

# Jaguar 5.5

---

*User Manual*

Copyright © 2003 Schrödinger, L.L.C. All rights reserved.  
Schrödinger, FirstDiscovery, Glide, Impact, Jaguar, Liaison, LigPrep, Maestro, Prime, QSite, and  
QikProp are trademarks of Schrödinger, L.L.C.  
MacroModel is a registered trademark of Schrödinger, L.L.C.  
To the maximum extent permitted by applicable law, this publication is provided “as is” without  
warranty of any kind. This publication may contain trademarks of other companies.

October 2003

---

# Contents

---

<b>Chapter 1: Introduction</b> .....	<b>1</b>
1.1 Conventions Used in This Manual.....	2
1.2 Citing Jaguar in Publications.....	3
<b>Chapter 2: The Maestro Graphical User Interface</b> .....	<b>5</b>
2.1 Starting Maestro.....	5
2.2 The Maestro Main Window.....	7
2.3 Maestro Projects.....	7
2.4 Building a Structure.....	9
2.5 Atom Selection.....	10
2.6 Toolbar Controls.....	11
2.7 Mouse Functions.....	14
2.7.1 Mouse Functions in the Workspace.....	14
2.7.2 Mouse Functions in the Project Facility.....	15
2.8 Shortcut Key Combinations.....	16
2.9 Undoing an Operation.....	17
2.10 Maestro Command Scripts.....	17
2.11 Specifying a Maestro Working Directory.....	18
2.12 Running and Monitoring Jobs.....	19
2.13 Help.....	20
2.14 Ending a Maestro Session.....	21
<b>Chapter 3: Running Jaguar From Maestro</b> .....	<b>23</b>
3.1 Sample Calculation.....	23
3.2 Molecular Structure Input.....	26
3.2.1 Entering or Editing a Geometry Using the GUI.....	26
3.2.2 Cartesian Format for Geometry Input.....	28
3.2.3 Variables in Cartesian Input.....	28
3.2.4 Constraining Cartesian Coordinates.....	29
3.2.5 Z-Matrix Format for Geometry Input.....	29
3.2.6 Variables and Dummy Atoms in Z-Matrix Input.....	31
3.2.7 Constraining Z-Matrix Bond Lengths or Angles.....	32
3.2.8 Counterpoise Calculations.....	32
3.2.9 Specifying Coordinates for Hessian Refinement.....	32
3.3 Charge and Multiplicity (State).....	33

---

3.4	Reading Files .....	34
3.4.1	Reading in Geometries Only .....	34
3.4.2	Reading in Geometries and Job Settings.....	35
3.4.3	Read as Geometry 2 or Geometry 3 Settings .....	35
3.5	Cleaning up Molecular Geometries .....	36
3.5.1	Quick Geometry Optimization .....	36
3.5.2	Symmetrization .....	37
3.6	Running Jobs.....	38
3.6.1	Starting Individual Jobs.....	38
3.6.2	Running Batch Jobs or Scripts .....	40
3.7	Saving Input Files .....	44
3.8	Output .....	45
3.9	Other Maestro Features.....	46
3.9.1	Checking Jobs With the Monitor Panel.....	46
3.9.2	The Reset Option.....	46
3.9.3	Editing Input.....	46
3.9.4	The About and Help Buttons.....	47
3.9.5	Closing the Jaguar Panel .....	48
3.9.6	Other Jaguar Panel Options.....	48
<b>Chapter 4:</b>	<b>Options .....</b>	<b>49</b>
4.1	Density Functional Theory (DFT) Settings .....	49
4.1.1	Stage and Grid Density .....	50
4.1.2	DFT Model Options .....	51
4.1.3	Custom Functionals.....	53
4.2	Local MP2 Settings.....	54
4.2.1	Summary of the LMP2 Method in Jaguar.....	54
4.2.2	Setting up an LMP2 Calculation .....	55
4.3	Generalized Valence Bond (GVB) Settings.....	56
4.3.1	GVB or GVB-RCI Pair Input.....	57
4.4	GVB-LMP2 Calculations .....	57
4.5	Solvation .....	58
4.5.1	Solvent Parameters.....	59
4.5.2	Performing or Skipping a Gas Phase Optimization .....	60
4.6	Properties .....	60
4.6.1	Electrostatic Potential Fitting.....	60
4.6.2	Multipole Moments.....	62
4.6.3	Polarizability and Hyperpolarizability .....	62
4.6.4	Electron Density.....	63

---

4.6.5 Mulliken Population Analysis .....	64
4.6.6 Natural Bond Orbital (NBO) Analysis .....	64
4.7 Frequencies and Related Properties .....	64
4.7.1 Frequencies .....	65
4.7.2 Atomic Masses .....	66
4.7.3 Scaling of Frequencies .....	66
4.7.4 Animation of Frequencies .....	67
4.7.5 Infrared Intensities .....	69
4.7.6 Thermochemical Properties .....	69
4.8 Basis Set .....	70
4.9 Methods .....	74
4.9.1 Wavefunction Type (Restricted or Unrestricted) .....	74
4.9.2 Selecting Electronic States .....	75
4.9.3 Choosing an Initial Guess Type .....	76
4.9.4 Convergence Issues .....	77
4.9.5 Accuracy Level .....	78
4.9.6 Analytic Corrections .....	78
4.9.7 Final Localization of the Orbitals .....	79
4.9.8 Symmetry .....	79
4.10 Surfaces .....	79
4.11 J2 Theory Calculations .....	81
<b>Chapter 5: Optimizations and Scans .....</b>	<b>83</b>
5.1 Geometry Optimization: The Basics .....	83
5.1.1 Maximum Iterations .....	83
5.1.2 Geometry Convergence Issues .....	84
5.1.3 The Initial Hessian .....	85
5.1.4 Trust Radius .....	85
5.2 Constraining Coordinates .....	86
5.2.1 Freezing All Bond Lengths, Bond Angles, or Torsional Angles .....	86
5.2.2 Freezing Specific Coordinates .....	86
5.2.3 Applying Constraints by Using Variables .....	87
5.2.4 Applying Dynamic Constraints .....	88
5.3 Transition State Optimizations .....	88
5.3.1 Transition State Search Method .....	89
5.3.2 Specifying Different Structures for the Reaction .....	90
5.3.3 Initial LST Guess .....	90
5.3.4 Searching Along a Particular Path or Eigenvector .....	91
5.3.5 Eigenvector Following .....	91

---

5.3.6 Refinement of the Initial Hessian .....	92
5.3.7 Specifying Coordinates for Hessian Refinement .....	93
5.4 Geometry Scans .....	93
5.5 Intrinsic Reaction Coordinate Calculations .....	95
<b>Chapter 6: Output .....</b>	<b>97</b>
6.1 Summarizing Jaguar Results.....	97
6.1.1 Reporting Final Results From One or More Jobs .....	100
6.1.2 Reporting Intermediate Results .....	101
6.1.3 Reporting Results for Each Atom .....	102
6.2 Output From a Standard HF Calculation .....	102
6.3 Output File Content for Calculation Options.....	106
6.3.1 DFT .....	106
6.3.2 LMP2.....	106
6.3.3 GVB .....	107
6.3.4 GVB-RCI .....	108
6.3.5 Geometry or Transition State Optimization (HF, GVB, DFT, and LMP2) 109	
6.3.6 Optimizations With GVB-RCI Wavefunctions .....	112
6.3.7 Solvation.....	112
6.3.8 Geometry Optimization in Solution .....	116
6.3.9 Properties.....	116
6.3.10 Frequency, IR Intensity, and Thermochemistry Output .....	121
6.3.11 Basis Set.....	123
6.3.12 Methods.....	123
6.4 Standard Output Options .....	124
6.5 File Output Options .....	129
6.6 Output Options Per Iteration.....	131
6.7 Output Options for Orbitals .....	133
6.8 The Log File.....	136
<b>Chapter 7: Tips and Suggestions.....</b>	<b>139</b>
7.1 Tips for Various Types of Jobs.....	139
7.1.1 Organometallics and Other Difficult-to-Converge Systems.....	139
7.1.2 GVB Calculations: GVB Pair Selection.....	141
7.1.3 Geometry Optimization.....	141
7.1.4 Electrostatic Potential Charge Fitting .....	142
7.2 Restarting Jobs and Using Previous Results .....	142

---

7.3	Suggestions for GAUSSIAN 9x Users.....	143
7.3.1	Generating GAUSSIAN 9x Input Files With Jaguar.....	143
7.3.1.1	Making Input Files for GVB Calculations.....	144
7.3.1.2	Other Jaguar Options for the .g92 File.....	144
7.3.2	Getting Basis Sets or Orbitals for GAUSSIAN 9x.....	145
7.3.3	Using GAUSSIAN 9x Files as Jaguar Input.....	145
<b>Chapter 8:</b>	<b>Theory.....</b>	<b>147</b>
8.1	The Pseudospectral Method.....	147
8.2	Pseudospectral Implementation of the GVB Method.....	149
8.3	GVB-RCI Wavefunctions.....	153
8.4	Pseudospectral Local MP2 Techniques.....	156
8.5	Density Functional Theory.....	159
<b>Chapter 9:</b>	<b>The Jaguar Input File.....</b>	<b>161</b>
9.1	General Description of the Input File.....	161
9.1.1	Sections Describing the Molecule and Calculation.....	162
9.2	The <b>zmat</b> , <b>zmat2</b> , and <b>zmat3</b> Sections.....	164
9.3	The <b>zvar</b> , <b>zvar2</b> , and <b>zvar3</b> Sections.....	166
9.4	The <b>coord</b> and <b>connect</b> Sections.....	166
9.5	The <b>gen</b> Section.....	168
9.5.1	Geometry Input Keywords.....	168
9.5.2	Molecular State Keywords (Charge and Multiplicity).....	169
9.5.3	Atomic Mass Keyword.....	169
9.5.4	Symmetry-Related Keywords.....	169
9.5.5	GVB and Lewis Dot Structure Keywords.....	170
9.5.6	LMP2 Keywords.....	172
9.5.7	DFT Keywords.....	173
9.5.8	CIS Keywords.....	179
9.5.9	Geometry Optimization and Transition State Keywords.....	179
9.5.10	Intrinsic Reaction Coordinate (IRC) Keywords.....	185
9.5.11	Solvation Keywords.....	187
9.5.12	Properties Keywords.....	188
9.5.13	Frequency-Related Keywords.....	191
9.5.14	Basis Set Keywords.....	193
9.5.15	Keywords for SCF Methods.....	193
9.5.16	Initial Guess Keywords.....	198
9.5.17	Localization Keywords.....	200
9.5.18	File Format Conversion Keywords.....	201
9.5.19	Standard Output Keywords.....	204

9.5.20	File Output Keywords .....	206
9.5.21	Output Keywords for Each Iteration .....	207
9.5.22	Orbital Output Keywords .....	208
9.5.23	Grid and Dealiasing Function Keywords .....	210
9.5.24	Memory Use Keywords.....	212
9.5.25	Plotting Keywords .....	214
9.6	The <b>gvb</b> Section .....	216
9.7	The <b>imp2</b> Section.....	217
9.8	The <b>atomic</b> Section.....	218
9.8.1	General Format of the <b>atomic</b> Section .....	218
9.8.2	Keywords That Specify Physical Properties .....	220
9.8.3	Basis, Grid, Dealiasing Function, and Charge Usage for Individual Atoms 221	
9.8.4	Defining Fragments .....	225
9.9	The <b>hess</b> Section .....	226
9.10	The <b>guess</b> Section .....	227
9.11	The <b>pointch</b> Section .....	229
9.12	The <b>efields</b> Section.....	229
9.13	The <b>ham</b> Section.....	230
9.14	The <b>orbman</b> Section.....	230
9.15	The <b>echo</b> Section .....	231
9.16	The <b>path</b> Section .....	231
9.17	The <b>plot</b> Section.....	234
9.18	NBO Sections .....	236
<b>Chapter 10:</b>	<b>Other Jaguar Files.....</b>	<b>237</b>
10.1	The Basis Set File .....	237
10.1.1	Format of the Basis Set File .....	237
10.1.2	Customizing Basis Sets .....	241
10.2	The Initial Guess Data File .....	242
10.3	The Dealiasing Function File.....	243
10.3.1	File Format and Description.....	244
10.3.2	Sample File.....	247
10.4	The Grid File.....	248
10.4.1	File Format and Description.....	249
10.5	The Cutoff File.....	252
10.6	The Lewis File .....	253
10.6.1	Describing Bonding Types in the Lewis File .....	255



