of 100 and 2500 minimization iterations instead of 500. You can also turn on solvation in the optimization of the conformers with Jaguar by setting **ipka_solv_opt** to 1. The default is gas-phase optimization, but use of solvation can help in some cases.

The macroscopic K_a value is given by

$$K_a = \sum_{j=1}^{N_d} d_j \left[\sum_{i=1}^{N_p} p_i / K_a(i,j) \right]^{-1} \qquad (2)$$

where N_d and N_p are the numbers of deprotonated and protonated conformers, d_j and p_i are the populations of these conformers, and $K_a(i,j)$ is the K_a value for protonated conformer i going to deprotonated conformer j. The populations are calculated with the following expressions:

$$1/p_i = \sum_{k=1}^{N_p} K_a(i,1)/K_a(k,1); \qquad 1/d_j = \sum_{k=1}^{N_p} K_a(1,j)/K_a(1,k)$$

Setting `ipkaverbose=1` produces more output, including intermediate p$K_a(i,j)$ values. When running the job, subjobs for each conformer can be distributed over multiple processors.

You can run pK_a calculations on multiple sites in the same molecule, by adding the **ipkasites** keyword to the **gen** section. The value for this keyword is a comma-separated list of atom numbers or atom labels, with no spaces, that identify the pK_a atoms. You can mix atom numbers and atom labels, e.g. `ipkasites=3,H5,O2,7`. The job is run as separate pK_a calculations with a single pK_a atom in each, and the results are collected at the end. The output structure has the pK_a values for each pK_a atom stored as an atom property (`r_j_pKa_water`), and the pK_a value for the first pK_a atom stored as an entry property (also named `r_j_pKa_water`).

13.4.3 Running pK_a Calculations from the Command Line

To submit a pK_a job for a single molecule using the command line, use the following command:

```
jaguar pka [options] [-csrch] {jobname.in|acid-and-base-files}
```

If the acid and base conformations are similar, you need only specify one input file, *jobname*`.in`, which can contain a structure for an acid or a base. The output Maestro file contains the optimized geometry of the input structure.

If the acid and base conformations are different, you can specify input files for both the acid and the base. If you do, you must give two filenames, in one of the following forms:

```
acidfile -deprot basefile
basefile -prot acidfile
-prot acidfile -deprot basefile
```

The order of the options can be inverted in the last form. The input file names *acidfile* and *basefile* must include the `.in` extension. In this description, "acid" means either the acid or the protonated base, and "base" means the base or the deprotonated acid. If you want to run a conformational search on both protonated and deprotonated species, using MacroModel, include the `-csrch` option. The lowest-energy conformer is taken for each species by default, but you can calculate a macroscopic pKa with all the conformers by setting keywords (see Section 13.4.2). The output Maestro file contains the optimized geometry of the structure specified with `-prot`.

The structures that are specified in these files must have the same protonation state. Thus, if you specify an acid in *acidfile*, the file *basefile* must also contain a structure for the acid, but in the conformation that is appropriate to the conjugate base. Jaguar automatically changes the protonation state and adjusts the charge when it uses these structures. The same rules apply to the structure as for a single input file: if you are doing calculations for an acid, both structures must be protonated, and if you are doing calculations for a base, both structures must be deprotonated. So, for example, if you performed a conformational search with MacroModel on an amine and its corresponding ammonium ion, you would have to remove a proton from the ammonium ion structure before putting it in the *acidfile* input file.

If you have many pK_a jobs to run, and if you have enough licenses, you can distribute multiple jobs across multiple processors with the following command:

```
jaguar pka *.in -HOST hostname:njobs
```

where *njobs* is the number of jobs that you want to run simultaneously (this requires the same number of Jaguar and Jaguar pKa licenses).

Each pKa calculation runs 5 subjobs on each of the acid and base species (10 subjobs total). Most of these subjobs can use OpenMP or OpenMPI if you have extra processors available, so for example you can add `-TPP` *nthreads* to the above command to run each subjob with *nthreads* threads for an additional speedup. If you specify multiple processors, the first two are used to run the acid and base species on separate processors, then the rest are used for parallel or threaded execution. It is therefore a good idea to specify an even number of processors.

13.4.4 Monitoring pK_a Calculations

The pK_a calculations can be monitored from the Maestro Monitor panel or by looking at the file `pka.*.blog` (where `*` is a process identification number).

For each molecule Jaguar creates a *jobname*`_pka` subdirectory in the local directory and writes the input and output for each job step there. The input and output filenames have suffixes appended to *jobname* that explain what is calculated in each step. These suffixes are listed in Table 13.3.

Table 13.3. File suffixes for pKa calculations

Suffix	Job Step Explanation
`dft_h`	B3LYP/6-31G* geometry optimization for conjugate acid
`nrg_h`	B3LYP/cc-pVTZ(-f)(+) single point energy for conjugate acid
`solv_h`	B3LYP/6-31G**(+) single point solution phase calculation for conjugate acid
`pr#_h`	input file preparation runs for conjugate acid
`dft`	B3LYP/6-31G* geometry optimization for conjugate base
`nrg`	B3LYP/cc-pVTZ(-f)(+) single point energy for conjugate base
`solv`	B3LYP/6-31G**(+) single point solution phase calculation for conjugate base
`pr#`	input file preparation runs for conjugate base

The energies of the gas-phase optimized structures and the solvation energies for the protonated and deprotonated forms are extracted from the output files and written to a file named *jobname*`.jres` in the *jobname*`_pka` directory. The pK_a value is then calculated from the data in this file and is written to the output file, *jobname*`.out`. For example, here is the final output file for methylamine:

```
This molecule was recognized as a primary amine.
Correction factors have been applied, and have an RMSD of 0.5.
The pKa for 'methylamine' is: 10.3
```

For a molecule in which the shell model was used, the following output is generated (in this case for an imide of the type R-(C=O)-NH-C(=O)-R).

```
#                     Shell                          pKa    RMSD
---    -----------------------------------      -----   -----
```

```
1:      imide                                          7.0   0.699
2:      Protonation of sp2-like aromatic nitrogen      8.1   0.968
3:      Protonation of generic atom                    7.0   2.120
```

Even though protonation is mentioned, this is for the conjugate base, so the overall process is the deprotonation of the imide nitrogen.

13.4.5 Choice of Initial Geometry

It is critical to choose the lowest energy conformer for the pK_a calculations. If you have MacroModel, we recommend you use it to perform a conformational search on the protonated and deprotonated forms of your molecule. The Jaguar distribution comes with a Python script that automatically runs the conformational searches on both forms of the molecule and then runs the pK_a calculation. To use this script from Maestro, select Perform conformational searches on input structures in the Molecule tab of the Jaguar panel. The input molecule is automatically protonated or deprotonated. Both species then undergo a conformational search using mixed torsional/low-mode sampling with the OPLS_2005 force field and water as the solvent. The lowest-energy structure for each species is then used in a subsequent Jaguar pK_a calculation.

To run pK_a jobs from the command line, use this command to run the tasks described above:

```
$SCHRODINGER/jaguar pka -csrch input_file
```

Note that you only need to provide a single input structure. As long as you have the pK_a atom marked correctly, the script protonates or deprotonates the structure to form the conjugate species as appropriate.

If your input file is called `thcd24.in`, for example, the conformational searches are performed in a subdirectory of your working directory named `thcd24_csrch`. The new input files containing the best conformers are named `thcd24_prot.in` and `thcd24_deprot.in`, and are written to your working directory. The pK_a job directory is always named after the protonated species, so it is called `thcd24_prot_pka` in this case. The output file containing the calculated pK_a is named `thcd24_prot.out`. Your original input file is not overwritten.

When setting up a pK_a job for a zwitterion, make sure that the structure is set up such that a zwitterion species is present during the calculation. For example, if you are setting up an amino acid as base calculation with the amine group to be protonated, make sure that the carboxylic acid group is deprotonated with a charge of –1 on the oxygen, so that protonation of the amine results in a zwitterion species. You should not, however, create formal charges on a structure for which there is a resonance structure with no formal charges, such as a lactam. If you do, the formal charges and bond orders are reassigned to remove the formal charges before identifying the functional group.

13.4.6 Recalculating pK_a Values with New Parameters

The utility script `jaguar_pka` can be used to recalculate pK_a values using parameters for a specified functional group. The calculation takes the raw energies from a pK_a calculation and applies the correction factors for the specified functional group. The syntax of the command is

```
$SCHRODINGER/utilities/jaguar_pka [options] [jobname_pka.zip/]jobname.in
```

The optional zip file prefix can be used to indicate that the input file *jobname*`.in` is in the specified zip file, which will be unzipped temporarily (provided the unzipped directory does not exist; if it does, the command fails). This allows you to use the zipped results of a pK_a job for refitting, without unzipping it first. The options are listed in Table 13.4.

Table 13.4. Options for the `jaguar_pka` *script.*

Option	Description
`-i`	Run in identification mode, to identify the current functional group.
`-f` *name*	Specify the name of the functional group whose correction factors are to be used. The name of the functional group must be specified in lower case, and must be one of the functional group names listed in the `pka_match.xml` file. The standard names are the same as those in Table 13.1 except that they are always expressed in the singular, not plural, and given in lower case.
`-j` *filename*	Specify an alternate `.jres` file. Default is *jobname*`.jres`)
`-m` *filename*	Specify an alternate `pka_match.xml` file. Default is `$SCHRODINGER/mmshare-v`*version*`/data/jaguar/pka_match.xml`.
`-o` *filename*	Specify a Maestro output file.
`-s` *solvent*	Specify a solvent. The solvent must be one for which pK_a parameters are available.

As an example, if Jaguar has calculated the pK_a for an aniline, and you want to see what the pK_a would be if it had been corrected using the parameters for primary amines, you could change into the *jobname*`_pka` directory and run the following command:

```
    $SCHRODINGER/utilities/jaguar_pka -f "primary amine" jobname.in
```

The output looks something like this:

```
This molecule was specified by the user as a primary amine.
Correction factors have been applied, and have an RMSD of 0.5.
The pKa for '2clphnh2' is: 5.7
```

13.5 Developing Your Own pK_a Correction Parameters

If you want to develop your own pK_a parameters, you can do so by adding information to the file that contains the parameters that Jaguar uses to correct its calculated pK_a values, `$SCHRODINGER/mmshare-v`*version*`/data/jaguar/pka_match.xml`. This file also contains the SMARTS patterns that enable Jaguar to recognize functional groups. You can thus extend Jaguar's ability to calculate pK_a values for new functional groups simply by adding the appropriate SMARTS patterns and correction parameters to this file.

A description of the XML file format standard is beyond the scope of this document, but the format is very simple and resembles HTML in its use of tags to enclose sections of ordinary text. The tags identify the purpose of the enclosed text. For example, the pK_a module information for carboxylic acids looks like this:

```
<functional_group name="carboxylic acid" jaguar_id="4">
        <jaguar f1="0.4451" f2="-0.2516"/>
        <smarts>
         [#1]O[CX3]=O
        </smarts>
        <smarts>
         [OX1-][CX3]=O
        </smarts>
    </functional_group>
```

where `name` is a double-quoted string that describes the functional group, `jaguar_id` is an optional arbitrary index number for the functional group, and `f1` and `f2` are the pK_a correction factors. The first SMARTS pattern describes the acidic form of the molecule, while the second SMARTS pattern describes the basic form of the molecule. For more information on SMARTS patterns, see the web page http://www.daylight.com/dayhtml/doc/theory/theory.smarts.html.

The two pK_a correction factors, `f1` and `f2`, come from a linear fit of the calculated pK_a values to the experimental values for a particular training set of molecules. `f1` is the slope and `f2` is the intercept. You can perform linear fits with many commonly available software programs.

Here are some suggestions for selecting a set of molecules to use as a training set for the development of new pK_a correction parameters:

- All of the experimental pK_a values must be in the same solvent at the same temperature, plus or minus a few degrees. pK_a values in mixed solvents are not a good choice. This is because the continuum solvation model used by Jaguar requires the specification of a single solvent dielectric constant and probe radius, and it is not known how these parameters should be specified for a mixed solvent system, especially when the degree of preferential solvation of the solute is unknown.

- Select molecules for which the experimental pK_a values are accurately known. Aqueous pK_a values near 14 and beyond, or near 0 and beyond, are not generally accurate because of the difficulty in measuring the concentration of acid or base in the presence of high concentrations of hydronium or hydroxide (the leveling effect).
- Try to obtain experimental pK_a values that cover as wide a pK_a range as possible for the given functional group. This ensures that the fitting parameters are broadly applicable to molecules containing that functional group.
- The more molecules you use in the training set, the more clearly you can see how well the calculated pK_a correlates with the experimental pK_a, and the better idea you will have of the RMS error.
- Do not select training set molecules that contain symmetrically equivalent functional groups. An additional correction term is required in this case to account for the increased entropy when degenerate sites are present. This correction can be applied manually, as needed, *after* the `f1` and `f2` correction factors have been automatically applied by Jaguar—see Section 13.2.2 on page 352.