---

10.6.2	Describing Hybridization Types in the Lewis File.....	256
10.6.3	Setting van der Waals Radii From Lewis File Data .....	258
10.6.4	Default Behavior for Setting Radii.....	262
<b>Chapter 11:</b>	<b>Running Jobs.....</b>	<b>263</b>
11.1	Customizing Host Configurations.....	263
11.1.1	The name and host Settings.....	265
11.1.2	The user Setting.....	265
11.1.3	The tmpdir Setting.....	265
11.1.4	The processors Setting .....	266
11.2	The jaguar Command .....	266
11.2.1	Selecting a Calculation Host .....	269
11.2.2	Selecting Particular Jaguar Executables.....	269
11.2.3	Running a Jaguar Job From the Command Line.....	269
11.2.4	Killing a Jaguar Job.....	271
11.2.5	Converting File Formats.....	272
11.3	Running Multiple Jobs: jaguar batch.....	275
11.3.1	Batch Input File Format .....	276
11.3.2	Batch Input File Example.....	279
11.3.3	Running jaguar batch.....	279
<b>Chapter 12:</b>	<b>Troubleshooting.....</b>	<b>281</b>
12.1	Problems Getting Started .....	281
12.1.1	The SCHRODINGER Environment Variable .....	281
12.1.2	Including the jaguar Command in Your Path.....	282
12.1.3	Problems Starting Maestro.....	283
12.1.4	Problems Related to Your Temporary Directory .....	284
12.1.5	Problems Running Jaguar Calculations on Other Nodes .....	285
12.2	Other Problems .....	286
<b>Chapter 13:</b>	<b>Parallel Jaguar .....</b>	<b>289</b>
13.1	Installing Parallel Jaguar.....	289
13.1.1	SGI Installation .....	290
13.1.2	LINUX Installation .....	291
13.1.2.1	Installing MPICH .....	291
13.1.2.2	Configuration.....	291
13.1.2.3	Launching the Secure Servers .....	293
13.1.2.4	Selecting Nodes for a Job.....	295
13.1.2.5	Troubleshooting Parallel Job Problems .....	295
13.1.3	IBM Installation .....	296
13.2	Running Jobs in Parallel .....	298

<b>Chapter 14: The pKa Prediction Module</b> .....	<b>299</b>
14.1 Introduction .....	299
14.2 Theory of pKa Calculation .....	300
14.2.1 Ab initio Quantum Chemical Calculation of pKa Values .....	300
14.2.2 Empirical Corrections .....	302
14.3 Predicting pKa Values in Complex Systems .....	303
14.3.1 Conformational Flexibility .....	303
14.3.2 Equivalent Sites .....	304
14.3.3 Multiple Protonation Sites .....	305
14.4 Training Set Results .....	305
14.5 Running pK <sub>a</sub> Calculations .....	315
14.5.1 Activating the pKa Module .....	315
14.5.2 Jaguar Input Files for pKa Calculations .....	315
14.5.3 Running pKa Calculations .....	316
14.5.4 Monitoring pKa Calculations .....	317
14.5.5 Initial Geometry .....	318
<b>Chapter 15: Getting Help</b> .....	<b>319</b>
<b>References</b> .....	<b>321</b>
<b>Index</b> .....	<b>329</b>
<b>Keyword Index</b> .....	<b>345</b>

---

# Chapter 1: Introduction

---

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 graphical user interface (GUI). Online help is available in the GUI, although the information in this manual is generally more comprehensive.

The GUI for Jaguar is part of the Maestro GUI. [Chapter 2](#) introduces the main features of Maestro and provides instructions for setting up your environment and running Maestro.

[Chapter 3](#) 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 3.1](#). 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 4](#) and [5](#) 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 4](#) includes information on using generalized valence bond (GVB), restricted configuration interaction (RCI), Møller-Plesset second-order perturbation theory, and density functional theory (DFT) techniques; calculating solvation energies, vibrational frequencies, hyperpolarizabilities, multipole moments, and other properties; fitting charges; specifying basis sets; and various other options. [Chapter 5](#) describes optimizations of the molecular structure, transition state searches, and geometry scans.

[Chapter 6](#) describes how to summarize Jaguar output and the output or printing options available from the GUI. 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 92.

[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 GVB, GVB-RCI, and 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 configuring hosts, the environment, and the Schrödinger software to run Jaguar, submitting jobs from the command line, and running multiple Jaguar jobs using batch scripts.

[Chapter 12](#) contains troubleshooting hints concerning various problems you might encounter, especially when first setting up Jaguar on your system. [Chapter 13](#) contains information on running calculations on parallel computers. [Chapter 14](#) describes the pKa calculation module.

## 1.1 Conventions Used in This Manual

In addition to the normal use of italics for names of documents, the font conventions that are used in this manual are summarized in [Table 1.1](#).

*Table 1.1. Font Conventions*

Font	Example	Use
Sans serif	Project Table	Names of GUI features such as panels, menus, menu items, buttons, labels
Monospace	<code>\$SCHRODINGER/maestro</code>	File names, directory names, commands, and environment variables
Italics	<i>filename</i>	Text that the user must replace with a value
Sans serif uppercase	ALT+H	Keyboard keys
Bold	<b>atomic</b>	Input keywords
Bold italic	<b><i>0</i></b>	Default value of a keyword. Used in tables of keyword values

In descriptions of command syntax, the usual UNIX conventions are used: braces { } enclose a choice of required items, square brackets [ ] enclose optional items, and the pipe symbol | separates items in a list from which one item must be chosen.

References to literature sources are given in square brackets, like this: [13]. The reference list begins on [page 321](#).

Superscripts in the text correspond to footnotes that list the Jaguar input file entries that correspond to particular GUI settings. You can ignore the footnotes if you like, but you may find them useful for setting up files to run calculations without using the GUI, or for interpreting the input file.

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

## 1.2 Citing Jaguar in Publications

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

Jaguar 5.5, Schrödinger, L.L.C., Portland, OR, 1991-2003.



---

# Chapter 2: The Maestro Graphical User Interface

---

Maestro is the graphical user interface for all of Schrödinger's products: FirstDiscovery (Glide, Impact, Liaison, and QSite), Jaguar, MacroModel, Prime, and QikProp. It contains tools for building, displaying, and manipulating chemical structures, for organizing, loading, and storing these structures and associated data, and for setting up, monitoring, and visualizing the results of calculations on these structures. This chapter provides a brief introduction to Maestro and some of its capabilities. For more information, see the *Maestro User Manual*.

Most Maestro panels are amodal: more than one panel can be open at a time, and a panel need not be closed for an action to be carried out. Instead of a Close option in one of the panel menus, each Maestro panel has a Hide button so that you can hide the panel from view.

## 2.1 Starting Maestro

In order to launch Maestro, you must first set the `SCHRODINGER` environment variable to point to the installation directory. You can set this variable by entering the following command at a shell prompt:

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

You might also need to set the `DISPLAY` environment variable if it is not set automatically when you log in. To determine if you need to set this variable, enter the command

```
echo $DISPLAY
```

If the response is a blank line, set the variable by entering the appropriate version of the following command:

```
csh/tcsh:    setenv DISPLAY display-machine-name:0.0
sh/bash/ksh: export DISPLAY=display-machine-name:0.0
```

After you set the `SCHRODINGER` and `DISPLAY` environment variables, you can launch Maestro using the command

```
$SCHRODINGER/maestro
```

If the `§SCHRODINGER` directory has been added to your path, you can simply enter the command `maestro`. Options for this command are given in the *Maestro User Manual*.

The directory you were in when you launched Maestro is the current working directory, and all data files are written to and read from this directory unless otherwise specified (see [Section 2.11 on page 18](#)). You can change directories by entering the following command in the main window's command input area:

```
cd directory_name
```

where *directory\_name* is either a full path or a relative path.

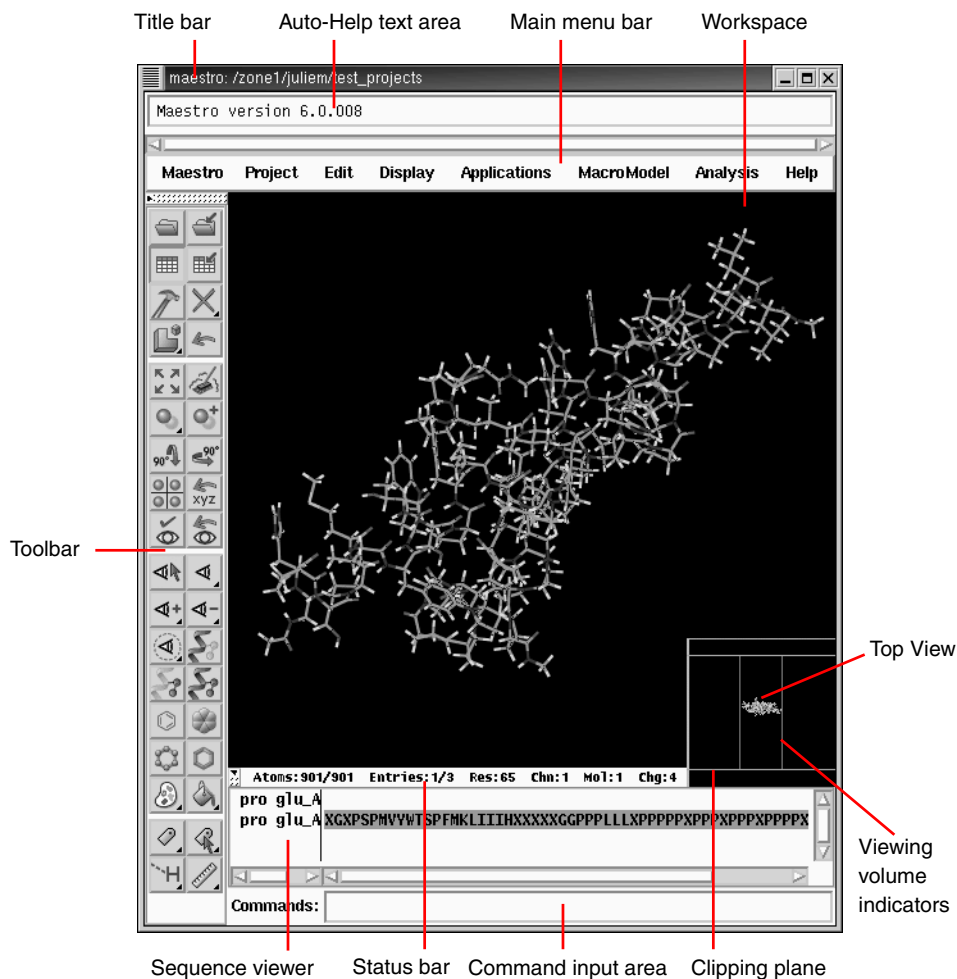


Figure 2.1. The Maestro main window.



## 2.2 The Maestro Main Window

The Maestro main window is shown in [Figure 2.1 on page 6](#). This window is composed of the title bar, the Auto-Help window, the main menu bar, the Workspace, the top view area, the toolbar, the command input area, and the status bar. You can display any of the last five of these using the Display menu.

The functions of the main window components are as follows:

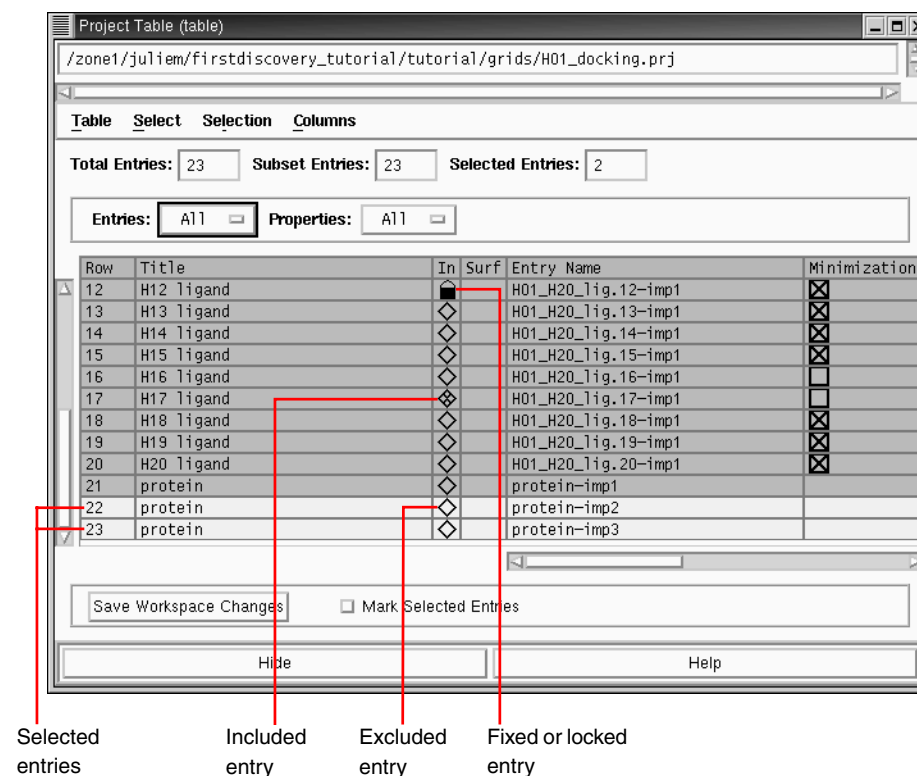
- **Title bar**—displays the project name and current working directory
- **Auto-Help**—automatically displays context-sensitive help
- **Main menu bar**—provides access to panels via pull-down menus
- **Workspace**—displays molecular structures
- **Top view**—displays Workspace structures as viewed from above
- **Toolbar**—contains shortcuts for many menu items, and also tools for structure display and manipulation, and organization of the Workspace
- **Status bar**—displays the number of atoms, entries, residues, chains, and molecules currently displayed in the Workspace
- **Sequence viewer**—displays the sequences for proteins currently displayed in the Workspace
- **Command input area**—provides a way to execute Maestro commands

## 2.3 Maestro Projects

The *project* is a central concept in Maestro. A project is a collection of chemical structures (*entries*) and their associated data. These structures and their data are represented in the Project Table, which displays an ordered list of entries and any associated data. You can open the Project Table panel by choosing Show Table from the Project menu.

There is always a project open in Maestro. If you do not specify a project when you start Maestro, a *scratch* project is created. You can work in a scratch project without saving it, but you must save it in order to use it in future sessions.

Entries are represented by rows in the Project Table. Each row contains the row number, the title, the entry's Workspace inclusion state (the In column), a button to open the Surfaces panel if there are surfaces associated with the entry, the entry name, and any properties associated with the entry. If there are no surfaces associated with any entry, the Surf column of the Project Table is empty.



**Figure 2.2. The Project Table panel.**

You can choose the entries to include in the Workspace using the In column of the Project Table panel. To include a single entry and exclude all other entries, click in the In column for that entry. To include a range of entries, hold down SHIFT and click the first and last entries. To choose individual entries for inclusion (or for exclusion if they are already included), hold down CTRL and click the entries. Including an entry causes any structures associated with that entry to be displayed in the Workspace. You can also fix, or lock, entries in the Workspace by selecting the entries and choosing Selection > Fix in Workspace or pressing ALT+F. A padlock icon is displayed in the In column to denote fixed entries. To remove a fixed entry, exclude it from the Workspace. Fixed entries are not affected by the inclusion or exclusion of other entries. There are shortcuts for selecting classes of entries on the Select menu and in the Select panel. You can open the Select panel from the Select menu.

You can use entries as input for all of the computational programs—Glide, Impact, Jaguar, Liaison, MacroModel, Prime, QikProp, and QSite. You can select entries as input for the ePlayer, which displays the selected structures in sequence. You can also duplicate, combine, rename, and sort entries, create properties, import structures as entries, and export structures and properties from entries in various formats.

When you build molecules in the Workspace, they constitute the *scratch* entry until you save the structures as project entries. The scratch entry is not saved with the project unless you explicitly incorporate it into the project. However, you can use a scratch entry as input for some computations.

## 2.4 Building a Structure

After you start Maestro, the first task is usually to create or import a structure. You can open existing Maestro projects or import structures from other sources to obtain a structure. To build a structure, you use the Build panel, which you can open by clicking the hammer icon in the toolbar, or by choosing Build from the Edit menu.

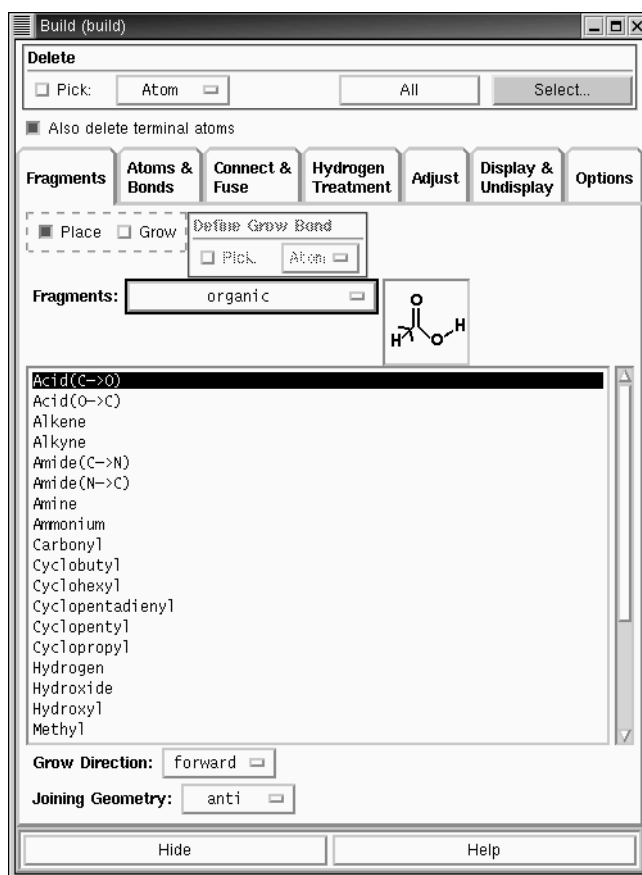


Figure 2.3. The Build panel.

The Build panel allows you to create structures by placing atoms or fragments in the Workspace and connecting them into a larger structure, to adjust atom positions and bond orders, and to change atom properties. The Fragments folder offers a variety of molecular fragments from which to build a structure. To place a fragment in the Workspace, select the fragment and click in the Workspace where you want the fragment to be placed. You can place several fragments in the Workspace and connect them using the Connect & Fuse tab, you can attach a fragment to a structure by selecting it and clicking on an atom in the Workspace, or you can place a fragment and *grow* another fragment onto it by defining a *grow bond* and then selecting a fragment. The new fragment replaces the atom at the head of the arrow on the grow bond and all atoms attached to it. To draw a structure freehand, use the Draw feature of the Atoms & Bonds tab.

Once you have a structure, you can adjust the bond lengths, bond angles, and dihedral angles using the Adjust tab. In the Atoms & Bonds tab, you can change the bond orders, retype atoms, invert structures around chiral centers, and make adjustments on individual atoms. In the Properties tab, you can set properties such as charges, partial charges, and names of structural units.

## 2.5 Atom Selection

Maestro has a powerful set of tools for selecting atoms in a structure that takes advantage of chemical information about the structure. These tools are embedded in each panel in which you might need to select atoms to apply some operation. Once you have decided which operations to apply, you can select, or *pick*, the atoms to which you want to apply the chosen operation using the tools provided.

If you want to select all atoms in a molecule, a chain, a residue, or an entry, you can choose a *pick state* using the Pick list. Once you have chosen the pick state, you can click on an atom in the Workspace, and all the atoms that belong to the same structural unit, as defined by the pick state, are selected. For example, if you choose Residue and click on any atom in a glycine residue, all the atoms in that glycine residue are selected. To select individual atoms, choose Atoms from the Pick list. The Pick list varies from panel to panel, because not all pick states are appropriate for a given operation. For example, some panels have only Atoms and Bonds in the Pick list.

If you want to make atom selections based on more complex criteria, such as selecting all the carbon atoms in a protein backbone, you can use the Atom Selection dialog box. To open this dialog box, click Select. You can select an atom grouping from any of the tabs in the dialog box: Atom, Residue, Molecule, Chain, Entry, Substructure Notation, or Set. You can then combine this grouping with another grouping using the buttons on the right: the Add button (Boolean OR) includes the current selection with the existing selection, the Remove button (Boolean AND NOT) excludes atoms in the current selection from the existing selection, and the Intersect button (Boolean AND) includes only those atoms that

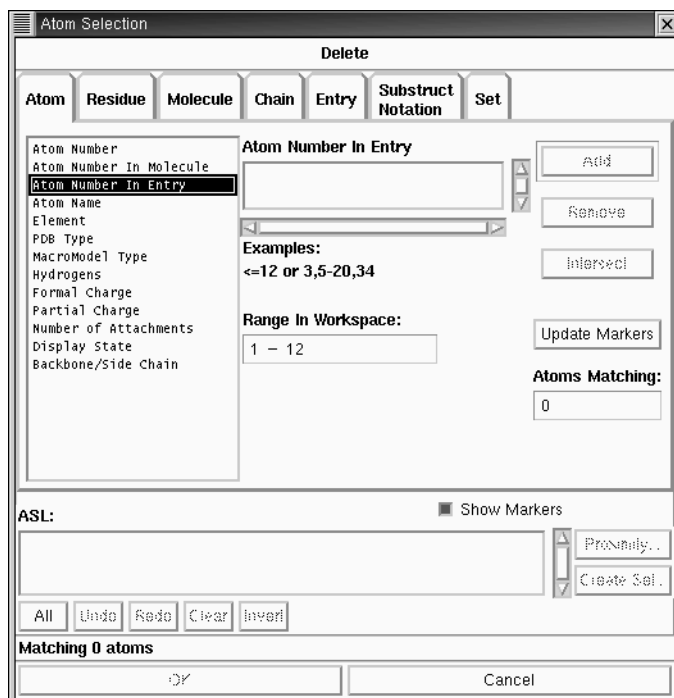


Figure 2.4. The Atom Selection dialog box.

are in both the current selection and the existing selection. The existing selection is expressed in ASL language in the ASL text box, and is shown with light blue markers in the Workspace. The current selection is shown with purple markers.