Getting Help

Information about Schrödinger software is available in two main places:

- The `docs` folder (directory) of your software installation, which contains HTML and PDF documentation. Index pages are available in this folder.
- The Schrödinger web site, http://www.schrodinger.com/, In particular, you can use the Knowledge Base, http://www.schrodinger.com/kb, to find current information on a range of topics, and the Known Issues page, http://www.schrodinger.com/knownissues, to find information on software issues.

Finding Information in Maestro

Maestro provides access to nearly all the information available on Schrödinger software.

To get information:

- Pause the pointer over a GUI feature (button, menu item, menu, ...). In the main window, information is displayed in the Auto-Help text box, which is located at the foot of the main window, or in a tooltip. In other panels, information is displayed in a tooltip.

  If the tooltip does not appear within a second, check that Show tooltips is selected under General → Appearance in the Preferences panel, which you can open with CTRL+, (⌘,). Not all features have tooltips.
- Click the Help button in the lower right corner of a panel or press F1, for information about a panel or the tab that is displayed in a panel. The help topic is displayed in the Help panel. The button may have text or an icon:

  ?
- Choose Help → Online Help or press CTRL+H (⌘H) to open the default help topic.
- When help is displayed in the Help panel, use the navigation links in the help topic or search the help.
- Choose Help → Documentation Index, to open a page that has links to all the documents. Click a link to open the document.

- Choose Help → Search Manuals to search the manuals. The search tab in Adobe Reader opens, and you can search across all the PDF documents. You must have Adobe Reader installed to use this feature.

For information on:

- Problems and solutions: choose Help → Knowledge Base or Help → Known Issues → *product*.
- New software features: choose Help → New Features.
- Python scripting: choose Help → Python Module Overview.
- Utility programs: choose Help → About Utilities.
- Keyboard shortcuts: choose Help → Keyboard Shortcuts.
- Installation and licensing: see the *Installation Guide*.
- Running and managing jobs: see the *Job Control Guide*.
- Using Maestro: see the *Maestro User Manual*.
- Maestro commands: see the *Maestro Command Reference Manual*.

Contacting Technical Support

If you have questions that are not answered from any of the above sources, contact Schrödinger using the information below.

Web:	http://www.schrodinger.com/supportcenter
E-mail:	help@schrodinger.com
Mail:	Schrödinger, 101 SW Main Street, Suite 1300, Portland, OR 97204
Phone:	+1 888 891-4701 (USA, 8am – 8pm Eastern Time)
	+49 621 438-55173 (Europe, 9am – 5pm Central European Time)
Fax:	+1 503 299-4532 (USA, Portland office)
FTP:	ftp://ftp.schrodinger.com

Generally, using the web form is best because you can add machine output and upload files, if necessary. You will need to include the following information:

- All relevant user input and machine output
- Jaguar purchaser (company, research institution, or individual)
- Primary Jaguar user
- Installation, licensing, and machine information as described below.

Gathering Information for Technical Support

The instructions below describe how to gather the required machine, licensing, and installation information, and any other job-related or failure-related information, to send to technical support. Where the instructions depend on the profile used for Maestro, the profile is indicated.

For general enquiries or problems:

1. Open the Diagnostics panel.
   - **Maestro:** Help → Diagnostics
   - **Windows:** Start → All Programs → Schrodinger-2015-4 → Diagnostics
   - **Mac:** Applications → Schrodinger2015-4 → Diagnostics
   - **Command line:** `$SCHRODINGER/diagnostics`
2. When the diagnostics have run, click Technical Support.

   A dialog box opens, with instructions. You can highlight and copy the name of the file.
3. Upload the file specified in the dialog box to the support web form.

   If you have already submitted a support request, use the upload link in the email response from Schrödinger to upload the file. If you need to submit a new request, you can upload the file when you fill in the form.

If your job failed:

1. Open the Monitor panel, using the instructions for your profile as given below:
   - **Maestro/Jaguar/Elements:** Tasks → Monitor Jobs
   - **BioLuminate/MaterialsScience:** Tasks → Job Monitor
2. Select the failed job in the table, and click Postmortem.

   The Postmortem panel opens.
3. If your data is not sensitive and you can send it, select Include structures and deselect Automatically obfuscate path names.
4. Click Create.

   An archive file is created, and an information dialog box with the name and location of the file opens. You can highlight and copy the name of the file.
5. Upload the file specified in the dialog box to the support web form.

   If you have already submitted a support request, use the upload link in the email response from Schrödinger to upload the file. If you need to submit a new request, you can upload the file when you fill in the form.

6. Copy and paste any log messages from the window used to start the interface or the job into the web form (or an e-mail message), or attach them as a file.

   - **Windows:** Right-click in the window and choose Select All, then press ENTER to copy the text.
   - **Mac:** Start the Console application (Applications → Utilities), filter on the application that you used to start the job (Maestro, BioLuminate, Elements), copy the text.

If Maestro failed:

1. Open the Diagnostics panel.
   - **Windows:** Start → All Programs → Schrodinger-2015-4 → Diagnostics
   - **Mac:** Applications → SchrodingerSuite2015-4 → Diagnostics
   - **Linux/command line:** `$SCHRODINGER/diagnostics`
2. When the diagnostics have run, click Technical Support.

   A dialog box opens, with instructions. You can highlight and copy the name of the file.
3. Upload the file specified in the dialog box to the support web form.

   If you have already submitted a support request, use the upload link in the email response from Schrödinger to upload the file. If you need to submit a new request, you can upload the file when you fill in the form.
4. Upload the error files to the support web form.

   The files should be in the following location:

- **Windows:** `%LOCALAPPDATA%\Schrodinger\appcrash`
  (Choose Start → Run and paste this location into the Open text box.)
  Attach `maestro_error`*pid*`.txt` and `maestro.exe`*pid_timestamp*`.dmp`.
- **Mac:** `$HOME/Library/Logs/CrashReporter`
  (Go → Home → Library → Logs → CrashReporter)
  Attach `maestro_error`*pid*`.txt` and `maestro`*_timestamp_machinename*`.crash`.
- **Linux:** `$HOME/.schrodinger/appcrash`
  Attach `maestro_error`*pid*`.txt` and `crash_report_`*timestamp_pid*`.txt`.

If a Maestro panel failed to open:

1. Copy the text in the dialog box that opens.
2. Paste the text into the support web form.

References

The first 18 references listed below provide general information about the algorithms used in Jaguar and some of their applications. Their titles are included in the listings. The other listings in this section are referenced throughout this manual.

1. Friesner, R. A. Solution of Self-Consistent Field Electronic Structure Equations by a Pseudospectral Method. *Chem. Phys. Lett.* **1985,** *116,* 39.

2. Friesner, R. A. Solution of the Hartree-Fock equations by a pseudospectral method: Application to diatomic molecules. *J. Chem. Phys.* **1986,** *85,* 1462.

3. Friesner, R. A. Solution of the Hartree-Fock equations for polyatomic molecules by a pseudospectral method. *J. Chem. Phys.* **1987,** *86,* 3522.

4. Friesner, R. A. An Automatic Grid Generation Scheme for Pseudospectral Self-Consistent Field Calculations on Polyatomic Molecules. *J. Phys. Chem.* **1988,** *92,* 3091.

5. Ringnalda, M. N.; Won, Y.; Friesner, R. A. Pseudospectral Hartree-Fock calculations on glycine. *J. Chem. Phys.* **1990,** *92,* 1163.

6. Langlois, J. -M.; Muller, R. P.; Coley, T. R.; Goddard, W. A., III; Ringnalda, M. N.; Won, Y.; Friesner, R. A. Pseudospectral generalized valence-bond calculations: Application to methylene, ethylene, and silylene. *J. Chem. Phys.* **1990,** *92,* 7488.

7. Ringnalda, M. N.; Belhadj, M.; Friesner, R. A. Pseudospectral Hartree-Fock theory: Applications and algorithmic improvements. *J. Chem. Phys.* **1990,** *93,* 3397.

8. Won, Y.; Lee, J. -G.; Ringnalda, M. N.; Friesner, R. A. Pseudospectral Hartree-Fock gradient calculations. *J. Chem. Phys.* **1991,** *94,* 8152.

9. Friesner, R. A. New Methods for Electronic Structure Calculations on Large Molecules. *Ann. Rev. Phys. Chem.* **1991,** *42,* 341.

10. Pollard, W. T.; Friesner, R. A. Efficient Fock matrix diagonalization by a Krylov-space method. *J. Chem. Phys.* **1993,** *99,* 6742.

11. Muller, R. P.; Langlois, J. -M.; Ringnalda, M. N.; Friesner, R. A.; Goddard, W. A., III. A generalized direct inversion in the iterative subspace approach for generalized valence bond wave functions. *J. Chem. Phys.* **1994,** *100,* 1226.

12. Murphy, R. B.; Friesner, R. A.; Ringnalda, M. N.; Goddard, W. A., III. Pseudospectral Contracted Configuration Interaction From a Generalized Valence Bond Reference. *J. Chem. Phys.* **1994,** *101,* 2986.

13. Greeley, B. H.; Russo, T. V.; Mainz, D. T.; Friesner, R. A.; Langlois, J. -M.; Goddard, W. A., III; Donnelly, R. E., Jr., Ringnalda, M. N. New Pseudospectral Algorithms for Electronic Structure Calculations: Length Scale Separation and Analytical Two-Electron Integral Corrections. *J. Chem. Phys.* **1994,** *101,* 4028.

14. Langlois, J. -M.; Yamasaki, T.; Muller, R. P.; Goddard, W. A. Rule Based Trial wave functions for Generalized Valence Bond Theory. *J. Phys. Chem.* **1994,** *98,* 13498.

15. Tannor, D. J.; Marten, B.; Murphy, R.; Friesner, R. A.; Sitkoff, D.; Nicholls, A.; Ringnalda, M.; Goddard, W. A., III; Honig, B. Accurate First Principles Calculation of Molecular Charge Distributions and Solvation Energies from Ab Initio Quantum Mechanics and Continuum Dielectric Theory. *J. Am. Chem. Soc.* **1994,** *116,* 11875.

16. Murphy, R. B.; Beachy, M. D.; Friesner, R. A.; Ringnalda, M. N. Pseudospectral Localized MP2 Methods: Theory and Calculation of Conformational Energies. *J. Chem. Phys.* **1995,** *103,* 1481.

17. Lu, D.; Marten, B.; Cao, Y.; Ringnalda, M. N.; Friesner, R. A.; Goddard, W. A., III. *Ab initio* Predictions of Large Hyperpolarizability Push-Pull Polymers: Julolidinyl-n-isoxazolone and Julolidinyl-n-*N,N'*-diethylthiobarbituric acid. *Chem. Phys. Lett.* **1995,** *242,* 543.

18. Cao, Y.; Friesner, R. A. Molecular (hyper)polarizabilities computed by pseudospectral methods. *J. Chem. Phys.* **2005**, *122*, 104102.

19. Cao, Y.; Beachy, M. D.; Braden, D. A.; Morrill, L. A.; Ringnalda, M. N.; Friesner, R. A. Nuclear-magnetic-resonance shielding constants calculated by pseudospectral methods. J. Chem. Phys. **2005**, *122*, 224116.

20. Murphy, R. B.; Pollard, W. T.; Friesner, R. A. Pseudospectral localized generalized Møller-Plesset methods with a generalized valence bond reference wave function: Theory and calculation of conformational energies. *J. Chem. Phys.* **1997,** *106,* 5073.

21. Vacek, G.; Perry, J. K.; Langlois, J. -M. *Chem. Phys. Lett.* **1999,** *310,* 189.

22. Bobrowicz F. W.; Goddard, W. A., III. Chapter 4. In *Modern Theoretical Chemistry: Methods of Electronic Structure Theory;* Schaefer, H. F., III, Ed., 3; Plenum: New York, 1977.

23. BIOGRAF manual.

24. MacroModel manual.

25. Frisch, M. J.; Trucks, G. W.; Head-Gordon, M.; Gill, P. M. W.; Wong, M. W.; Foresman, J. B.; Johnson, B. G.; Schlegel, H. B.; Robb, M. A.; Replogle, E. S.; Gomperts, R.; Andres, J. L.; Raghavachari, K.; Binkley, J. S.; Gonzalez, C.; Martin, R. L.; Fox, D. J.; DeFrees, D. J.; Baker, J.; Stewart, J. J. P.; Pople, J. A. GAUSSIAN 92. Gaussian, Inc.: Pittsburgh, PA, 1992.