When you are satisfied with the selection, click OK to apply the operation you have chosen with the selection you have made. The operation is described in a bar at the top of the Atom Selection dialog box (ASD). Some operations, such as Delete, take effect immediately. Others merely define a set of atoms to be used in a subsequent task, such as selecting atoms for creation of a surface.













While the Atom Selection dialog box is open, you cannot perform other actions, with the exception of interacting with the structures displayed in the Workspace, such as rotation, translation, and picking.





















## 2.6 Toolbar Controls

The toolbar contains buttons for performing common tasks. There are several kinds of buttons on the toolbar. Some buttons perform simple tasks like Fit to Screen or Clear Workspace. Other buttons open panels such as the Import panel. Buttons that have a small

triangle in the lower right corner have menus that you can open by holding down the left mouse button. Some of the toolbar buttons retain the last menu choice, and you only need to click them to use that choice. Some buttons have different actions depending on whether you click them or double-click them.

You can show or hide the toolbar using the collapse button at the top or from the Display menu, and you can hide it or move it to the right or left of the workspace by right-clicking in the toolbar and selecting the appropriate option. The buttons are described below.

<p><b>Open a project</b> Displays the Open Project dialog box.</p>	 	<p><b>Import structure(s)</b> Displays the Import panel.</p>
<p><b>Show/Hide Project Table</b> Displays the Project Table panel or hides it if it is displayed.</p>	 	<p><b>Create entry from Workspace</b> Creates an entry in the current project using the contents of the Workspace.</p>
<p><b>Show/Hide Build panel</b> Displays the Build panel or hides it if it is displayed.</p>	 	<p><b>Delete</b> Displays a three-section menu from which you can choose an object to delete. The first section is a pick menu for deleting atoms by picking. If you select one of these items, the delete button is indented to show that you are picking to delete atoms. The pick state is persistent. The second section opens the Atom Selection dialog box to define atoms to delete. The third section allows you to delete other objects associated with the structures in the Workspace.</p>
<p><b>Local transformation</b> Displays a menu from which you can select the object for local transformation, or open the Advanced Transformation panel. The selection is persistent.</p>	 	<p><b>Undo/Redo</b> Undoes or redoes the last action. Performs the same function as the <b>Undo</b> item on the <b>Edit</b> menu, and changes to an arrow pointing in the opposite direction when an <b>Undo</b> has been performed, indicating that its next action is <b>Redo</b>.</p>
<p><b>Fit to screen</b> Scales what is displayed so it fits into the Workspace</p>	 	<p><b>Clear Workspace</b></p>
<p><b>Set fog display state</b> Displays a menu from which you choose an item. <b>Automatic</b> means <i>on</i> when there are more than 40 atoms in the Workspace, <i>off</i> when there are fewer.</p>	 	<p><b>Enhance depth cues</b> Optimizes fogging and other depth cues based on what is in the Workspace.</p>

Rotate around X-axis by 90 degrees			Rotate around Y-axis by 90 degrees
<b>Tile entries</b> Arranges entries in a rectangular grid in the Workspace.			<b>Reset Workspace</b> Resets the rotation, translation, and zoom of the Workspace to the default state.
<b>Save view</b> Saves the current view of the Workspace: orientation, location, zoom.			<b>Restore view</b> Restores the last saved view of the Workspace: orientation, location, zoom.
<b>Display only this molecule</b> Picks the molecule to display. Double-click to display all atoms.			<b>Display only</b> Displays a menu from which you can select a category of atoms to display, or open the Atom Selection dialog box to select atoms to display.
<b>Also display</b> Displays a menu from which you can select a category of atoms to add to the display, or open the Atom Selection dialog box to select atoms to add to the display.			<b>Undisplay</b> Displays a menu from which you can select a category of atoms to undisplay, or open the Atom Selection dialog box to select atoms to undisplay.
<b>Display atoms within N angstroms of currently displayed atoms</b> Displays a menu from which you can select a value, or open a dialog box to enter a value. The value selected is persistent.			<b>Display ribbons only</b> Displays ribbons for protein atoms.
<b>Display ribbon atoms only</b> Displays protein atoms used to define ribbons.			<b>Display both ribbons and atoms</b> Displays ribbons along with the protein atoms used to define them.
<b>Draw bonds in wire representation</b> Double-click to apply to all bonds.			<b>Draw atoms in CPK representation</b> Double-click to apply to all atoms.
<b>Draw atoms in Ball &amp; Stick representation</b> Double-click to apply to all atoms.			<b>Draw bonds in tube representation</b> Double-click to apply to all bonds.
<b>Color all atoms by scheme</b> Displays a menu from which you can choose the scheme.			<b>Color by constant color</b> Displays a menu from which you can choose the color. The color choice is persistent. You can then pick molecules to apply the color, or double-click the button to apply the color to all atoms.

#### Label atoms

Displays a menu from which you can choose a single type of label to apply to all atoms.



#### Label picked atoms

Displays a menu from which you can choose a pick state, open the **Atom Selection** dialog box to select the atoms, or open the **Atom Labels** panel with the **Composition** tab displayed to select the composition of the label. The picking choice is persistent. Double-clicking applies the label to all atoms.

#### Display H-bonds

Displays a menu from which you can choose to display H-bonds within the selected molecule (intra) or between the selected molecule and all other atoms in the Workspace (inter). The choice is persistent.



#### Measure distances, angles or dihedrals

Displays a menu from which you can choose between distance, angle, or dihedral. The choice is persistent. The default is distance. After clicking this button, pick atoms in the workspace to define the measurement.

## 2.7 Mouse Functions

The mouse functions common to graphical user interfaces are supported in Maestro. The left button is used for selecting: choosing menu items, clicking buttons, and selecting objects. This button is also used for resizing and moving panels. Other common mouse functions, such as combinations with the **SHIFT** or **CTRL** keys, are used in some contexts for selecting a range of items and selecting or deselecting a single item without affecting other items.

### 2.7.1 Mouse Functions in the Workspace

The Workspace has special uses for the middle and right mouse buttons. These are used alone and in combination with the **SHIFT** and **CTRL** keys to perform common operations such as rotation, translation, centering, and zooming. Apart from centering a molecule on an atom, all these operations involve dragging.

If you have the handedness on your mouse set to “left,” the mouse functions are the mirror image of those described: the right mouse button is used for picking, and the left button is used for translating. If you have a two-button mouse, make sure that it is configured for three-button mouse simulation. Then the middle mouse button is simulated by pressing or holding down both buttons.



Table 2.1. Mapping of Workspace Operations to Mouse Buttons and Keyboard Keys

Operation	Keyboard/Mouse Combination
Rotation about the $x$ - or $y$ -axis	Middle mouse button
Rotation about the $x$ -axis only	SHIFT and middle mouse button with vertical movement
Rotation about the $y$ -axis only	SHIFT and middle mouse button with horizontal movement
Rotation about the $z$ -axis	CONTROL and middle mouse button with horizontal movement
Spot-centering on an atom	Right-click
Translation in the $x$ - $y$ plane	Right mouse button
Translation along the $y$ -axis	SHIFT and right mouse button with vertical movement
Translation along the $x$ -axis	SHIFT and right mouse button with horizontal movement
Translation about the $z$ -axis	CONTROL and right mouse button with horizontal movement
Zoom	Middle and right mouse buttons or SHIFT+CONTROL and middle mouse button with horizontal movement

## 2.7.2 Mouse Functions in the Project Facility

The standard combinations of SHIFT and CTRL keys with a click to select objects are supported in the Project Table. To select a range of entries, click the first entry, hold down the SHIFT key, and click the last entry in the range. To select or deselect an entry without affecting the selection of other entries, hold down the CTRL key and click the entry. Similarly, to include or exclude individual entries or a range of entries from the Workspace, use the SHIFT and CTRL keys in combination with mouse clicks.

Once you have selected project entries, you can manipulate them using tools available from the Selection menu. Clicking the right mouse button while the pointer is positioned over the Project Table (so long as there is at least one entry in the table) displays the Selection menu at the mouse position.

You can use the mouse to perform operations besides selecting entries. Clicking and dragging with the middle mouse button on the boundary of a row resizes that row. Clicking on selected entries and dragging allows you to reposition the entry in the Project Table. The entries are placed after the first unselected entry that precedes the entry on which the cursor is resting when you release the mouse button. For example, if you select entries 2, 4, and 6, and release the mouse button on entry 3, these three entries are placed after entry 1, because entry 1 is the first unselected entry that precedes entry 3. To move entries to the top of the table, drag them above the top of the table; to move entries to the end of the table, drag them below the end of the table.

A summary of project-based mouse functions is provided in [Table 2.2](#).

*Table 2.2. Mouse Operations in the Project Table*

<b>Task</b>	<b>Mouse Operation</b>
Select or include entry, deselect or exclude all others	Click
Display Selection menu	Right-click
Select or include multiple entries	Shift-click
Toggle entry selection or inclusion	Control-click
Move entry	Drag

## 2.8 Shortcut Key Combinations

Some frequently used operations have been assigned shortcut key combinations. The shortcuts, their functions, and their GUI equivalents are listed in [Table 2.3](#).

*Table 2.3. Shortcut Keys in the Maestro GUI*

<b>Keys</b>	<b>Action</b>	<b>Equivalent GUI Operation</b>
<i>In the Main Window:</i>		
ALT+B	Show build panel	Edit > Build
ALT+C	Create entry	Project > Create Entry From Workspace
ALT+E	Show script panel	Edit > Command Script Editor
ALT+H	Show help panel	Help > Help
ALT+I	Show import panel	Project > Import Structures
ALT+M	Show measurement panel	Analysis > Measurement
ALT+N	New project	Project > New Project
ALT+O	Open project	Project > Open Project
ALT+P	Print	Maestro > Print
ALT+Q	Quit	Maestro > Quit
ALT+S	Show sets panel	Analysis > Sets
ALT+T	Show project table panel	Project > Show Table
ALT+W	Close project	Project > Close Project
ALT+Z	Undo/Redo last command	Edit > Undo/Redo

Table 2.3. Shortcut Keys in the Maestro GUI (Continued)

Keys	Action	Equivalent GUI Operation
<i>In the Project Table Panel:</i>		
ALT+A	Select all entries	Select > Select All
ALT+F	Fix entry in Workspace	Selection > Fix In Workspace
ALT+I	Show import panel	Table > Import Structures
ALT+N	Include only selected entries	Selection > Include Only These In Workspace
ALT+U	Unselect all entries	Select > Unselect All
ALT+X	Exclude selected entries	Selection > Exclude From Workspace
<i>In the Plot Panel:</i>		
ALT+A	Select all	Plot > Select All
ALT+U	Unselect all	Plot > Unselect All
ALT+P	Show plot settings panel	Settings > Plot Settings

## 2.9 Undoing an Operation

To undo a single operation, click the Undo icon in the toolbar, choose Undo from the Edit menu, or use the key combination ALT+Z. The word Undo in the menu is followed by some text that describes the operation or operations to undo. Not all operations can be undone: global rotations and translations are not undoable operations, for example. However, you can use the Save view and Restore view buttons in the toolbar to save a molecular orientation and restore it at a later time.

If you know in advance that you might want to undo a series of operations, you can start an undo block by selecting Begin Undo Block from the Edit menu. When you have completed the group of operations you want to undo, end the block by selecting End Undo Block from the Edit menu. Then, to undo the operations in the block, choose Undo from the Edit menu. Undo is not supported for all Maestro operations. An undo block will be created only if at least one undoable operation has been performed since the Begin Undo Block command was issued.

## 2.10 Maestro Command Scripts

Although there is a provision for performing nearly all Maestro-supported operations through menus and panels, the operations can also be performed using Maestro commands, or compilations of these commands, called scripts.

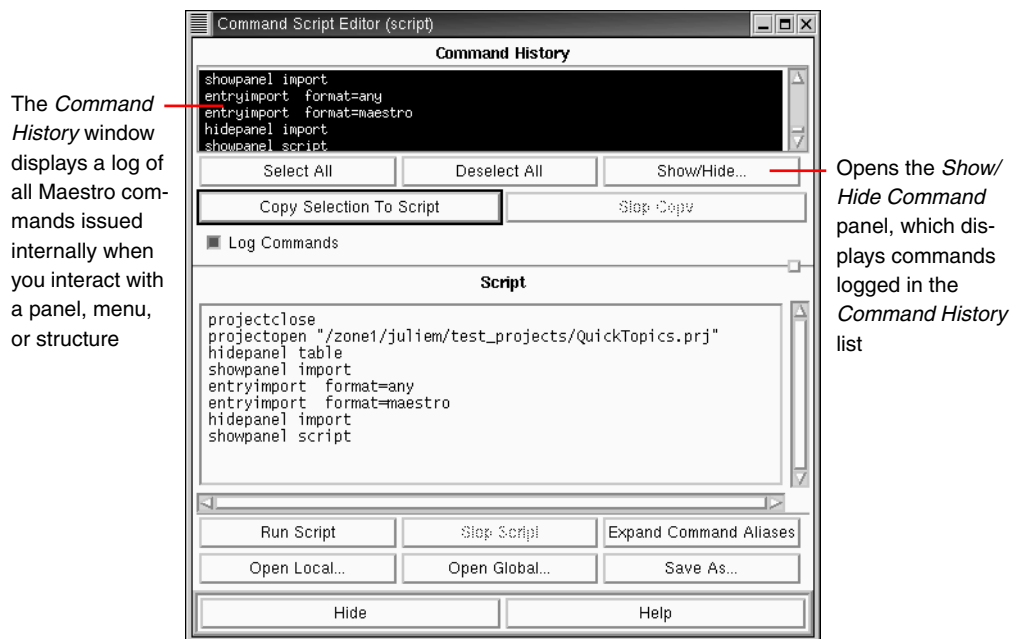


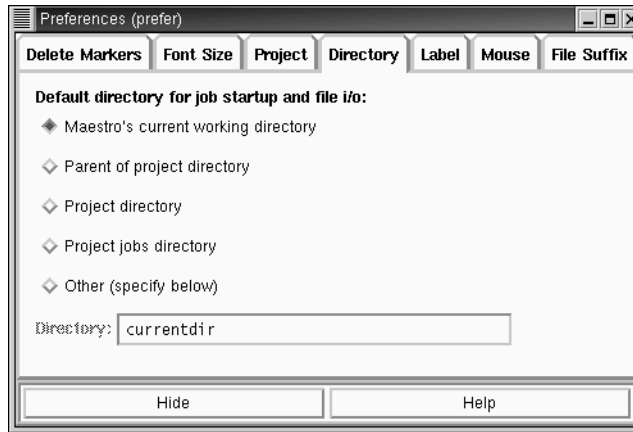
Figure 2.5. The Command Script Editor panel.

Command scripts can be used to automate lengthy procedures or repetitive tasks. Because all Maestro commands are logged and displayed in the Command Script Editor panel, it is not necessary to memorize the commands for every operation or to be familiar with the command language. Simply perform the desired operations using the GUI controls, copy the logged commands from the Command History list into the Script area of the panel, and then save the list of copied commands as a script.

To run an existing command script, open the Command Script Editor panel from the Edit menu in the main window. Click *Open Local* and navigate to the directory containing the desired script. Select a script in the Files list, and then click *Open*. The command script is loaded into the Script window of the Command Script Editor panel. To execute the script, click *Run Script*. Command scripts cannot be used for Prime operations.

## 2.11 Specifying a Maestro Working Directory

When you use Maestro to launch Jaguar jobs, Maestro writes job output to the directory specified in the *Directory* folder of the *Preferences* panel. By default, the directory to which Maestro writes files (the file i/o directory) is the directory from which you launched Maestro. To write the output files to another directory, change the preferences as described below.



*Figure 2.6. The Directory folder of the Preferences panel.*

1. Open the Preferences panel from the Maestro menu.
2. Click the Directory tab.
3. Select the option for the directory you want files to be read from and written to.

## 2.12 Running and Monitoring Jobs

While Jaguar jobs can be run from the command line, we suggest that you use the Maestro GUI to set up and launch these jobs, at least until you have some experience with the programs and understand the directory structure and the input file requirements. Maestro has dedicated panels for preparing and submitting Jaguar jobs. To use these panels, choose Jaguar from the Applications menu.

Maestro also has a job control panel for monitoring the progress of jobs and for pausing, resuming, or killing jobs. All jobs that belong to your userid can be displayed in the Monitor panel. The text pane shows some kinds of output from the job that is being monitored. The Monitor panel opens automatically when you start a job. If it is not open, you can open it by choosing Monitor from the Applications menu in the Maestro main window. You can monitor jobs from this panel whether or not they were started from Maestro.

When a job that is being monitored ends, results from that job are automatically incorporated into the project. If a job that is not currently being monitored ends, you can select it in the Monitor panel and incorporate the results. Monitored jobs are incorporated only if they are part of the project. You can monitor jobs that are not part of the project, but their results are not incorporated. To add their results to the project, you must import them.

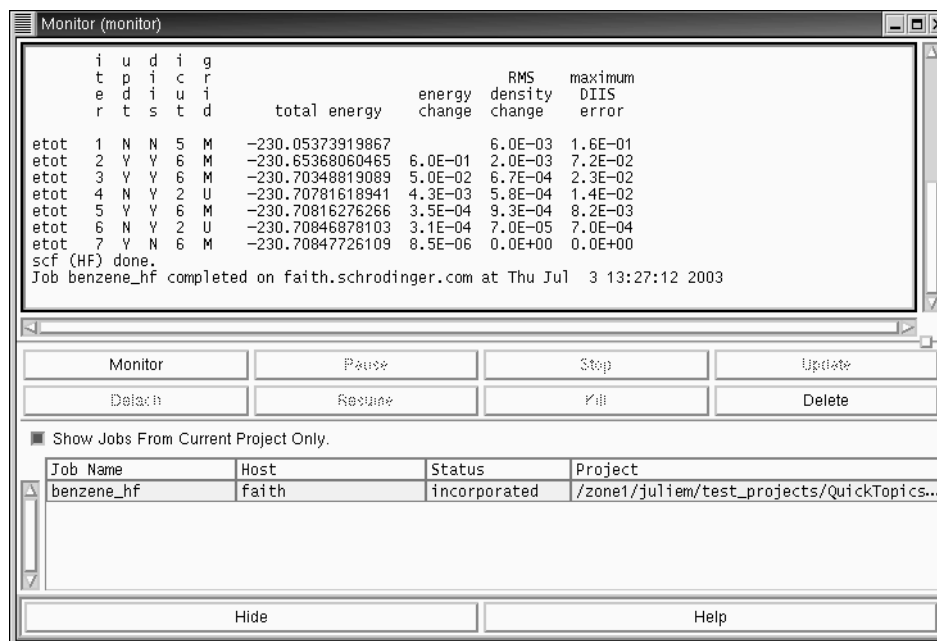


Figure 2.7. The Monitor panel.

## 2.13 Help

Maestro comes with automatic context-sensitive help (Auto-Help), Balloon help (tool-tips), an online help facility, and a user manual. To get help, follow the steps below:

- Check the Auto-Help text box located below the title bar of the main window. If help is available for the task you are performing, it is automatically displayed there. It describes what actions are needed to perform the task.
- If your question concerns a GUI element, e.g., a button or option menu, there may be Balloon help for the item. Move the mouse pointer over the element. If there is Balloon help for the element, it appears within a few seconds.
- If you do not find the help you need using the steps above, click the Help button in the lower right corner of the panel for whose settings you are seeking help. The Help panel is displayed with a relevant help topic.
- For help with a concept or action not associated with a panel, open the Help panel from the Help menu or use the key combination ALT+H.

If you do not find the information you need in the Maestro help system, check the following sources:

- The *Maestro User Manual*
- The *Maestro Release Notes*
- The Frequently Asked Questions page, found at <http://www.schrodinger.com/Support/faq.html>

If you do not find answers to your questions in any of these places, contact Schrödinger using the information below.

Schrödinger

E-mail: [help@schrodinger.com](mailto:help@schrodinger.com)

USPS: 1500 SW First Ave. Suite 1180, Portland, OR 97201

Phone: (503) 299-1150

Fax: (503) 299-4532

WWW: <http://www.schrodinger.com>

FTP: <ftp://ftp.schrodinger.com>

Generally, e-mail correspondence is best because you can send machine output, if necessary. When sending e-mail messages, please include the following information, most of which can be obtained by entering `$SCHRODINGER/machid` at a command prompt:

- Purchaser of the software
- Primary user of the software
- Platform type
- Jaguar version number
- mmshare version number
- Maestro version number
- Operating system with version number

## 2.14 Ending a Maestro Session

To end a Maestro session, choose Quit from the Maestro menu. To save a log file with a record of all operations performed in the current session, click Quit, save log file in the Quit panel. This information can be useful to Schrödinger support staff when responding to any problem you report.





---

## Chapter 3: Running Jaguar From Maestro

---

The Jaguar panel in the Maestro GUI can simplify the submission of jobs. You can run the GUI and the Jaguar calculation on different machines. In addition, as with any X program, the machine running the GUI (the X client) does not need to be the machine or terminal that displays the GUI (the X server). This means that from any X terminal or workstation running X, you can log on to a machine where Maestro is installed and submit jobs on another machine on which the Jaguar executables are installed.

Without the GUI, you would have to create input files with particular formats in order to run Jaguar. The GUI creates these input files for you, based on information you give it, and submits the job. This frees you from learning the input format and program sequences and instead allows you to concentrate on the science involved. The GUI also provides a convenient method of incorporating other data, such as molecular geometries produced by modeling packages.

Try the sample calculation in [Section 3.1](#) to get some experience running Jaguar and to make sure your system is set up properly. If you have problems starting or using the GUI 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 using the GUI, including entering a geometry and submitting a job. The footnotes describe 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 later 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 using the Edit Job window.

### 3.1 Sample Calculation

This section provides instructions on running a sample calculation on the water molecule. The sample calculation works only if Jaguar has been correctly installed. If the calculation does not work, 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. Change to the directory where you want the Jaguar output files for the sample job to be written, then start Maestro by entering the command

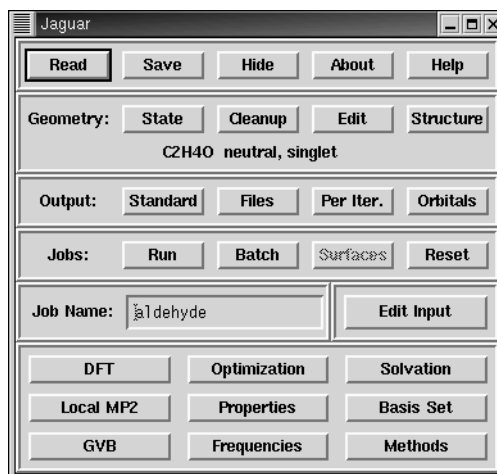


Figure 3.1. The Jaguar panel.

```
$SCHRODINGER/maestro
```

If `$SCHRODINGER` is in your `$PATH`, you can simply type `maestro`. Once Maestro is running, choose Jaguar from the Applications menu to open the Jaguar panel.

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 use the Edit Geometry window. Click the Edit button, then click in the text entry area in the Edit Geometry window, 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 different items. When you finish entering the water geometry, choose Save from the File menu to save your changes, then choose Close from the File menu to close the Edit Geometry window.