26. Babel version 1.6, copyright © 1992-96 W. Patrick Walters and Matthew T. Stahl, All Rights Reserved. (Permission of authors granted to incorporate Babel into Jaguar.)

27. Dunietz, B. D.; Murphy, R. B.; Friesner, R. A. Calculation of enthalpies of formation by a multi-configurational localized perturbation theory - application for closed shell cases. *J. Chem. Phys.* **1999,** *110,* 1921.

28. Kaminski, G. A.; Maple, J. R.; Murphy, R. B.; Braden, D. A.; Friesner, R. A. *J. Chem. Theory Comput.* **2005**, *1*, 248.

29. Boys, S. F.; Bernardi, F. *Mol. Phys.* **1970**, *19*, 553.

30. Hirata, S.; Head-Gordon, M. *Chem. Phys. Lett.* **1999,** *314*, 291.

31. Becke, A. D. *J. Chem. Phys.* **1993,** *98,* 1372.

32. Becke, A. D. *J. Chem. Phys.* **1993,** *98,* 5648.

33. Stephens, P. J.; Devlin, F. J.; Chabalowski, C. F.; Frisch, M. J. *J. Phys. Chem.* **1994,** *98,* 11623.

34. Slater, J. C. *Quantum Theory of Molecules and Solids, Vol. 4: The Self-Consistent Field for Molecules and Solids*. McGraw-Hill: New York, 1974.

35. Vosko, S. H.; Wilk, L.; Nusair, M. *Can. J. Phys.* **1980,** *58,* 1200.
(The VWN correlation functional is described in the paragraph below equation [4.4] on p. 1207, while the VWN5 functional is described in the caption of Table 5 and on p. 1209.)

36. Perdew, J. P. In *Electronic Structure Theory of Solids;* Ziesche, P., Eschrig, H., Eds.; Akademie Verlag: Berlin, 1991. Perdew, J. P.; Chevary, J. A.; Vosko, S. H.; Jackson, K. A.; Pederson, M. R.; Singh, D. J.; Fiolhais, C. *Phys. Rev. B* **1992,** *46,* 6671.

37. Becke, A. D. *Phys. Rev. A* **1988,** *38,* 3098.

38. Lee, C.; Yang, W.; Parr, R. G. *Phys. Rev. B* **1988,** *37,* 785; implemented as described in Miehlich, B.; Savin, A.; Stoll, H.; Preuss, H. *Chem. Phys. Lett.* **1989,** *157,* 200.

39. Perdew, J. P.; Zunger, A. *Phys. Rev. B* **1981,** *23,* 5048.

40. Perdew, J. P. *Phys. Rev. B* **1986,** *33,* 8822; and Perdew, J. P. *Phys. Rev. B* (Erratum) **1986,** *34,* 7406.

41. Becke, A. D. *J. Chem. Phys.* **1996,** *104,* 1040.

42. Becke, A. D. *J. Chem. Phys.* **1997,** *107,* 8554.

43. Becke, A. D. *J. Chem. Phys.* **1998,** *109,* 2092.

44. Schmider, H. L.; Becke, A. *J. Chem. Phys.* **1998,** *109,* 8188; Schmider, H. L.; Becke, A. *J. Chem. Phys.* **1998,** *108,* 9624.

45. Hamprecht, F. A.; Cohen, A. J.; Tozer, D. J.; Handy, N. C. *J. Chem. Phys.* **1998,** *109,* 6264.

46. Boese, A. D.; Handy, N. C. *J. Chem. Phys.* **2001,** *114,* 5497.

47. Perdew, J. P.; Burke, K.; Ernzerhof, M. *Phys. Rev. Lett.* **1996,** *77,* 3865; *Phys. Rev. Lett.* (Erratum) **1997,** *78,* 1386.

48. Handy, N. C.; Cohen, A. J. *Mol. Phys.* **2001,** *99*, 403.

49. Grimme, S. *J. Comp. Chem.* **2006**, *27*, 1787.

50. Grimme, S.; Antony, J.; Ehrlich, S.; Krieg, H. *J. Chem. Phys.* **2010**, *132*, 154104.

51. Goerigk, L.; Grimme, S. P*hys. Chem. Chem. Phys.* **2011**, *13*, 6670.

52. Adamo, C.; Barone, V. *J. Chem. Phys.* **1998,** *108*, 664.

53. Adamo, C.; Barone, V. *J. Chem. Phys.* **1999,** *110*, 6158.

54. Lynch, B. J.; Fast, P. L.; Harris, M.; Truhlar, D. G. *J. Phys. Chem. A* **2000**, *104*, 4811.

55. Xu, X; Goddard, W. A., III. *Proc. Natl. Acad. Sci.* U.S.A. **2004**, *101*, 2673.

56. Zhao, Y.; Lynch, B. J.; Truhlar, D. G. *J. Phys. Chem. A* **2004**, *108*, 2715.

57. Zhao, Y.; Truhlar, D. G. *J. Phys. Chem. A* **2005**, *109*, 5656.

58. Zhao, Y.; Schultz, N. E.; Truhlar, D. G. *J. Chem. Phys.* **2005**, *123*, 161103.

59. Zhao, Y.; Schultz, N. E.; Truhlar, D. G. *J. Chem. Theory Comput.* **2006**, *2*, 364.

60. Zhao, Y.; Truhlar, D. G. *J. Chem. Phys.* **2006**, *125*, 194101.

61. Zhao, Y.; Truhlar, D. G. *J. Phys. Chem. A* **2006**, *110*, 13126.

62. Zhao, Y.; Truhlar, D. G. *Theor. Chem. Acc.* **2008**, *120*, 215.

63. Zhao, Y.; Truhlar, D. G. Exploring the Limit of Accuracy of the Global Hybrid Meta Density Functional for Main-Group Thermochemistry, Kinetics, and Noncovalent Interactions *J. Chem. Theory Comput.* **2008**, *4*, 1849.

64. Iikura, H.; Tsuneda, T.; Yanai, T.; Hirao, K. A long-range correction scheme for generalized-gradient-approximation exchange functionals. *J. Chem. Phys.* **2001**, *115*, 3540.

65. Yanai, T.; Tew, D. P.; Handy, N. C. A new hybrid exchange–correlation functional using the Coulomb-attenuating method (CAM-B3LYP). *Chem. Phys. Lett.* **2004**, *393*, 51.

66. Henderson, T. M.; Janesko, B. G.; Scuseria, G. E. Generalized gradient approximation model exchange holes for range-separated hybrids. *J. Chem. Phys.* **2008**, *128*, 194105.

67. Vydrov O. A.; Scuseria, G. E. Assessment of a long-range corrected hybrid functional. *J. Chem. Phys.* **2006**, *125*, 234109.

68. Rohrdanz, M. A.; Martins, K. M.; Herbert, J. M. A long-range-corrected density functional that performs well for both ground-state properties and time-dependent density functional theory excitation energies, including charge-transfer excited states. *J. Chem. Phys.* **2009**, *130*, 054112.

69. Heyd, J.; Scuseria, G. E.; Ernzerhof, M. Hybrid functionals based on a screened Coulomb potential. *J. Chem. Phys.* **2003**, *118*, 8207.

70. Heyd, J.; Scuseria, G. E.; Ernzerhof, M. Erratum “Hybrid functionals based on a screened Coulomb potential” [J. Chem. Phys.118, 8207 (2003)]. *J. Chem. Phys.* **2006**, *124*, 219906.

71. Krukau, A. V.; Vydrov O. A.; Izmaylov A. F.; Scuseria, G. E. Influence of the exchange screening parameter on the performance of screened hybrid functionals. *J. Chem. Phys.* **2006**, *125*, 224106.

72. Peverati, R.; Truhlar, D.G. Improving the Accuracy of Hybrid Meta-GGA Density Functionals by Range Separation. *J. Phys. Chem. Lett.* **2011**, *2*, 2810.

73. Peverati, R.; Truhlar, D.G. M11-L: A Local Density Functional That Provides Improved Accuracy for Electronic Structure Calculations in Chemistry and Physics. *J. Phys. Chem. Lett.* **2012**, *3*, 117.

74. Chai, J.-D.; Head-Gordon M. Systematic optimization of long-range corrected hybrid density functionals. *J. Chem. Phys.* **2008**, *128*, 084106.

75. Baer, R.; Neuhauser, D. Density functional theory with correct long-range asymptotic behavior. *Phys. Rev. Lett.* **2005**, *94*, 043002.

76. Livshits, E.; Baer, R. A well-tempered density functional theory of electrons in molecules. *Phys.Chem.Chem.Phys.* **2007**, *9*, 2932.

77. Schneebeli, S. T.; Bochevarov, A. D.; Friesner R. A. *J. Chem. Theory Comput.* **2011**, *7* 658.

78. Friesner, R. A.; Knoll, E. H.; Cao Y. *J. Chem. Phys.* **2006**, *125*, 124107.

79. Friesner, R. A.; Knoll, E. H. *J. Phys. Chem. B* **2006**, *110*, 18787.

80. Goldfeld, D. A.; Bochevarov, A. D.; Friesner R. A. *J. Chem. Phys.* **2008**, *129*, 214105.

81. Hall, M. L.; Goldfeld, D. A.; Bochevarov, A. D.; Friesner R. A. *J. Chem. Theory Comput.* **2009**, *5*, 2996.

82. Hall, M. L.; Zhang, J.; Bochevarov, A. D.; Friesner R. A. *J. Chem. Theory Comput.* **2010**, *6*, 3647.

83. Rinaldo, D.; Tian,L.; Harvey, J. H.; Friesner R. A. *J. Chem. Phys.* **2008**, *129*, 164108.

84. Hughes, T. F.; Friesner R. A. *J. Chem. Theory Comput.* **2011**, *7*, 19.

85. Hughes, T. F.; Friesner R. A. *J. Chem. Theory Comput.* **2012**, *8*, 442.

86. Hughes, T. F.; Harvey, J. H.; Friesner R. A. *Phys. Chem. Chem. Phys.* **2012**, *14*, 7724.

87. Kim, H.; Choi, J.-M.; Goddard, W. A., III *J. Phys. Chem. Lett.* **2012**, *3*, 360.

88. Møller, C.; Plesset, M. S. *Phys. Rev.* **1934,** *46,* 618.

89. Pulay, P.; Saebø, S. *Theor. Chim. Acta* **1986,** *69,* 357.

90. Saebø, S.; Pulay, P. *Ann. Rev. Phys. Chem.* **1993,** *44,* 213.

91. Saebø, S.; Tong, W.; Pulay, P. *J. Chem. Phys.* **1993,** *98,* 2170.

92. Foster, J. M.; Boys, S. F. *Rev. Mod. Phys.* **1960,** *32,* 300.

93. Pipek, J.; Mezey, P. G. *J. Chem. Phys.* **1989,** *90,* 4916.

94. Harding, L. B.; Goddard, W. A., III. *J. Am. Chem. Soc.* **1975,** *97,* 6293.

95. Carter, E. A.; Goddard, W. A., III. *J. Chem. Phys.* **1987,** *86,* 862.

96. Fischer, T. H.; Almlöf, J. *J. Phys. Chem.* **1992,** *96,* 9768.

97. Schlegel, H. B. *Theor. Chim. Acta* **1984,** *66,* 333.

98. Fogarasi, G.; Zhou, X.; Taylor, P. W.; Pulay, P. *J. Am. Chem. Soc.* **1992,** *114*, 8191.

99. *CRC Handbook of Chemistry and Physics;* Weast, R. C., Ed.; 60th edition; CRC Press: Boca Raton, FL, 1979. Dielectric constants for 20 deg. C were used.

100. Water's probe radius is set to 1.40 to reproduce solvation energies properly. All other probe radii are calculated from $r^3 = (3m\Delta)/(4\pi\rho)$ (10^{24} A^3/cm^3), where *r* is the solvent probe radius in Angstroms, *m* is the molecular mass obtained by dividing the molecular weight given in ref. [99] in grams per mole by 6.02 x 10^{23}, Δ is the packing density, and ρ is the density in g/cm^3 at 20 deg. C obtained from ref. [99]. Finding the actual Δ would require a detailed knowledge of the structure of the liquid. Currently, all Δ values for

these liquids are assumed to be 0.5. (For FCC lattices, Δ is 0.7405, and for BCC lattices, Δ is 0.6802.)

101. Rappé, A. K.; Casewit, C. J.; Colwell, K. S.; Goddard, W. A.; Skiff, W. M. *J. Am. Chem. Soc.* **1992,** *114,* 10024.