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

```
$SCHRODINGER/jaguar-vversion/samples
```

where *version* is the 5-digit 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 Maestro main window.

If you entered the geometry by hand, you must give the job a name by entering a single word in the Job Name text box in the Jaguar panel. If you read in the sample input file, the name H2O appears in this box. The names of the input, output, and log files for your job depend on your entry: 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 your Job Name entry.

When you finish setting up your calculation, click the Run button in the Jaguar panel. The Jaguar Run window is displayed. The calculation host (the machine that the job will run on) is listed at the top of the window. If Jaguar is installed on more than one machine at your site, you can change the choice of calculation host by selecting another name in the list. The Temp directory selection is a directory on the calculation host that will be used during the calculation to store temporary files. You should check that the directory already exists on the calculation host. If it does not exist, you should create it.

If you read the input file from the samples directory, the Job Directory is set to `$SCHRODINGER/jaguar-vversion/samples`. You must change it to some other directory, because you normally would not have permission to overwrite files in this directory. For this exercise, choose the directory you were in when you launched Maestro.

You should not need to change any other settings. Click the RUN button to start the job. An alert box that contains information about the job is displayed.

After you start the job and dismiss the alert box, the Maestro Monitor panel is displayed. 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 6.8 on page 136](#) on the log file for more information on this data. You can close the Monitor panel by clicking the Hide button. If you want to reopen it later, you can do so by choosing Monitor from the Applications menu in the Maestro main window.

When the job finishes, its output file is copied to the directory specified in the Job Directory text box of the Jaguar Run window. If you did not change this directory, it is the directory from which you started the GUI. 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. For this exercise, choose Quit, do not save log file. 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, 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 the GUI. If you were unable to run the sample calculation, see the troubleshooting suggestions in [Chapter 12](#).

If you want to experiment with the sample calculation, use the buttons in the bottom section of the Jaguar panel to open other windows (the DFT window, for example), which you can use to set up a calculation. These possible selections are described in [Chapter 4](#). If you don't change any settings in these windows, Jaguar runs a single-point Hartree-Fock calculation.

## 3.2 Molecular Structure Input

After you start Maestro, the first task for any Jaguar calculation is to enter a molecular structure (geometry).<sup>1</sup> 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 3.4 on page 34](#), or you can enter and edit the geometry yourself using the Edit Geometry window. This section describes how to create or edit a geometry using the Edit Geometry window, and also describes the input formats for Cartesian and Z-matrix geometries.

The geometries that you enter are displayed in the Maestro Workspace, in which you can rotate and translate the structure, edit the geometry, display in various representations, and perform many other tasks. For information on using Maestro, see [Chapter 2](#).

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.

### 3.2.1 Entering or Editing a Geometry Using the GUI

To enter or edit a geometry by hand (or to examine the coordinates), click the Edit button in the Geometry section of the Jaguar panel. The Edit Geometry window is displayed. If you have not read in a geometry file or created a geometry using Maestro, you can type the geometry into the text entry area, or cut and paste the geometry from another text window.

---

1. If you were working directly from an input file without using the GUI, the geometry input would be in the **zmat** and **zvar** sections of the input file.

The Edit menu contains Cut, Copy, and Paste options to facilitate this task. If you already have a geometry, you can use the Edit Geometry window to change it. To clear the entire geometry input, choose Clear from the Edit menu. The geometry can be entered in Cartesian (x,y,z) coordinates or in Z-matrix format. These formats are described below.

You can also alter the geometry input using the Z-matrix menu. The Convert to Z-matrix and Convert to Cartesians options switch between Z-matrix format and Cartesian format. The option Assign standard 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.

To save or remove your changes, or to close the Edit Geometry window, use the File menu. The Save option stores the changed geometry internally but leaves the window open. The changes are not saved to disk until you select OK in the Run window or the Save window. The Close option closes the window. If you select Close and there are unsaved changes, you are asked if you want to save them. The Revert option lets you return to the original geometry (if any) in the window when you opened it, and Cancel closes the window without retaining any changes made since you last saved a geometry.

The options in the Structure menu and the Use initial geometry Z-matrix option in the Z-matrix menu are useful for certain types of transition state optimization jobs, but not for other Jaguar jobs. These options are described in [Section 5.3 on page 88](#).

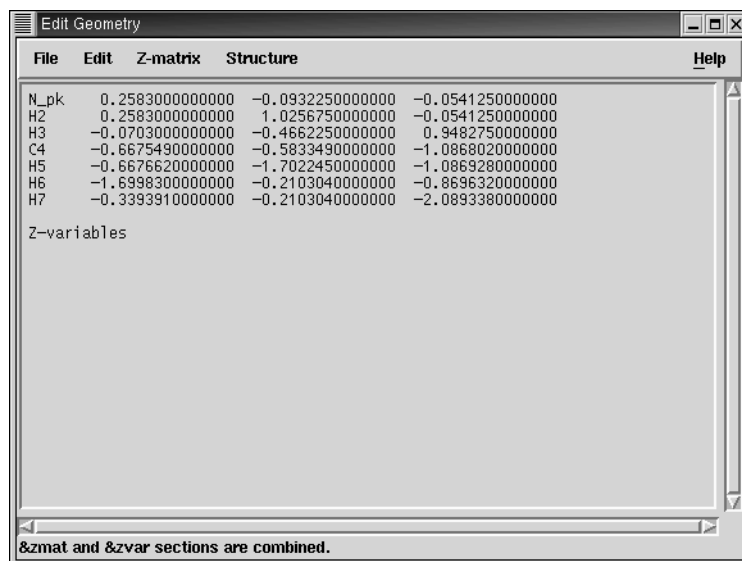


Figure 3.2. The Edit Geometry window.

If you edit a geometry and do not save it, and you try to run a job by clicking RUN in the Run window or save an input file by clicking OK in the Save window, a warning is posted. If you ignore the warning and proceed, the last geometry saved will be used instead of the edited version.

### 3.2.2 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.

### 3.2.3 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

```
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 Cartesian input

```
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
```

is equivalent to the previous Cartesian input example.

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

### 3.2.4 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.

### 3.2.5 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 colinear or very nearly colinear, O4’s position would be poorly defined. You should avoid defining torsional angles relative to three colinear (or nearly colinear) angles. In such a case you should use dummy atoms to define the torsional angle (see [Section 3.2.6 on page 31](#)).

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  $\mathbf{r}_{12}$ .



( $\mathbf{r}_{23} \times \mathbf{r}_{24}$ ), 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  $150^\circ$  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
```

### 3.2.6 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  $\text{CH}_3\text{OH}$  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
```

### 3.2.7 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 type 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 5.2 on page 86](#), which describes how to set up constraints for optimizations.

To constrain two quantities to be the same during a geometry optimization, use variables in Z-matrix input (see [Section 3.2.6 on page 31](#)). 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
```

You should not make any constraint changes from the Edit Geometry window or the Optimization window while both windows are open, because the Optimization settings could conflict with your hand-assigned constraints.

### 3.2.8 Counterpoise Calculations

To perform counterpoise calculations, you can use a Cartesian or Z-matrix geometry that includes counterpoise atoms, which have the usual basis functions for that element but include no nuclei or electrons. These calculations can be useful for obtaining an estimate of basis set superposition error (BSSE). For LMP2 calculations (see [Section 4.2 on page 54](#)), the LMP2 correction is already designed to avoid basis set superposition error, so we advise computing only the Hartree-Fock counterpoise correction term.

To specify a counterpoise atom, place an @ sign after the atom's label. For example, to place sodium basis functions at the Cartesian coordinates (0.0, 0.0, 1.0), you could include the following line in an input file:

```
Na1@ 0.0 0.0 1.0
```

### 3.2.9 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 5.3 on page 88](#) 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 5.3.6 on page 92](#)—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, if you type either of the following two input geometries in the Edit Geometry window:

```
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*
```

they will have the same effect: a job from either input that includes Hessian refinement will use both O–H bonds and the H–O–H angle in the refinement.

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 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.

### 3.3 Charge and Multiplicity (State)

You can set the charge and the spin multiplicity of the molecule in the Molecular State window, which you open by clicking the State button in the Geometry section of the Jaguar panel. The default molecular charge is 0, and the default spin multiplicity is singlet if the molecule has an even number of electrons, and doublet if it has an odd number of electrons.<sup>2</sup> You can change the spin multiplicity to anything up to octet by choosing a value from the Spin Multiplicity list, and to a higher value by choosing other and entering a value in the `multip (2S+1)` text box.<sup>3</sup> The spin multiplicity is always displayed in this text

---

2. Keyword **molchg** in **gen** section of input file.

3. Keyword **multip** in **gen** section of input file.

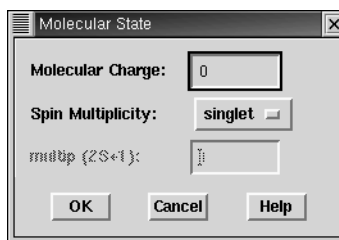


Figure 3.3. The Molecular State window.

box. If the molecular charge and spin multiplicity settings you make do not agree for your molecular input—for instance, if your molecule has an odd number of electrons and you set the spin multiplicity to singlet—Jaguar warns you to reset one or the other.

## 3.4 Reading Files

If you already have files containing geometries (either with or without information on the type of calculation to perform), you can read them using the Read File window, which you open by clicking the Read button in the Jaguar panel.

You can read Jaguar input files or files generated or used by other programs that are recognized by the file format conversion program, Babel [24]. Most of these files can be used to provide only geometries. Files that can provide other information are explicitly identified in the text below.

The Read File window contains 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 in the lists and the filter for the current working directory.

To select a file type, choose from the File Format list. The default format is Jaguar input.

### 3.4.1 Reading in Geometries Only

You can read a geometry (molecular structure) from any of the supported file types. To read only the geometry and set all calculation settings to their default values, choose Geometry (new job) from the Read as list. To read only the geometry and retain the current calculation settings, choose Initial geometry from the Read as list.

When you read in a geometry from a file, Jaguar also tries to obtain information on the molecular charge. This information is always obtained for Jaguar input files, but might not be obtained for other file types. For non-Jaguar input files, check the molecular charge setting in the Molecular State window after reading in the geometry.

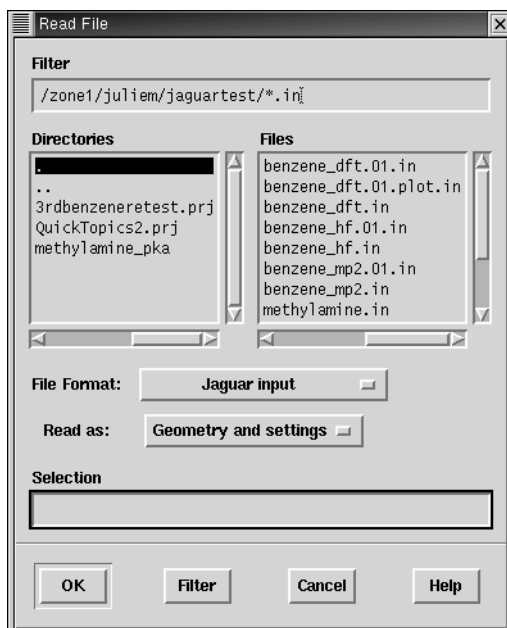


Figure 3.4. The Read File window.

### 3.4.2 Reading in Geometries and Job Settings

The only file types for which Jaguar can read any calculation information *besides* the geometry and molecular charge are Jaguar input files, GAUSSIAN 92 [23] input files, and BIOGRAF [21] Hessian files. To read calculation information, choose Geometry and settings from the Read as option menu.

Geometries and settings read in from a Jaguar or GAUSSIAN 92 input file are displayed in the GUI. Settings that do not have GUI controls are read in even if they are not displayed.