102. Chirlian, L. E.; Francl, M. M. *J. Comput. Chem.* **1987,** *8,* 894; Woods, R. J.; Khalil, M.; Pell, W.; Moffat, S. H.; Smith, V. H., Jr. *J. Comput. Chem.* **1990,** *11,* 297.

103. Breneman, C. M.; Wiberg, K. B. *J. Comput. Chem.* **1990,** *11,* 361.

104. Mayo, S. L.; Olafson, B. D.; Goddard, W. A., III. *J. Phys. Chem.* **1990,** *94,* 8897.

105. Mulliken, R. S. *J. Chem. Phys.* **1955,** *23,* 1833.

106. Glendening, E. D.; Badenhoop, J. K.; Reed, A. E.; Carpenter, J. E.; Bohmann, J. A.; Morales, C. M.; Landis, C. R.; Weinhold, F. *NBO 6.0*; Theoretical Chemistry Institute: University of Wisconsin, Madison, WI, 2013, http://nbo6.chem.wisc.edu/.

107. Baker, J.; Jarzecki, A. A.; Pulay, P. *J. Phys. Chem A* **1998,** *102,* 1412.

108. Scott, A. P.; Radom, L. *J. Phys.Chem.* **1996,** *100,* 16502.

109. Alecu, I. M.; Zheng, J.; Zhao, Y.; Truhlar, D. G. *J. Chem. Theory Comput.* **2010**, *6*, 2872.

110. Sjoberg, P.; Murray, J. S.; Brinck, T.; Politzer, P. *Can. J. Chem.* **1990,** *68*, 1440.

111. Murray, J. S.; Abu-Awwad, F.; Politzer, P. *J. Mol. Struct. (Theochem)* **2000,** *241*, 501.

112. Hehre, W. J.; Stewart, R. F.; Pople, J. A. *J. Chem. Phys.* **1969,** *51,* 2657.

113. Hehre, W. J.; Ditchfield, R.; Stewart, R. F.; Pople, J. A. *J. Chem. Phys.* **1970,** *52,* 2769.

114. Pietro, W. J.; Levi, B. A.; Hehre, W. J.; Stewart, R. F. *Inorg. Chem.* **1980,** *19,* 2225.

115. Pietro, W. J.; Blurock, E. S.; Hout, R. F., Jr.; Hehre, W. J.; DeFrees, D. J.; Stewart, R. F. *Inorg. Chem.* **1980,** *20,* 3650.

116. Collins, J. B.; Schleyer, P. von R.; Binkley, J. S.; Pople, J. A. *J. Chem. Phys.* **1976,** *64,* 5142.

117. Binkley, J. S.; Pople, J. A.; Hehre, W. J. *J. Am. Chem. Soc.* **1980,** *102,* 939.

118. Gordon, M. S.; Binkley, J. S.; Pople, J. A.; Pietro, W. J.; Hehre, W. J. *J. Am. Chem. Soc.* **1982,** *104,* 2797.

119. Pietro, W. J.; Francl, M. M.; Hehre, W. J.; DeFrees, D. J.; Pople, J. A.; Binkley, J. S. *J. Am. Chem. Soc.* **1982,** *104,* 5039.

120. Pulay, P.; Fogarasi, G.; Pang, F.; Boggs, J. E. *J. Am. Chem. Soc.* **1979,** *101,* 2550.

121. Dill, J. D.; Pople, J. A. *J. Chem. Phys.* **1975,** *62,* 2921.

122. Ditchfield, R.; Hehre, W. J.; Pople, J. A. *J. Chem. Phys.* **1971,** *54,* 724.

123. Hehre, W. J.; Pople, J. A. *J. Chem. Phys.* **1972,** *56,* 4233.

124. Binkley, J. S.; Pople, J. A. *J. Chem. Phys.* **1977,** *66,* 879.

125. Hariharan, P. C.; Pople, J. A. *Theor. Chim. Acta* **1973,** *28,* 213.

126. Hehre, W. J.; Ditchfield, R.; Pople, J. A. *J. Chem. Phys.* **1972,** *56,* 2257.

127. Francl, M. M.; Pietro, W. J.; Hehre, W. J.; Binkley, J. S.; Gordon, M. S.; DeFrees, D. J.; Pople, J. A. *J. Chem. Phys.* **1982,** *77,* 3654.

128. Rassolov, V. A.; Pople, J. A.; Ratner, M. A.; Windus, T. L. *J. Chem. Phys.* **1998,** *109,* 1223.

129. Clark, T.; Chandrasekhar, J.; Spitznagel, G. W.; Schleyer, P. von R. *J. Comput. Chem.* **1983,** *4,* 294.

130. Frisch, M. J.; Pople, J. A.; Binkley, J. S. *J. Chem. Phys.* **1984,** *80,* 3265.

131. Krishnan, R.; Binkley, J. S.; Seeger, R.; Pople, J. A. *J. Chem. Phys.* **1980,** *72,* 650.

132. McLean, A. D.; Chandler, G. S. *J. Chem. Phys.* **1980,** *72,* 5639.

133. Mitin, A. V.; Baker, J.; Pulay, P. J. Chem. Phys. **2003**, *118*, 7775.

134. Dunning, T. H., Jr.; Hay, P. J. Chapter 1 in *Modern Theoretical Chemistry: Methods of Electronic Structure Theory;* Schaefer, H. F., III, Ed.; Plenum: New York, 1977; Vol. 3.

135. Rappé, A. K.; Goddard, W. A. Unpublished work.

136. Dunning, T. H., Jr., *J. Chem. Phys.* **1989,** *90,* 1007.

137. Kendall, R. A.; Dunning, T. H., Jr.; Harrison, R. J. *J. Chem. Phys.* **1992,** *96,* 6796.

138. Woon, D. E.; Dunning, T. H., Jr. *J. Chem. Phys.* **1993,** *98,* 1358.

139. Woon, D. E.; Dunning, T. H., Jr. *J. Chem. Phys.* **1994,** *100,* 2975.

140. Easton, R. E.; Giesen, D. J.; Welch, A.; Cramer, C. J.; Truhlar, D. G. *Theor. Chim. Acta* **1996,** *93,* 281.

141. Thompson, J. D.; Winget, P.; Truhlar, D. G. *Phys. Chem. Comm.* **2001**, *4*, 72.

142. Li, J.; Cramer, C. J.; Truhlar, D. G. *Theor. Chem. Acc.* **1998**, *99*, 192.

143. Schafer, A.; Huber, C.; Ahlrichs, R. *J. Chem. Phys.* **1994,** *100,* 5829.

144. Rappoport, D.; Furche, F. *J. Chem. Phys.* **2010**, *133*, 134105.

145. Pantazis, D. A.; Chen, X-Y.; Landis, C. R.; Neese, F. All-electron scalar relativistic basis sets for third-row transition metal atoms, *J. Chem. Theory Comput.* **2008**, *4*, 908.

146. Pantazis, D. A.; Neese, F. All-electron scalar relativistic basis sets for the lanthanides, *J. Chem. Theory Comput.* **2009**, *5*, 2229.

147. Pantazis, D. A.; Neese, F. All-electron scalar relativistic basis sets for the 6p elements, *Theor. Chem. Acc.* **2012**, *131*, 1292.

148. Dyall, K. G. Relativistic and nonrelativistic energy-optimized polarized double zeta basis sets for the 4p, 5p and 6p elements. *Theor. Chim. Acta* **1998,** *99,* 366.

149. Dyall, K. G. Relativistic double-zeta, triple-zeta, and quadruple-zeta basis sets for the 5d elements Hf – Hg. *Theor. Chem. Acc.* **2004,** *112,* 403.

150. Dyall, K. G. Relativistic quadruple-zeta basis sets and revised triple-zeta and double-zeta basis sets for the 4p, 5p and 6p elements. *Theor. Chem. Acc.* **2006,** *115,* 441.

151. Dyall, K. G. Relativistic double-zeta, triple-zeta, and quadruple-zeta basis sets for the 4d elements Y – Cd. *Theor. Chem. Acc.* **2007,** *117,* 483.

152. Dyall, K. G. and Gomes, A. S. P. Revised relativistic basis sets for the 5d elements Hf – Hg. *Theor. Chem. Acc* **2010,** *125,* 97.

153. Gomes, A. S. P.; Dyall, K. G.; Visscher, L. Relativistic double-zeta, triple-zeta, and quadruple-zeta basis sets for the lanthanide elements La – Lu. *Theor. Chem. Acc.* **2010,** *127,* 369.

154. Dyall, K. G. Relativistic double-zeta, triple-zeta, and quadruple-zeta basis sets for the light elements H – Ar. In preparation.

155. Dyall, K. G. and Gomes, A. S. P. Relativistic double-zeta, triple-zeta, and quadruple-zeta basis sets for the 3d elements Sc – Zn. In preparation.

156. Hay, P. J.; Wadt, W. R. *J. Chem. Phys.* **1985,** *82,* 270.

157. Hay, P. J.; Wadt, W. R. *J. Chem. Phys.* **1985,** *82,* 284.

158. Hay, P. J.; Wadt, W. R. *J. Chem. Phys.* **1985,** *82,* 299.

159. The LACV3P basis set is a triple-zeta contraction of the LACVP basis set developed and tested at Schrödinger, Inc.

160. Peterson, K. A. *J. Chem. Phys.* **2003,** *119*, 11099.

161. Peterson, K. A.; Figgen, D.; Goll, E; Stoll, H.; Dolg, M. *J. Chem. Phys.* **2003,** *119*, 11113.

162. Metz, B.; Stoll, H.; Dolg, M. *J. Chem. Phys.* **2000,** *113*, 2563.

163. Cundari, T. R.; Stevens, W. J. *J. Chem. Phys.* **1993,** *98,* 5555.

164. Hurley, M.; Pacios, L. F.; Christiansen, P. A.; Ross, R. B.; Ermler, W. C. *J. Chem. Phys.* **1986,** *84,* 6840.

165. Lajohn, L.; Christiansen, P. A.; Ross, R. B.; Atashroo, T.; Ermler, W. C. *J. Chem. Phys.* **1987,** *87,* 2812.

166. Ross, R. B.; Powers, J. M.; Atashroo, T.; Ermler, W. C.; Lajohn, L.; Christiansen, P. A. *J. Chem. Phys.* **1990,** *93,* 6654.

167. Ross, R. B.; Gayen, S.; Ermler, W. C. *J. Chem. Phys.* **1994,** *100,* 8145.

168. Ermler, W. C.; Ross, R. B.; Christiansen, P. A. *Int. J. Quantum Chem.* **1991,** *40,* 829.

169. Nash, C. S.; Bursten, B. E.; Ermler, W. C. *J. Chem. Phys.* **1997,** *106,* 5133.

170. Wildman, S. A.; DiLabio, G. A.; Christiansen, P. A. *J. Chem. Phys.* **1997,** *107,* 9975.

171. Diffuse and polarization functions for Ga-Kr, In-Xe, Tl-Rn taken from Ref. 148; diffuse functions for rare gases extrapolated from those for the other elements in the row. Polarization functions for Sc-Zn, Y-Cd, Hf-Hg taken from Refs. 155, 151, 149.

172. Hamilton, T. P.; Pulay, P. *J. Chem. Phys.* **1986,** *84,* 5728; Pulay, P. *J. Comput. Chem.* **1982,** *3,* 556; Pulay, P. *Chem. Phys. Lett.* **1980,** *73,* 393.

173. Obara, S.; Saika, A. *J. Chem. Phys.* **1986,** *84,* 3963.

174. Gill, P. M. W.; Head-Gordon, M.; Pople, J. A. *J. Chem. Phys.* **1990,** *94,* 5564; Gill, P. M. W.; Head-Gordon, M.; Pople, J. A. *Int. J. Quantum Chem.* **1989,** *S23,* 269; Head-Gordon, M.; Pople, J. A. *J. Chem. Phys.* **1988,** *89,* 5777.

175. Murphy, R. B.; Messmer, R. P. *J. Chem. Phys.* **1993,** *98,* 10102.

176. For information on Molden, see the Molden web site http://www.caos.kun.nl/~schaft/molden/molden.html.

177. Stewart, J. J. P. *MOPAC 6;* QCPE #455.

178. Hohenberg, P.; Kohn, W. *Phys. Rev. B* **1964,** *136,* 864.

179. Kohn, W.; Sham, L. J. *Phys. Rev. A* **1965,** *140,* 1133.

180. Parr, R. G.; Yang, W. *Density-Functional Theory of Atoms and Molecules;* Oxford: New York, 1989.

181. *Density Functional Methods in Chemistry;* Labanowski, J. K., Andzelm, J. W., Eds.; Springer-Verlag: Berlin, 1991.

182. Gill, P. M. W.; Adamson, R. D.; Pople, J. A. *Mol. Phys.* **1996**, *88*, 1005.

183. Leiniger, T.; Stoll, H.; Werner, H.-J.; Savin, A. *Chem. Phys. Lett.* **1997**, *275*, 151.

184. Dyall, K. G.; Fægri, K. *Introduction to Relativistic Quantum Chemistry;* Oxford: New York, 2007.

185. Chang, Ch.; Pélissier, M; Durand, P. *Physica Scripta* **1986**, *34*, 394.

186. van Lenthe, E.; Baerends E. J.; Snijders, J. G. *J. Chem. Phys.* **1993,** *99,* 4597.

187. van Wüllen, C. *J. Chem. Phys.* **1998**, *109*, 392.

188. Faas, S.; Snijders, J. G.; van Lenthe, J. H.; van Lenthe, E.; Baerends E. J. *Chem. Phys. Lett.* **1995**, *246*, 632.

189. van Lenthe, E.; Baerends E. J.; Snijders, J. G. *J. Chem. Phys.* **1994,** *101,* 9783.

190. Colle, R.; Salvetti, O. *J. Chem. Phys.* **1990,** *93,* 534.

191. Kraka, E. *Chem. Phys.* **1992,** *161,* 149.

192. Audi, G.; Wapstra, A. H. *Nuclear Phys.* **1995,** *A595 4,* 409.

193. Császár, P.; Pulay, P. *J. Mol. Struct.* **1984,** *114,* 31.

194. Schlegel, H. B. *J. Comput. Chem.* **1982,** *3,* 214.

195. Powell, M. J. D. *Math. Prog.* **1971,** *1,* 26.

196. Bofill, J. M. *J. Comp. Chem.* **1994,** *15,* 1.

197. Murtagh, B. A.; Sargent, R. W. H. *Comp. J.* **1970,** *13,* 185.

198. Fletcher, R. In *Practical Methods of Optimization;* Wiley: New York, 1987.

199. Banerjee, A.; Adams, N.; Simons, J.; Shepard, R. *J. Phys. Chem.* **1985,** *89,* 52.

200. Culot, P.; Dive, G.; Nguyen, V. H.; Ghuysen, J. M. *Theor. Chim. Acta* **1992,** *82,* 189.

201. Simons, J.; Jorgensen, P.; Taylor, H.; Ozment, J. *J. Phys. Chem.* **1983,** *87,* 2745.

202. Häser, M.; Ahlrichs, R. *J. Comput. Chem.* **1989,** *10,* 104; Cremer, D.; Gauss, J. *J. Comput. Chem.* **1986,** *7,* 274; Almlöf, J.; Faegri, K., Jr.; Korsell, K. *J. Comput. Chem.* **1982,** *3,* 385.

203. Rabuck, A. D.; Scuseria, G. E. *J. Chem. Phys.* **1999,** *110,* 695.

204. Bayly, C. I.; Cieplak, P.; Cornell, W. D.; Kollman, P. A. *J. Phys. Chem.* **1993,** *97,* 10269.

205. Stroud, A. H. *Approximate Calculation of Multiple Integrals;* Prentice-Hall: New York, 1971.

206. Lebedev, V. I. *Zh. vychisl. Mat. mat. Fiz.* **1975,** *15,* 48-54.

207. Lebedev, V. I. *Zh. vychisl. Mat. mat. Fiz.* **1976,** *16,* 293-306.

208. Lebedev, V. I. *Sibirsk. Mat. Zh.* **1977,** *18,* 132-142.

209. Lebedev, V. I. In *Theory of Cubature Formula and Numerical Mathematics* (in Russian); Sobolev, S. L, Ed. "Nauka" Sibirsk, Otdel.: Novosibirsk, 1980; pp 110-114.

210. Cramer, C. J.; Truhlar, D. G. *J. Comp. Aided Mol. Design* **1992,** *6,* 629.

211. Marten, B.; Kim, K.; Cortis, C.; Friesner, R. A.; Murphy, R. B.; Ringnalda, M. N.; Sitkoff, D.; Honig, B. New Model for Calculation of Solvation Free Energies: Correction of Self-Consistent Reaction Field Continuum Dielectric Theory for Short-Range Hydrogen-Bonding Effects. *J. Phys. Chem.* **1996,** *100,* 11775.

212. Marenich, A. V.; Olson, R. M.; Kelly, C. P.; Cramer, C. J.; Truhlar, D. G. Self-consistent reaction field model for aqueous and nonaqueous solutions based on accurate polarized partial charges, *J. Chem. Theory Comput.* **2007**, *3*, 2011.

213. Hoijtink, G. J.; Boer, E. D.; Meij, P. H. v. D.; Weijland, W. P. *Recl. Trav. Chim. Pays-Bas.* **1956**, *75*, 487.

214. Tucker, S. C.; Truhlar, D. G. *Chem. Phys. Lett.* **1989**, *157*, 164.

215. Still, W. C.; Tempczyk, A.; Hawley, R. C.; Hendrickson, T. *J. Am. Chem. Soc.* **1990**, *112*, 6127.

216. Zhu, T.; Li, J.; Hawkins, G. D.; Cramer, C. J.; Truhlar, D. G. Density Functional Solvation Model Based on CM2 Atomic Charges. *J. Chem. Phys.* **1998**, *109*, 9117.

217. Cramer, C. J.; Truhlar, D. G. In Reviews in Computational Chemistry; Boyd, D. B., Lipkowitz, K. B., Eds.; VCH Publishers: New York, **1995**; Vol. 6; pp 1.

218. Cramer, C. J.; Truhlar, D. G. *Chem. Rev.* **1999**, *99*, 2161.

219. Liotard, D. A.; Hawkins, G. D.; Lynch, G. C.; Cramer, C. J.; Truhlar, D. G. Improved Methods for Semiempirical Solvation Models. *J. Comput. Chem.* **1995**, *16*, 422.

220. Kelly, C. P.; Cramer, C. J.; Truhlar, D. G. SM6: A Density Functional Theory Continuum Solvation Model for Neutrals, Ions, and Ion–Water Clusters. *J. Chem. Theory Comput.* **2005**, *1*, 1133.

221. Olson, R. M.; Marenich, A. V.; Cramer, C. J.; Truhlar, D. G. Charge Model 4 and intramolecular charge polarization, *J. Chem. Theory Comput.* **2007**, *3*, 2046.

222. Löwdin, P.-O. *J. Chem. Phys.* **1950**, *18*, 365.

223. Baker, J. *Theor. Chim. Acta* **1985**, *68*, 221.

224. Thompson, J. D.; Xidos, J. D.; Sonbuchner, T. M.; Cramer, C. J.; Truhlar, D. G. *PhysChemComm* **2002**, *5*, 117.

225. Mayer, I. *Chem. Phys. Lett.* **1983**, *97*, 270.

226. Mayer, I. *Chem. Phys. Lett.* **1985**, *117*, 396.

227. Mayer, I. *Int. J. Quantum. Chem.* **1986**, *29*, 73.

228. Chasman, D.; Beachy, M. D.; Wang, L.; Friesner, R. A. Parallel Pseudospectral Electronic Structure. I. Hartree-Fock Calculations. *J. Comput. Chem.* **1998,** *19,* 1017.

229. Beachy, M. D.; Chasman, D.; Murphy, R. B.; Friesner, R. A. Parallel Pseudospectral Electronic Structure. II. Localized Møller-Plesset Calculations. *J. Comput. Chem.* **1998,** *19,* 1030.

230. Jang, Y. H.; Sowers, L. C.; Cagin, T.; Goddard, W. A., III. *J. Phys. Chem. A* **2001,** *105,* 274.

231. Langlois, J. -M. Ph.D. Dissertation, California Institute of Technology, Pasadena, CA, 1994.

232. Perez-Jorda, J. M.; Becke, A. D.; San-Fabian, E. *J.Chem. Phys.* **1994,** *100,* 6520.

233. Baker, J.; Andzelm, J.; Scheiner, A.; Delley, B. *J.Chem. Phys.* **1994,** *101,* 8894.

234. Mura, M. E.; Knowles, P. J. *J. Chem. Phys.* **1996,** *104,* 9848.

235. Gonzalez, C.; Schlegel, H. B. *J. Chem. Phys.* **1989,** *90,* 2154; *J. Chem. Phys.* **1990,** *94,* 5523.

236. Klicic, J. J.; Friesner, R. A.; Liu, S.-Y.; Guida, W. C. *J. Phys. Chem. A* **2002,** *106,* 1327.

237. De Proft, F.; Van Alsenoy, C.; Peeters, A.; Langenaeker, W.; Geerlings, P. *J. Comp. Chem.* **2002,** *23*, 1198.

238. Contreras, R. R.; Fuentealba, P.; Galvan, M.; Perez, P. *Chem. Phys. Lett.* **1999,** *304*, 405.

239. Chamorro, E.; Perez, P. *J. Chem. Phys.* **2005,** *123*, 114107.

240. Ko, C.; Malick, D. K.; Braden, D. A.; Friesner, R. A.; Martinez, T. J. *J. Chem. Phys.* **2008**, *128*, 104103.

241. Michalak, A.; De Proft, F.; Geerlings, P.; Nalewajski, R. F. *J. Phys. Chem. A* **1999**, *103*, 762.

242. Hussein, W.; Walker, C. G.; Peralta-Inga, Z.; Murray, J. S. *Int. J. Quantum Chem.* **2001**, *82*, 160.

243. Ehresmann, B.; de Groot, M. J.; Alex, A.; Clark, T. *J. Chem. Inf. Comput. Sci.* **2004**, *44*, 658.

244. Jurecka, P.; Sponer, J.; Cerny, J.; Hobza, P. *Phys. Chem. Chem. Phys.* **2006**, *8*, 1985.

245. Bobadova-Parvanova, P; Galabov, B. *J. Phys. Chem. A*, **1998**, *102*, 1815.

246. Dimitrova, M; Galabov, B. *Croat. Chem. Acta*, **2009**, *82*, 21.

247. Stephens, P. J.; Ashvar, C. S.; Devlin, F. J.; Cheeseman, J. R.; Frisch, M. J. *Mol. Phys.* **1996**, *89*, 579.

248. Cheeseman, J. R.; Frisch, M. J.; Devlin, F. J.; Stephens, P. J. *Chem. Phys. Lett.* **1996**, *252*, 211.

249. Johnson, E. R.; Keinan, S.; Mori-Sanchez, P.; Contreras-Garcia, J.; Cohen, A. J.; Wang, W. *J. Am. Chem. Soc.* **2010**, *132*, 6498.

250. Liu, J.; Kelly, C. P.; Goren, A. C.; Marenich, A. V.; Cramer, C. J.; Truhlar, D. G.; Zhan, C.-G. *J. Chem. Theor. Comput.* **2010**, *6*, 1109.