### 3.4.3 Read as Geometry 2 or Geometry 3 Settings

In the Read File window, two options in the Read as option menu, Geometry 2 and Geometry 3, are designed for input only for certain types of transition state optimizations. These options are described in Section 5.3 on page 88, which explains special options for transition state optimizations.

## 3.5 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). Jaguar has options to clean up the geometry for calculations in both of these cases. The options are available from the Geometry Cleanup window, which you open by clicking Cleanup in the Jaguar panel.

The changes made in this window are not applied until you click OK. To check the effect on the geometry, you must close the window and open the Edit Geometry or Edit Input window. If you want to be able to restore the previous geometry after inspecting the new geometry, save the geometry using the Save dialog box before opening the Geometry Cleanup window.

### 3.5.1 Quick Geometry Optimization

When you click Clean up geometry in the Geometry Cleanup window, Jaguar first performs a quick charge-equilibration (Qeq) calculation to obtain partial charges for all atoms in the system, and then uses those charges in an energy minimization, based on Goddard and Rappe's Universal Force Field (UFF). 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, the label on the Clean up geometry button changes to Halt cleanup. Click this button at any time to stop the minimization. After the cleanup is finished, Jaguar reanalyzes the symmetry of the molecule and displays the point group of the minimized structure. If you are satisfied with the results of the cleanup procedure, click OK to accept the geometry from the minimization. Clicking Cancel discards the optimized geometry and reverts to the initial geometry.

The convergence criteria for the cleanup minimization are deliberately set fairly loose, so that even 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

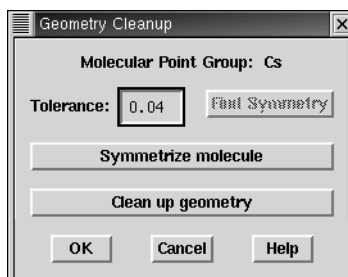


Figure 3.5. The Geometry Cleanup window.

that the geometry continues to change if you perform a second cleanup minimization 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 creating a new molecule from Maestro's Build panel. 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.

### 3.5.2 Symmetrization

By default, Jaguar takes advantage of molecular symmetry<sup>4</sup> 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 using the Symmetrize Molecule button in the Geometry Cleanup window. The point group symmetry that is used is displayed at the top of the window, and is determined by Jaguar 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 Symmetry 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

---

4. Keyword **isymm** = 8 in **gen** section of input file.

main Jaguar programs use a small tolerance ( $1.0 \times 10^{-6}$  bohr), which should result in molecular energy changes of 1 microHartree or less.

If you want, you can turn the use of symmetry off<sup>5</sup> in the Methods window. For some calculations, including GVB, LMP2, GVB-LMP2, and GVB-RCI, and those of IR intensities or hyperpolarizabilities, symmetry is not yet 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 Geometry window to insure that symmetrizing had the desired effect and did not discard any important information about the molecular geometry.

When you symmetrize the molecule, the comment for the job (which is described in [Section 3.6](#) and which appears in the input and output files for the job) includes the note, “Geometry symmetrized to point group,” followed by the point group name.

## 3.6 Running Jobs

You can submit a job either from the GUI or from the command line. You might need to submit jobs from the command line if for any reason you cannot display the GUI on your monitor or terminal. Information on submitting jobs from the command line with the `jaguar run` command can be found in [Section 11.2 on page 266](#). This section describes the submission of jobs from the GUI.

### 3.6.1 Starting Individual Jobs

Once you have read in a geometry, you can submit a Jaguar job by clicking Run in the Jobs section of the Jaguar panel and entering the appropriate information. Before you open the Run window, close any other open windows to save settings you might have changed.

The information you enter in the Run window mainly tells Jaguar how and where to launch a job. The choices available in the Run window option depend on the `schrodinger.hosts` configuration file. See [Section 11.1 on page 263](#) for more information on this file. If you do not change the entries in the Run window, the settings shown are used for the run.

---

5. Keyword `isymm = 0` in `gen` section of input file.



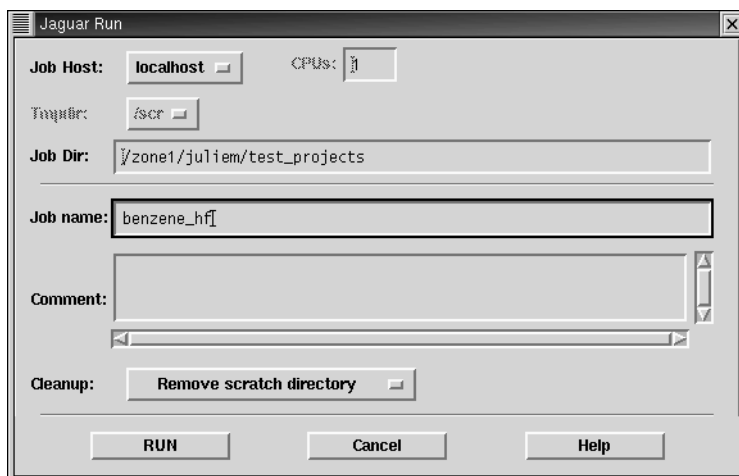


Figure 3.6. The Jaguar Run window.

If Jaguar is installed on more than one host at your location, you can select a host to run the calculation from the Job Host option menu.

A temporary directory on the calculation host is used to store intermediate files during the calculation. If there is more than one possible choice listed for the Tmpdir setting, you should pick one. A subdirectory with the given job name (“h2o,” for example) is created within the temporary directory, and the files from the calculation will be stored in this subdirectory. If the subdirectory and directory do not have sufficient disk space for the job, the job fails.

If your temporary directory does not exist, you should create it, or choose a directory which already exists. If none of the temporary directory choices already exist and you do not want to create the necessary directories, you can change the `schrodinger.hosts` file so that the list offers you different choices (see [Section 11.1 on page 263](#)).

The Job Dir setting is the local directory where input and output files created by Jaguar are written. The default local job directory is the directory from which you read the input file, if you read one, or the directory you were in when you started Maestro. You can change the default selection by editing the directory name.

If the job host you choose is identified in the `schrodinger.hosts` file as having more than one processor, you can run Jaguar in parallel on that host. When you choose a multi-processor host, the # of Processors section becomes active, and the number of processors available is displayed to the right of the text box. To select the number of processors on which to run the job, change the value in the text box. The default is one processor.

The text in the Job name box determines the names of many of the files created by Jaguar, as well as the name of the subdirectory within the temporary directory, which is described

above. The files whose names depend on the job name include the input file, the log file (which shows the job's progress), and the output file listing the calculation results. For instance, if the job name is `h2o`, the results are stored in a file called `h2o.out` within the local job directory.

The default setting for the job name is the base of the input file name, if any, from which the molecular geometry was read. For example, if you read the geometry from a file called `h2o1.in`, the default job name setting would be `h2o1`. You can change the job name in Job name box in either the Run window or the Jaguar panel. If you did not read in the geometry from a file, the job name `Scratch` is supplied.

Any text entered in the box marked Comment appears in the input and output files for the job. If you symmetrize the molecule, a procedure described in [Section 3.5.2 on page 37](#), the comment contains text noting that the geometry was symmetrized to a certain point group. You can enter other text describing the job if you like. *The comment cannot contain the characters \$ or &.* The comment appears in the input file immediately above any keyword settings corresponding to later selections, and in the output file under the heading "Input file comments."

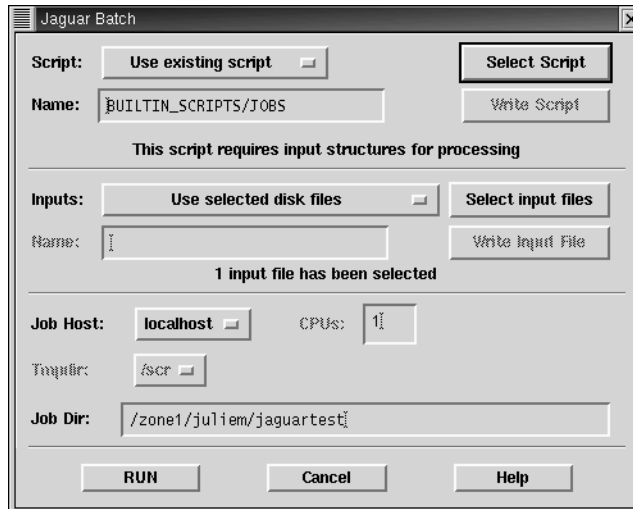
By default, all temporary files and directories are deleted when the job finishes, after the output file, restart file (which is described in [Section 7.2 on page 142](#)), and other useful files are copied back to the local job directory. If you want to save the binary files generated in the temporary directory's job subdirectory and used during the run, use the Cleanup option menu. Note, however, that these files are often large and should be saved only if necessary, and any files in the temporary directory may be deleted automatically if your site has automatic purging of scratch disks.

When you are satisfied with the run-time settings, start the job by clicking RUN. You can then check the current status of the job from the Monitor panel. You can close the Monitor panel by clicking the Hide button. If you want to reopen it later, choose Monitor from the Applications menu in the Maestro main window. Any additional jobs that you submit run concurrently. If you exit Maestro, any Jaguar jobs still running continue to run to completion. For more information on the Monitor panel, see [Section 3.9.1 on page 46](#), or see the *Maestro User Manual*.

### 3.6.2 Running Batch Jobs or Scripts

You can run multiple Jaguar calculations in a single run using the Jaguar Batch window, which you open using the Batch button in the Jobs section of the Jaguar panel. For instance, you can run

- 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



*Figure 3.7. The Jaguar Batch window.*

Several Jaguar batch scripts are included with the program. You can also write your own batch scripts or save job setup in the Jaguar Batch window as a batch script. [Section 11.3 on page 275](#) provides details on batch scripts.

To run a Jaguar batch job from the Jaguar Batch window, you first need a batch script. You can create a batch script from the current job settings (**gen** section keywords) by choosing Use current job settings from the Script menu. You can name the batch script in the Name text box, and write it to disk in the current working directory, by clicking Write Script. A batch script is written to the launch directory using the name in the Name text box when the batch job is launched if you do not explicitly save it.

To use an existing script, choose Use existing script from the Script menu, then click Select Script and choose a script in the Select Batch Script window. The batch script can be in any of three directories:

- The batch script directory installed with Jaguar (identified as BUILTIN\_SCRIPTS)
- Your own Jaguar batch script directory, which can be set in the environment variable JAGUAR\_SCRIPTS, and by default is ~/jaguar\_scripts
- The current directory (the directory containing the last input file you read in or wrote out or, if you have not read or written any files, the directory you were in when you started Maestro)

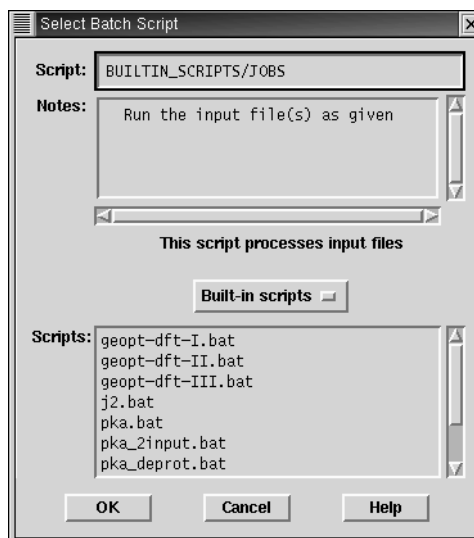


Figure 3.8. The Select Batch Script window.

The default selection in the Select Batch Script window is the built-in JOBS.bat script. The Notes window shows comments from the JOBS.bat script. As they indicate, this script simply runs a series of jobs from the input files it is passed.