251. Jain, R.; Bally, T.; Rablen, P. R. *J. Org. Chem.* **2009**, *74*, 4017.

252. Giesen, D. J.; Zumbulyadis, N. *Phys. Chem. Chem. Phys.* **2002**, *22*, 5498.

Index

A

a posteriori corrections................ 46–49, 137, 213
accuracy level.. 54, 92
  keyword 246
acidic site, designating for pK_a calculations... 362
AIMPAC `.wfn` file, keyword........................ 260
all-analytic calculation, keyword.................. 247
analytic gradient of energy............................... 93
  convergence criteria..................................... 93
  keywords... 217, 226
analytic integral corrections
  keyword 247
  specifying in `.cutoff` file....................... 310
  theory 185
angles—*see* bond angles
anions, solvating 230
antiferromagnetic systems 169, 281
`.ark` file
  specifying in input file............................. 195
atom labels
  format 13, 15
  in output... 132
  orbital output 88, 263
atomic charges—*see* charges, atomic
atomic masses
  for frequency calculations 70
  keyword 205
  setting in atomic section 273–275
atomic properties,
 setting in atomic section 273–281
atomic units—*see* units
`.atomig` file
  default... 293
  description and format...................... 298–300
  specifying in input file............................. 195
atoms
  counterpoise...................................... 18, 276
  dummy... 17
average local ionization energy................... 75, 77

B

babel
  using to convert file formats............ 254–255
  using to read input files 21
`.basis` file
  description and format..................... 293–297
  specifying in input file............................. 195
basis functions
  contracted 160, 161
  derivatives of, list in output 258
  file containing.................................. 293–297
  for individual atoms........................... 33, 276
  in counterpoise calculations 18–19, 199
  keyword for printing................................ 258
  listing in output... 153
  Mulliken populations for.......................... 152
  number of 34, 132, 299
  output... 160–163
  type, as listed in output............................... 88
  uncontracted 160, 161
basis set.. 33–38
  conversion to Jaguar format 297
  diffuse functions.................. 33–34, 244, 294
  file containing.................................. 293–297
  for individual atoms........................... 33, 276
  for initial guess 168, 283, 298
  for SM6 solvation....................................... 61
  in generated GAUSSIAN input file 180
  keyword for list in output 258
  keywords... 244
  listed in output 132
  names.. 34–37
  polarization functions........... 33–37, 244, 294
  specifying for GAUSSIAN input......... 181, 260
  with ECP, in output................................... 160
basis set superposition error.......... 18, 50, 52, 186
batch input file
  example 327
  format 322–325
batch jobs
  distributing over multiple CPUs................. 28
  jaguar batch command for................ 322–326
  remote 327
  running from Maestro......................... 26–29
batch scripts 322
  built-in 28
  selecting in Maestro 26
BFGS method for Hessian updating, keyword 223
BIOGRAF `.hes` files, format....................... 282
bond angles
  freezing all.. 218
  freezing for geometry optimization........... 17,
   96–97, 200
  in Z-matrix.. 15
  output keyword... 258

scan range 106
bond critical points,
displaying and undisplaying 180
bond lengths
freezing all 218
freezing for geometry optimization 17, 96–97, 200
in Z-matrix 15
output keyword 258
bonds, specifying for internal coordinates 203
Boys localization 58
keywords 207, 254
orbital printing 87, 163, 254, 262
output from 163

C

canonical orbital space, output in 261
Cartesian coordinates
format for geometry 13–14, 198–199
freezing for geometry optimization 14, 96–97
`ch` program 289
output from 141, 142, 145–156
charge fitting—*see* ESP fitting, Mulliken population analysis
charge, molecular
keywords 205
setting in Maestro 21
charges, atomic
CM4 68
from ESP fit 62
keyword for formal 274
Mulliken 64
stockholder (Hirshfeld) 239
chemical shifts 67, 241
chirality, determining 74, 119, 242
CIS calculations 216
`cis` program 289
command options, `jaguar run` 317
comment lines
batch script 322
input file 195, 198
configuration interaction singles (CIS) calculations 50, 216
connectivity
keyword for bonding 205
output keyword 258
output option 84
constraints
dynamic 99, 202
equivalent coordinates 98
for geometry optimization 14, 17, 96–97, 200, 218
frozen coordinates 96, 202
harmonic 97, 203
natural torsional angle 97, 203
transition state searches 102
contour plots 110
conventions, document xiii
convergence criteria
geometry optimization 93, 141, 226
SCF energy 56, 175
SCF energy, keyword 250
solvation energy, keyword 232
convergence problems, troubleshooting 345
convergence schemes 57
DIIS 57, 163
keywords 250
OCBSE 57, 163
coordinates
Cartesian, in geometry input 13
constraining and freezing 96–99
for refinement of Hessian 19, 105
reaction 110, 229
redundant internal 95
scan 106–109
Coulomb field, charge fitting to 149
Coulomb operator (J) 247
keyword for output 261
obtaining i/o information for 83
pseudospectral assembly of 184–185
counterpoise calculations 18–19, 199
defining fragments for 281
Python script for 329
specifying atoms for 276
coupled perturbed Hartree-Fock (CPHF) terms
for LMP2 dipole moments 51, 65
for LMP2 ESP fitted charges 51, 63
covalent radii 276
`cpolar` program 289
cpu time 258
Culot-Fletcher method for trust radius adjustment, keyword 224
`.cutoff` file 308–310
default 293

description and format 308
specifying in input file 195
cutoff methods 54, 92, 246
cutoffs 246, 248, 308–310
shown in output 135, 164

D

.daf file
default 293
description and format 300–305
neighbor ranges 300
specifying in input file 195
dealiasing functions 35, 300
choices for calculation 301
contracted 301, 302–303
keyword for list in output 258
keywords 263–264
long-range 300, 302, 303
neighbor ranges for 84, 300–302
ordering of sets 302
output of number used 258
short-range 300–301, 303
uncontracted 300, 302
default.atomig file 298
delocalization of LMP2 pairs 207, 271
density difference matrix
keyword for output of 261
RMS of elements in output 135, 164
density functional theory (DFT) . 38–49, 188–189
a posteriori corrections 46–49, 137, 213
constrained 49
gradient-corrected 41–42
hybrid methods 42–44
keywords 208–215, 263
long-range corrections 44, 189
output from 136
standard functionals 40–49
time-dependent 39
density matrix
convergence criterion 56, 92
convergence criterion, keyword 250
in DIIS error vector 135, 164
keyword for output 261
density—*see* electron density, spin density
der1a program 289
output from 139–141
der1b program 289
output from 139–141
derivatives of basis functions 85
keyword for list in output 258
dftname values
name strings for construction of 212
standard functional names 209
DFT—*see* density functional theory
dielectric constant
keywords 232
keywords for 232
setting in the Solvation tab 60
dielectric continuum method—*see* pbf program
DIIS convergence scheme 57, 135, 164
keyword for coefficient output 261
keywords 250
output 135
dipole moment—*see* multipole moments
Direct Inversion in the Iterative Subspace method—*see* DIIS convergence scheme
directory
input 327
installation 2
local 327
Maestro working 3
output 323, 327
scratch 323
working 323
displacement
convergence criteria based on 93
keywords for convergence criteria 226
display host 341, 342–343
Display Options dialog box 79
distributed processing 28
dsolv program 290
dummy atoms
in charge fitting 149
in the Hessian 282
in Z-matrix input 17
dynamic constraints 99, 202

E

Edit Job dialog box 12
effective core potentials (ECPs)
basis sets for 36–38
electrostatic potential at nuclei 64
in atomic guess file 299

in basis set file 295–297
NMR shielding constants 67
eigenvector following in transition-state optimizations
keywords.. 222
option setting 104
use of Hessian refinement 105
`elden` program.. 289
electric field for polarizability calculations
input file section 285
keyword 238
setting 67
electric field, output on a grid........................ 240
electron density
keywords.. 238, 264
output from calculation 152
surface... 75, 77
use for solvation cavity............................ 230
electron density output file............................ 152
electronic spectra 40, 50
electrophilic attack 75
electrostatic potential
in `.resp` file.. 260
output on a grid.. 240
statistical analysis 156, 241
surface... 75, 77
electrostatic potential fitting 62–64
constraining to reproduce multipole moments 63–64
for LMP2 wave functions 63
for solvation calculations........... 58, 141, 142
grid for 64
keywords.. 237
keywords for grid............................ 237, 265
output from 149–150
recalculating multipole moments from. 64–65
RMS error in output 149
setting options for 62
energy components, keyword for output of 261
energy convergence criterion 56
keyword for... 250
energy difference
as geometry convergence criterion 93, 141
keyword for geometry convergence.......... 226
energy output
final SCF, components............................ 136
SCF components for each iteration 261
solvation 142–145
total SCF, for each iteration............. 135, 164
two-electron contributions....................... 163
enthalpy calculations 74, 157, 243
entropy calculations 74, 157, 243
environment variable
`DISPLAY`.. 342
`SCHRODINGER` 2, 342
error messages
can't open display.................................... 342
command not found................................. 343
memory-related 346
temp directory 344
ESP—*see* electrostatic potential
exchange corrections, in pseudospectral calculations 310
exchange operator (K)
keyword for per iteration output............... 261
keywords for calculation 247
obtaining i/o information for 83
pseudospectral assembly of............. 184–185
excited-state calculations
keywords.. 216
TDDFT settings.. 39
TDHF settings 50
executable directory 132
execution hosts
listing available....................................... 316
selecting.. 318
troubleshooting....................................... 344
execution path 288–291

F

file output options 85–86
GAMESS input file 260
GAUSSIAN input (`.gau`) file 180
keywords.. 259–260
XYZ file 86
first solvation shell
correction factor keyword........................ 231
SM8, SM6 corrections................................ 61
Fock matrix
in DIIS error vector 135, 164
keywords for output of 259, 261
new estimate from DIIS scheme 135, 164
pseudospectral assembly of............. 183–185
updating 135, 164

updating, keyword 247
forces, keywords 217, 218, 223
formal charge 274
fragments
defining 281
frequencies for 242, 281
`freq` program 289
frequencies 69–74
fragment 242, 281
keywords 223, 242–243
output 157–159
scaling 70–71
visualizing in Maestro 72
visualizing with Molden 73
frequency-related properties
Maestro settings 73–74
output 157–159
Fukui functions 75, 117
Fukui indices 117, 239
functionals—*see* density functional theory (DFT)

G

GAMESS input files
keyword for 260
option for 86
GAUSSIAN
Hessian format 282
orbital output in format for 88, 262
GAUSSIAN basis set file (`.gbs`), keyword for generation of 260
Gaussian function list in output, keywords 258
GAUSSIAN input file (`.gau`)
keyword for generation of 180, 260
option for generation of 85, 180–181
generalized gradient approximation 189
geometry input 13–20
Cartesian format 13–14, 198–199
format 198–204
input file sections 198–204
keywords 204
output file, echoed in 132
sample calculation 8
symmetrizing 23–24
troubleshooting 345
units, keyword for 198, 204, 284
Z-matrix format 15–17, 199–201
geometry optimization 91–99
calculating forces only 217
constraining bond lengths or angles 17, 96–97, 200
constraining Cartesian coordinates. 14, 96–97
convergence criteria 92, 93, 141, 226
coordinate system 95
detailed output, option for 84
frozen bond lengths or angles for .. 17, 96–97, 200
frozen Cartesian coordinates for 14, 96–97
GDIIS method 219
generating input with new geometry 171
in solution 60, 217, 230
initial Hessian for 19–20, 94–95, 199–200, 223, 282–283
input file section for Hessian 282–283
keywords 217–226
limiting step size for 224, 230
maximum number of iterations .. 93, 172, 218
output description 139
output of forces 139
output options 84
refinement of initial Hessian for 19–20, 104–105, 199–200, 223
saving intermediate structures 93
tips 171–172
troubleshooting 345
trust radius for 224, 230
updating of Hessian during 223
see also transition-state optimization
geometry scans 106–110
geometry, translation and rotation of during calculation 132
`geopt` program 289
output 139–140
ghost atoms 18
use in charge fitting to bond midpoints 149
Gibbs free energy calculations 74, 157, 243
gradient—*see* analytic gradient of energy
grid and RwR information, output keyword ... 258
`.grid` file
default 293
description and format 305–307
specifying in input file 195
`grid` program 289
output from 134, 140, 141

grid shell locations, keyword for output of..... 258
grids ... 92
  custom ... 196, 265
  for electrostatic potential fitting . 64, 237, 265
  grid file ... 195
  information in log file... 164
  information in output... 134–135
  keywords... 263–265
  selecting DFT ... 40
  shells for, in grid file... 306–307
  specified in cutoff file... 309–310
grids, pseudospectral
  accuracy level ... 54, 246
  basis set availability... 35
  keywords... 248, 264, 265

H

Hamiltonians
  information in output... 135
  relativistic ... 191, 272
  user input of... 286
harmonic constraints ... 97, 203
harmonic frequencies ... 157
Hartree-Fock (HF) calculations
  DIIS error vector definition ... 164
  keywords for SCF settings... 245–248
  output from standard ... 132
  printing orbitals ... 87, 262
  settings for ... 50
heat capacity calculations ... 74, 157, 243
Heat of Formation panel ... 121
Help panel ... 341
Hessian... 94–95, 104–105
  coordinates for refinement of ... 19, 105
  effect of quality on geometry convergence 93, 104
  input file section ... 282–283
  IRC calculations ... 112, 229
  keywords... 223–224
  level shifting ... 223, 224
  refinement of initial ... 19–20, 104–105, 199–200, 223
  selecting initial ... 94–95, 223
  updating, keyword for... 223
heteroatom pairs
  local LMP2 calculations....... 52–53, 207, 259
`hfig` program ... 289
  output from ... 132
hybrid methods, DFT... 40
hydrogen bond energies, batch script for 115, 330
hyperpolarizability
  keywords... 238–240
  output... 150
  selecting options ... 66–67

I

infrared intensities ... 73, 242
  output... 159
initial guess
  antiferromagnetic ... 169, 281
  choosing type... 167–170
  file information for ... 293, 298–300
  improving ... 168–170
  information in output... 132
  input file section for ... 283–284
  keywords... 252
  orbital output in format for... 88, 262
  printing orbitals after... 87, 262
  restarted calculations ... 168
  stopping after... 180, 251
  transition metal systems ... 167–168, 168–170
input file... 195–198
  defining fragments... 281
  description of sections ... 196–198
  echoing in output file... 82, 288
  editing... 293
  gen section... 181
  general description and format......... 195–198
  in jaguar run command... 317–318
  keywords... 204–266
  name ... 318
  reading in Maestro... 20–21
  restart... 174–175
  saving... 24–25
  section delineators ... 197
  selecting in Maestro ... 28
  spacing characters ... 196
  summary of sections... 197
input file sections
  atomic ... 273–281
  connect ... 201–204
  coord... 201–204
  echo ... 288

efields 285
gen 204–266
guess 283–284
ham 286
hess 282–283
lmp2 .... 207, 270–272
nbo 291
orbman 286–287
path 288–291
pointch 284–285
zmat, zmat2, zmat3 198–200
zvar 199, 200
zvar, zvar2, zvar3 200–201
input of molecular structure—*see* geometry input
Input tab 32
installation directory 341
integrals, one-electron 132
integrals, two-electron
pseudospectral approximation to 185
internal coordinates
in optimization, keyword for 218
specifying with connect section 201–204
specifying with coord section 201–204
internal energy calculations 74, 157, 243
intrinsic reaction coordinate (IRC) calculations 110, 228
IR intensities—*see* infrared intensities
`ira` program 289
`irb` program 289
IRC tab 111
isotopes
keyword, atomic section 275
keyword, gen section 205
iterations, maximum number
geometry optimization 93, 172, 218
SCF 56, 170, 246

J

`jagconvert` utility 319
`jaguar` command 315–319
`jaguar babel` 318–319
`jaguar batch` 322–326
`jaguar help` 316
`jaguar results` 125–131
`jaguar run` 317–318
Jaguar copyright information 132
Jaguar files 293
Jaguar panel 10, 116, 118
Jaguar programs 289
Jaguar Read dialog box 21
Jaguar Write dialog box 24
`jexec` program 289
job directory, local 29
job name 132, 316, 318
Job Settings dialog box 26

L

launch host 342–343
LDA—*see* Local Density Approximation
least-squares operator Q, description of . 183–184
level shifting 56
Hessian, keyword for 224
virtual orbitals, keyword 249
Lewis dot structure
setting van der Waals radii from 142, 231
`.lewis` file
description and format 311–313
Linear Synchronous Transit (LST) methods—*see* QST-guided transition-state searches
LMP2 method 50–53, 185–188
counterpoise corrections 19
grid 264
input file section 270–272
keywords 206–207
output from 138
pseudospectral implementation 185–188
settings 50–53
LMP2 pairs
delocalization of 207, 271–272
input file section for 207, 270–272
keywords 207, 208
`lmp2` program 289
`lmp2der` program 289
`lmp2dip` program 289
`lmp2gda` program 289
`lmp2gdb` program 289
output from 139
Local Density Approximation (LDA) 41, 189
dftname values for 209
local job directory 29
local LMP2 method 52–53
grid 264
keywords 259

local MP2 method—*see* LMP2 method, local LMP2 method
`local` program 289
localization of orbitals
  keywords 207, 253–254
  LMP2 calculations 52, 207
  options for final 58
log file 164–165
  batch jobs 29
long-range corrected functionals 44, 189
Löwdin population analysis 68

M

Maestro
  problems starting 342–343
Maestro file, specifying in input file 196
`makejbasis` utility 297
Manage Surfaces panel 78
masses
  for frequency calculations 70
  keyword 205
  setting in atomic section 273–275
Mayer bond orders 68, 148
memory
  keywords 265–266
  troubleshooting related to 346
memory, disk, and i/o information
  keyword 258
  output option 83
minimum energy path (MEP) calculations 110
Molden orbitals file (`.molf`), keyword for 260
molecular charge
  keyword 205
  Maestro setting 21
molecular properties—*see* properties
molecular state keywords 205
molecular structure—*see* geometry input, geometry optimization
Møller-Plesset second-order perturbation theory—*see* LMP2
MP2—*see* LMP2 206
MQM basis set file (`.bas`), keyword 260
Mulliken population analysis 64–65
  for basis functions 65
  keyword 238
  output from 152–153
  output of multipole moments from 153
  recalculating multipole moments from 64, 65
Mulliken spin populations 64
multiple Jaguar jobs, running
  from Maestro 26–29
  with `jaguar batch` 322–326
multiplicity
  keyword 205
  setting in Maestro 21–22
multipole moments
  calculating 65
  from electrostatic potential fitting 64–65, 150
  from Mulliken population analysis 64, 65, 153
  keyword 238
  output description 149
  output option 84
  tensors listed in output 149
  units keyword 258
Murtagh-Sargent method, keyword 223

N

Natural Bond Orbital (NBO) calculations 65, 156, 291
natural torsional angle constraints 97, 203
NCI plot
  coloring by interaction strength 179
NCI plots 76
neighbor ranges 300
NMR shielding constants
  keyword 238
  settings 67
noncovalent interactions 76
  bond critical point display 180
  calculating 75
  coloring NCI plot 179
  setting display for 79
non-local ensity approximation (NLDA) 189
nucleophilic attack 77
`nude` program 289
number of iterations for geometry convergence, maximum 93, 172, 218
number of processors, selecting 317
numeric updating of Hessian, keyword 223
numerical gradient of energy 91
  keywords 218, 223
numerical Hessian, printing in freq output 259
numerical methods 183–185
  `.cutoff` file determination of 310

numerical second derivative of energy.............. 70
  keywords.. 223, 242

O

OCBSE convergence scheme........................... 57
`onee` program 289
  output from 132, 140, 141
one-electron Hamiltonian
  keyword for output of.............................. 258
  output option... 85
  relativistic correction 191
one-electron integrals.................................... 132
  energy contributions 142, 144
open-shell singlet, keyword 252
open-shell systems
  energy contributions in output.................. 136
  keywords.. 246
Optimization tab 92
optimizations—*see* geometry optimization
orbital energies in output 136
orbitals
  combining.. 286–287
  fixing symmetry populations...................... 57
  information in output............................... 132
  keywords for output of 261–263
  output options 86–89
  plotting... 75–78
  printing for guess section 283
  reordering 286–287
  surfaces 75, 77
order of Jaguar programs run, specifying 288–291
ordering of dealiasing functions 302
organometallics,
 improving convergence for .. 167–168, 168–170
output file
  echoing input file in................................. 288
  location 9, 29
  reference in log file.................................. 165
  standard output settings 82–85
  summarizing 125–131
output file information
  basis set.. 160
  convergence methods other than DIIS...... 163
  DFT calculation options 136
  frequency, IR, and thermochemistry
   calculations.. 157
  geometry and transition-state
   optimizations.. 138
  LMP2 calculation options 138
  properties.. 149
  solvation calculations 141
output options 82–89
  atomic units 84
  bond lengths and angles 84
  connectivity table 84
  detailed timing information........................ 83
  echo input file and parameter list 82
  files 85–86, 259–260
  Gaussian function list 85
  geometry optimization details 84
  memory, disk, and i/o information 83
  one-electron Hamiltonian........................... 85
  orbitals 86–89, 261–263
  overlap matrix.. 85
  per iteration 261
  standard 82–85, 258–259
Output tab 83
output, summarizing 125–131
output—*see* output file, output options, standard
 output, file output options, per iteration output
 options, orbitals, babel
overlap matrix
  in DIIS error vector 135, 164
  keyword for eigenvector and eigenvalue
   output.. 258
  keyword for output of.............................. 258
  output option 85
  smallest eigenvalue, listed in output......... 132

P

parallel execution
  selecting number of CPUs.......................... 25
partial charges
  from ESP fit.. 62
  Mulliken 64
path specifying order of programs 288–291
`pbf` program... 290
  output from geometry optimizations........ 148
`pbf_radii.ark` file 293
per iteration output options 261
physical constants and conversion factors,
 keyword 246

Pipek-Mezey localization.............................. 58
in LMP2 calculations.............................. 52
keywords.. 207, 254
orbital printing.. 87
pK_a calculations
conformational flexibility........................ 351
empirical corrections.............................. 350
equivalent sites 351, 352
geometry optimization............................ 348
initial geometry.. 366
input files.. 362
monitoring.. 365
multiple protonation sites......... 351, 353, 360
running... 360, 363
single point energies................................ 349
solvation free energy 349
theory.. 348
pK_a prediction module
training set results.................................. 353
plot data, generating............................... 77, 267
point charges, input file section for......... 284–285
Poisson-Boltzmann equations........................... 58
Poisson-Boltzmann solver 58–60
output from 142
parameter file... 195
`polar` program... 289
polarizability
analytic 66
keywords... 238–240
Maestro options 66–67
polarization free energy 147
population analysis
Löwdin... 68
Mulliken 64
see also Mulliken population analysis
`post` program 290
output from 142
potential energy surface scan 106–110
potential, electrostatic
output on a grid... 240
surface.. 75, 77
Powell update method, keyword.................... 223
`pre` program.. 289
output from 132, 139, 142
pressure for thermochemical calculations
keyword 243
option... 74
output... 157
P-RFO level shifting, keyword for.................. 223
`probe` program 289
probe radius of solvent............................ 60, 380
processors, selecting number of..................... 317
product geometry
for IRC calculations 112
in transition-state search.......................... 100
specifying in input file............................. 200
specifying in Maestro 102
product installation 372
program order, specifying....................... 288–291
programs included in Jaguar.......................... 289
properties
ESP analysis 156, 241
ESP charge fitting................................ 62–64
IR intensities... 73
keywords... 236–241
Mulliken population analysis 64–65
multipole moments 65
NMR shielding constants 67
options 62–74
output... 149–156
polarizability and hyperpolarizability .. 66–67
thermochemical 74
Properties tab 63, 66, 69, 76
pseudospectral method........................... 183–185
Python scripts.. 327

Q

QST-guided transition-state searches...... 100–101
additional structures for.................... 200–201
keyword 222
LMP2 delocalization for.................. 207, 271
quadratic energy error, output keyword.......... 259
quadratic synchronous transit—*see* QST-guided transition-state searches
quitting Maestro 9

R

radian units for geometry input...................... 204
radius
covalent.. 276
van der Waals 275
Raman intensities..................................... 74, 242
output... 159

reactant geometry
 for IRC calculations ... 112
 in transition-state search ... 100
 specifying in input file ... 200
 specifying in Maestro ... 102
Reaction tab ... 122
redundant internal coordinates ... 95
 keyword ... 218
 suppression of use ... 14
relativistic effects ... 37
relativistic Hamiltonians ... 191, 272
resonance
 in LMP2 calculations ... 52
RESP file, keyword ... 260
restarting calculations ... 174–175
 initial guess ... 168
 restart file ... 259, 261
 with improved guess ... 171
restricted open-shell wave functions
 keyword ... 246
 Maestro setting ... 38, 50
results, summary of ... 125–131
 final ... 130
 for each atom ... 131
 intermediate ... 131
RFO level shifting, keyword for ... 223
Run Batch File panel ... 27
running jobs
 from Maestro ... 8–9, 25–29
 from the command line ... 317–318
 multiple ... 26–29
 on a remote host ... 318
 troubleshooting ... 341–344
RwR information, keywords for output of ... 258
`rwr` program ... 289
 execution sequence ... 139, 140

S

sample calculation ... 7–9
scaling frequencies ... 70–71
 keywords ... 243
Scan tab ... 107
scanning geometries ... 106–110
SCF energy output ... 135
SCF iterations
 keywords ... 261
 maximum number of ... 56, 170
SCF methods, keywords ... 245–250
`scf` program ... 289
 output from ... 134–136, 140, 141–144
 solvation calculations ... 142
SCF tab ... 54
 accuracy level ... 54
 convergence controls ... 57
 localization of orbitals ... 58
Schrödinger contact information ... 372
`SCHRODINGER` directory ... 341
`schrodinger.hosts` file ... 316, 344, 345
scratch directory ... 323
SCRF method—*see* self-consistent reaction field
scripts
 Jaguar batch ... 322
 Python ... 327
search method, transition-state ... 100
second derivative of energy, keywords ... 223, 242
Select Batch Script panel ... 27
self-consistent reaction field method for solvation calculations ... 58–62, 141
shells, information in output ... 132
Simons' method for trust radius adjustment, keyword ... 224
singlet, open shell, keyword for ... 252
`sm6` program ... 290
`sm8` program ... 290
`sole` program ... 290
 output from ... 144–145
`solv` program ... 290
 output from ... 142
solvation ... 58–60
 anions ... 230
 cavity models ... 230
 energy output ... 144
 keywords for ... 230–232
 output from calculations ... 141–148
 Poisson-Boltzmann parameter file ... 196
 probe radius ... 60, 232
 solvent choice ... 59–60
 van der Waals radii ... 142, 275–276, 311–313
solvation free energy
 PBF ... 58
 SM8, SM6 ... 61
Solvation tab ... 60
solvent parameters ... 59
SPARTAN archive files ... 86, 260

spectra
  averaged ... 81
  generating IR data ... 73
  generating UV/visible data ... 40, 50
  generating VCD data ... 75
  importing data ... 80
spectrum file
  infrared ... 73
  UV/visible ... 40, 50
  VCD ... 75
Spectrum Plot panel ... 80
spin density surface ... 75, 77, 268
spin multiplicity
  keyword ... 205
  setting in Maestro ... 21–22
spin populations, Mulliken ... 64
SQM frequency scaling method ... 70–71
standard functionals, dftname values for ... 209
standard output
  keywords ... 258–259
  Maestro options ... 82–85
standard state conversion for SM6 solvation ... 61
structure, input—*see* geometry input
structure, optimizing—*see* geometry optimization
structures
  selecting for IRC calculation ... 112
  selecting for transition state optimization 102
submitting jobs—*see* running jobs
superblocks ... 248
surfaces
  displaying in Maestro ... 78
  generating data for ... 77, 267
symmetrizing geometry input ... 23–24, 345
symmetry
  effect on structure ... 345
  in IRC calculations ... 112, 229
  keywords ... 206
  Maestro options ... 33, 103, 111
  orbital populations ... 57
  output information ... 132
  use of in calculation ... 23–24
synchronous transit quasi-Newton methods—*see* QST-guided transition-state searches

T

TDDFT calculations ... 39, 216
TDHF calculations ... 50, 216
temperatures for thermochemical calculations
  keywords ... 243
  output ... 157
  setting ... 74
temporary directory ... 344
  errors related to ... 343–344
  in output file ... 132
  saving at end of job ... 317
temporary files
  saving at end of job ... 317
Theory tab ... 39, 51
thermal smearing ... 57, 250
thermochemical properties ... 74
  keywords ... 243
  output ... 157
time stamps in log file, option for ... 317
`timex` program ... 290
timing information
  keyword ... 258
  Maestro option ... 83
torsional angles
  freezing all ... 218
  in Z-matrix ... 15
transition metals
  improving convergence ... 56, 168–170
  initial guess for ... 167–168, 169, 281
Transition State tab ... 101
transition vector ... 102, 229
transition-state optimization ... 91–105
  constraining bond lengths or angles ... 96–99
  convergence criteria ... 93, 226
  coordinate system ... 95
  eigenvector following ... 104, 222
  frozen bond lengths or angles ... 96–99
  in solution ... 60, 217
  initial Hessian ... 19–20, 94–95, 104–105, 199–200, 223
  keywords ... 217–226
  level shifting of Hessian ... 223
  limiting step size for ... 224
  maximum iterations ... 93, 218
  refinement of initial Hessian ... 19–20, 104–105, 199–200, 223
  search method ... 100
  trust radius ... 224–225
  updating of Hessian ... 223
trial wave function—*see* initial guess

troubleshooting 341–346
trust radius for optimizations 224–225, 230

U

units
    electrostatic potential 205
    for geometry 98, 198, 203
    keyword 204
    output options 84
    polarizability, hyperpolarizability............ 239
unrestricted wave functions
    keyword for.. 246
    setting for 38, 50
utilities
    `jagconvert` 319
    `makejbasis` 297
UV/visible spectra...................................... 40, 50

V

van der Waals radii... 275
    for ESP fitting 64
    for solvation calculations......................... 231
    input file sections for 273–275
    listed in output 142
van der Waals surface 64
variables in geometry input........... 13–14, 16–17, 96–99, 200–201
VCD spectra.................................... 75, 119, 333
VCD Workflow panel 119
vibrational circular dichroism... 74, 119, 242, 333
    spectra... 75
vibrational frequencies.................................... 70
    keywords for 223, 241–242
    scaling... 70–71, 243
    visualizing in Maestro 72
    visualizing with Molden 73
vibrational spectra.. 73
Vibrations panel 72
virial ratio.. 258
virtual orbitals
    keyword for number printed.................... 261
    number printed 87, 136

W

working directory.. 323

X

XYZ file (`.xyz`) output option 86

Z

zero-point energies........................... 74, 157, 243
Z-matrix format 15–17, 198–201
    dummy atoms in 17
    variables in 16–17, 17, 200–201
ZORA Hamiltonian 53, 190, 272

Keyword Index

Numerics

2spin ... 275

A

allow_scf_failure ... 247
atguess ... 251
auto_scale ... 243

B

babel ... 254–255
babelg ... 254–255
basgss ... 283
basis ... 244–245, 281

C

cfiterr ... 237, 240
charge ... 281
check_min ... 220
check_min_amide ... 221
check_min_eigcut ... 220
check_min_ergtol ... 220
check_min_geotol ... 221
check_min_hb_lim ... 221
check_min_pert ... 221
check_min_return ... 220
check_min_target ... 221
check_min_update ... 220
cov ... 275
covfac ... 84, 205
cpoise ... 215
cut20 ... 132, 251

D

daf ... 281
dcoarse ... 265
dconv ... 226, 240, 248, 250
dconvci ... 216
decut1 ... 249
decut2 ... 249
den_isoval ... 239
denspc ... 238, 240
dfine ... 265
dftname ... 208–212
dgrad ... 265
dmedium ... 265
dryaposteri ... 215
dufine ... 265

E

econv ... 226, 240, 248, 250
econvci ... 216
efield ... 238, 240
epn ... 239
epsin ... 232
epsout ... 232, 235
esolv0 ... 232
esp ... 275
espanal ... 239, 241
espunit ... 205
etransfer ... 244
eunit ... 98, 203, 204, 241

F

fall_through ... 215
fdtemp ... 249
formal ... 275
frag ... 281
freqcut ... 243
freqfrag ... 242, 281
fukui ... 239
fukui_delta ... 269
fukui_fn ... 269

G

gcharge ... 240, 265, 310
gcoarse ... 265, 310
gconv1-gconv7 ... 226
gdftcphf ... 263
gdftder2 ... 263
gdftfine ... 263
gdftgrad ... 263, 310
gdftmed ... 263, 310
geldens ... 240, 265, 310
geoconv_mode ... 219
gfine ... 265, 310
ggrad ... 265, 310
glmp2 ... 265
glmp2grad ... 265
gmedium ... 265, 310
grid ... 281
guess_thresh ... 252

gufine 265, 310

H

hirshfeld_basis 239
huge_diis 249

I

iacc 246
iaccg 219, 226
iacscf 246
icanorb 249
icavity 231
icdft 215
icfit 237
ichange 248
icis 216
iconv 246
icpfrag 281
idelfrag 281
idelocv 207
idenavg 215, 248
idfgrdX 263
idoabe 206
idynmem 265
ielec 239, 240
ifdtherm 249
iflinks 260
ifollow 222
ifreq 242
igeopt 217
igonly 180, 251
iguess 252, 283
ihamtyp 205, 246, 252, 286
iheter 207, 208
ihuptyp 223
ilagr 219
imw 243
incdip 237, 240
inewguess 252
inhess 223, 282
intopt 201, 218
iorb1a 268
iorb1b 268
iorb2a 268
iorb2b 268
iordboy 254
ioss 252
ip1, ip3-ip8, ip11-ip13, ip18-ip19, ip24-ip26, ip170, ip173, ip192-ip193 258–259
ip100 262
ip100-107 262
ip105 262, 263
ip107 254
ip15, ip17, ip110, ip121-ip123, ip149, ip188, ip201 261
ip151 175, 259
ip152 175, 261
ip160 180, 181, 260
ip165 260
ip170 138, 259
ip29 274
ip472 218, 228
ip90, ip160-ip161, ip163-ip165, ip168, ip172, ip175 260
ipka_csrch_acc 236
ipka_erg_window 236, 363
ipka_lower_cut 235
ipka_max_conf 236, 363
ipka_prot_deprot 235
ipka_solv_opt 236
ipka_tier_cutoff 236
ipka_upper_cut 236
ipkaraw 236, 362
ipkasearch 235
ipkashell 359
ipkasites 235, 363
ipkasymm 352
ipkat 362
ipkaverbose 236, 352
iplotalie 241, 269
iplotden 268
iplotesp 268
iplotnoncov 269
iplotspn 268
ipolar 237, 238, 240
ipopsym 206
ipvirt 261
iqcoarse 266
iqfine 266
iqgrad 266, 267
iqlmp2 267
iqlmp2grad 267
iqmedium 266

iqst ... 207, 222
iqufine ... 266, 267
iraman ... 242
irc ... 228
ircgcut ... 228
ircmax ... 228
ircmode ... 228
ircmxcyc ... 228
ircstep ... 228
irder ... 242
irefhup ... 223
ireson ... 207
irfo ... 223
isolv ... 231
isolvg ... 231
isotope ... 275
isotrymax ... 232
isqm ... 243
istate ... 252
istavg ... 248
isurf ... 231
isymm ... 206, 240
itda ... 216
itddft ... 216
iteravg ... 247
itradj ... 224
itrcut ... 224
itrvec ... 222
itwice ... 248
iuhf ... 246
iunit ... 98, 198, 203, 204, 267, 269, 284
iusediffuse ... 247
ivanset ... 231
ivcd ... 242
ixtrboy ... 253–254

J

jksep ... 247

K

kesep ... 231, 258

L

large_diis ... 248
lastwv ... 248
lcoarse ... 265
ldens ... 238, 240
ldips ... 238
lewstr ... 231
lfine ... 265
lgrad ... 265
lmedium ... 265
loclmp2c ... 207
loclmp2v ... 207
locpostc ... 254
locpostv ... 254
lufine ... 265

M

mass ... 275
massav ... 206
maxciit ... 217
maxdiis ... 247, 250
maxit ... 246
maxitcp ... 243
maxitg ... 218, 219, 249
molchg ... 205, 285
mp2 ... 207
mulk ... 275
mulken ... 238
multip ... 205, 275
mxpr ... 266
mxstrip ... 265

N

nbcmax ... 266
nbo ... 239
nboden ... 291
nbuck ... 266
ncanorb ... 249
ncicut_high ... 269
ncicut_low ... 269
ndfgrdX1 ... 263
ndfgrdX2 ... 263
ndisk ... 266
needgwd ... 219
negrid ... 239, 240
newcon ... 246
ngeorestart ... 219
nhesref ... 223
nmder ... 70, 73, 218, 241
nmr ... 238

noabortscf 247
noatcor 247
noauto 248
nofail 249
nogas 217
nogdiis 219, 221
noopta 218
nooptr 218
nooptt 218
nops 247, 310
nops_opt_switch 219
nosuper 248
noupdat 247
nrestart 217
nroot 216
ntemp 243
numd 36, 245

P

pbf_boundary_iso 232
pbf_boundary_res 232
pbf_fem_mxres_cg 232
pbf_isodens 232
pbf_molsurf_idx 232
pbf_molsurf_iso 232
pbf_nshells 231
pbf_protosph_res 232
pbf_radii 232
pdftname 213
pertnd 218, 223
plotres 269
plotresnoncov 269
potential 272
press 243
printaposteri 215
props_each_step 218

Q

qstinit 222

R

radprb 232
relham 208
res_boundary 232
res_inner 232
res_molsurf 232
res_outer 232
retryscf 247
rmscp 243
rsinglet 216
rtriplet 216

S

scalfr 243
scan_independent 227
scangeo 227
scanguess 227
scanhess 227
sconv 232
sm8_sola 235
sm8_solb 235
sm8_solc 235
sm8_solg 235
sm8_solh 235
sm8_soln 235
SMARTS patterns 311
solvent 231
stdiis 248, 250
stockholder_q 239
stop_rxn 225
stop_rxn_epsilon 226
stop_rxn_kval 226
stop_rxn_offset 226
stop_rxn_sigma 226

T

three_pt_irc 228
tmpini 243
tmpstp 243
tradmn 224
tradmx 224
tremx 224
treok 224
trescal 224
trgmx 224
trust 224, 230

U

ulg_blg 215
ulg_slg 215

V

vdw 275
vdw2 275
vshift 215, 246, 249, 250

W

wispc 237, 240

X

xadj 269
xcorl1-xcorl4 .... 213, 214
xcornl1-xcornl19 214
xcornl1-xcornl9 213
xexl1 213
xexl9 213
xexnl1-xexnl9 213
xhf 213
xmaxadj 269
xminadj 269

Y

yadj 270
ycorl1-ycorl4 213
ycornl1-ycornl9 213
yexl1 213
yexl9 213
yexnl1-yexnl9 213
yhf 213
ymaxadj 269
yminadj 270

Z

zadj 270
zmaxadj 270
zminadj 270
zmpmem 266