By default, the built-in scripts directory is selected. To select one of the other directories listed above, choose User scripts or Local scripts from the option menu in the middle of the window. 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 3.1.

Table 3.1. 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.
j2.bat	Run a J2 theory calculation [25].
pka.bat	Run a pKa calculation. See Chapter 14 for details.

Jaguar batch scripts can either require that you specify an input file (or list of input files) to be run with the “recipe” in the batch file, or they can include a self-contained list of Jaguar input files. The information in the kind of script is displayed in the middle of the Select Batch Script window. If the script you select processes input files, select an input file (or list of files).

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 Inputs option menu in the Jaguar Batch window. If you choose Use current structure from workspace or Use currently selected project entries, the structures are written to a single Maestro file, and the Jaguar input files are created later by the batch facility. The file name can be set in the Name text box in this section of the window.

To select pre-existing files, choose Use selected disk files from the Inputs menu, then click Select input files. The Select Batch Inputs window is displayed, with the current directory and its files listed. You can enter a directory in the Input Dir text box. When you press RETURN, the names of the available input files are displayed in the Files list. You can select either Jaguar input files or Maestro files for input. If you select Maestro files, Jaguar input files are constructed later. You can control whether Maestro (.mae) files and Jaguar restart files (*jobname.xx.in*, where *xx* is a two-digit number) are displayed with the Hide .mae files and Hide restart files options. By default neither is displayed. 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.

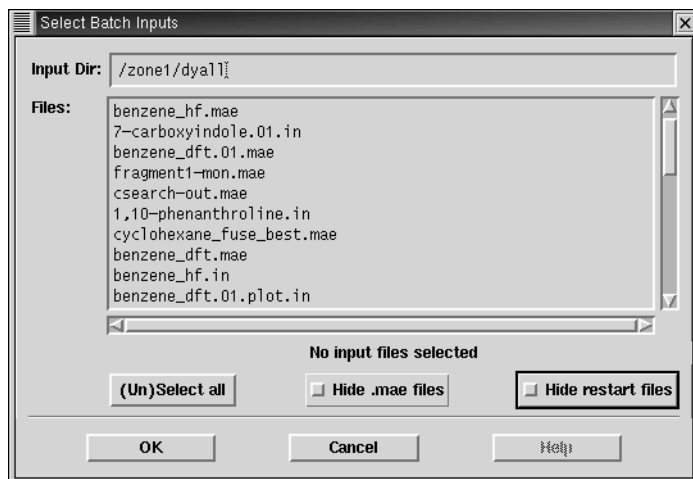


Figure 3.9. The Select Batch Inputs window.

After you finish selecting the batch script and input files, you can choose the host on which to run the job from the Job Host option menu. If the host has more than one processor, you can enter the number of processors to use in the CPUs text box. The number of processors available is displayed to the right of this text box. If you run the batch job on multiple processors, the individual Jaguar jobs are distributed over the processors. Each processor runs only one job at a time. If there are more jobs than processors, the remaining jobs will wait until a processor is available. If you run multiple Jaguar jobs on a single processor, the jobs run sequentially: the next job in the script is not started until the current job has finished.

**Note:** You can distribute batch jobs over multiple processors only if you are using the default batch script or the current settings for the batch script. The exception is pKa jobs, which can be run on two processors, because they consist of two independent jobs. You cannot run MPI parallel jobs as batch jobs.

In addition to choosing a host, you can also choose a scratch directory, if there is more than one defined for the host in the `schrodinger.hosts` file, and you can choose the job directory.

When you are ready, click RUN to launch the batch job. When you click RUN, the Monitor panel opens. 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. You can close the Monitor panel by clicking the Hide button. If you want to reopen it later, choose Monitor from the Applications menu in the Maestro main window.

## 3.7 Saving Input Files

You can use the Save window to store a Jaguar input file or to save a geometry in an appropriate format for another program. You can later read Jaguar input files, as described in [Section 3.4 on page 34](#), and run jobs from the GUI, as described earlier in this section. Alternatively, you can use a Jaguar input file as input for a job submitted from the command line. You must start jobs from the command line if you cannot display the GUI on your monitor or terminal. For information on submitting jobs from the command line, see [Section 11.2 on page 266](#). Jaguar input files can be copied to other machines that have Jaguar installed and used for runs there.

In the Save window, which you open by clicking the Save button in the Jaguar panel, the Input file directory is the directory on the local host in which the file is written. The default input file directory is the directory from which you most recently read a file, if you read one, or the directory you were in when you started Maestro. You can set the directory by typing the name in the Input file directory text box.



Figure 3.10. The Save window.

The Job name text determines the name of the input file created by Jaguar. For instance, if the job name is “h2o” and you save a Jaguar input file, this file is named `h2o.in`. The default setting for the job name is the stem of the input file name, if any, from which the molecular geometry was read. You can change the job name in Job name text box in the Save window or in the Job Name box in the Jaguar panel. If you did not read in the geometry from a file, you should enter a job name in either the Save window or the Jaguar panel.

You can save files in a variety of formats for other programs using the Save as menu. For any file formats other than the Jaguar input (`.in`) file, only the geometry is included in the file. Jaguar uses the Babel program for the file format conversions (see [Section 11.2.5 on page 272](#)). The file name is determined by appending the extension indicated in the file type list to the job name.

Any text entered in the box marked Comment appears in the input file for the job. If you symmetrize the geometry (see [Section 3.5.2 on page 37](#)), the comment notes that the geometry was symmetrized to a certain point group. You can type other text describing the job for your own convenience. *The comment cannot contain the characters \$ or &.* The comment appears in the Jaguar input file immediately above any keyword settings corresponding to other selections.

## 3.8 Output

A Jaguar log file contains comments on the progress of a job. If the job was started from the GUI, the log file is written to the local job directory selected in the Run window. 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 is displayed when a job is launched or when you choose Monitor from the Applications menu in the Maestro main window. See [Section 6.8 on page 136](#) for more information on this file.

The primary Jaguar 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](#).

## 3.9 Other Maestro Features

This section describes some features of Maestro that are not covered elsewhere. Note that sometimes a menu item is dimmed, which means that the option is currently unavailable. For example, the Run button is dimmed until a geometry is entered.

### 3.9.1 Checking Jobs With the Monitor Panel

The Monitor panel allows you to examine Jaguar log files. It opens automatically when you start a job. If you hide the Monitor panel, you can reopen it again later by choosing Monitor from the Maestro Applications menu. See [Section 2.12 on page 19](#) and the *Maestro User Manual* for more information on job control and monitoring.

The log file for the last job you ran (`h2o.log` for the job `h2o`, for example) is displayed automatically in the Monitor panel. The log file indicates when each Jaguar program has finished running. [Section 6.8 on page 136](#) contains more information about this file.

### 3.9.2 The Reset Option

The Reset button in the Jobs section resets many of the settings to the defaults. Clicking Reset clears the geometry and data from any other files read in, as well as all settings describing the wavefunction and properties to be calculated and any settings you may have made using the Geometry or Output buttons. It also sets the Job name value in the Run or Save window to whatever is appropriate when you read or enter the next geometry. However, the other selections you have made in the Run or Save window remain the same. Reset prompts you to confirm or cancel the operation.

### 3.9.3 Editing Input

The Edit Input button in the Job Name section opens the Edit Job window, which allows you to make changes to the entire input file. However, you cannot change between Z-matrix and Cartesian input formats in this window. Geometry input format conversion can be done only in the Edit Geometry window. The editing options in the Edit Job window are the same as in the Edit Geometry window. See [Section 3.2.1 on page 26](#) for more details. Changes you make in the Edit Job window are not saved to disk until you click OK in the Run window or the Save window.



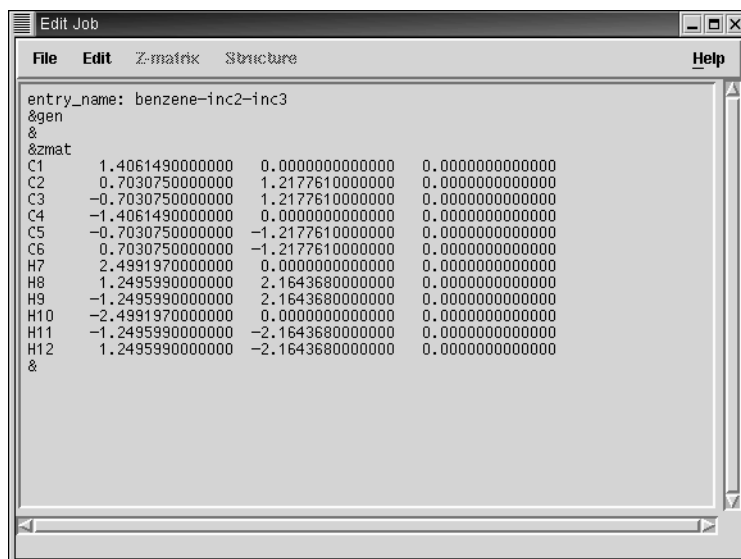


Figure 3.11. The Edit Job window.

You do not need to use the Edit Job window to do anything described in [Chapter 3](#) through [Chapter 6](#). However, if you prefer to set up jobs with keywords, or if you want to use any options described in [Chapter 9](#) that are not set by the GUI, the Edit Job window provides you with a convenient way to do so. To use keywords to set options that can be set by the GUI, refer to the footnotes in [Chapter 4](#) and [Chapter 6](#) to find out which keyword settings are appropriate. If you make and save a setting in the Edit Job window that corresponds to something shown by the GUI, the GUI selection shows the change.

If you select a keyword by double-clicking on it or dragging over it, a brief description is displayed at the bottom of the window if there is information available for the keyword. To see a fuller description, open the online help by clicking Help.

If the input file contains unrecognized input, a warning is displayed that the keyword is unrecognized when you run the job or save the input file. If you click OK, the unrecognized information is retained in the input for the job, and could cause your job to fail.

### 3.9.4 The About and Help Buttons

The About button displays information about Jaguar and Schrödinger. You must close the window before using other parts of the GUI.

The Help button opens the Help window. You can see on-line help on a variety of subjects by clicking on them as they are listed under the Help items heading so that they show up in the Selection bar, then clicking Select. You can also obtain help for any window by

clicking the Help button. The Help window appears with the appropriate topic selected. All of the information in the on-line help is also included in this manual.

### **3.9.5 Closing the Jaguar Panel**

Clicking Hide hides, or closes, the Jaguar panel. Hiding the panel merely means that it is hidden from view—none of the settings are lost, and jobs started from the panel continue to run.

### **3.9.6 Other Jaguar Panel Options**

Most of the settings that control the choice of method or theoretical model are described in the next chapter. The Output buttons, for requesting additional information in output files, are described in [Chapter 6](#).

---

## Chapter 4: Options

---

You can make many of the calculation settings for Jaguar jobs using the windows that are opened by clicking on the following buttons in the Jaguar panel:

<b>DFT</b>	Density functional theory calculations
<b>Local MP2</b>	Local Møller-Plesset second-order perturbation theory calculations
<b>GVB</b>	Generalized valence bond calculations
<b>Optimization</b>	Geometry optimization and forces
<b>Solvation</b>	Solvation energy calculations
<b>Properties</b>	Multipole moments and charge fitting and hyperpolarizability properties
<b>Frequencies</b>	Vibrational frequencies, IR intensities, and thermochemical properties
<b>Basis Set</b>	Basis set options
<b>Methods</b>	Initial guess, convergence, orbital localization, and various technical settings
<b>Surfaces</b>	Generate plotting data for visualization of surfaces in Maestro

In addition, you can perform  $pK_a$  calculations and J2 theory calculations using a built-in batch script.

Optimizations are described in [Chapter 5](#), and the other GUI options are described in [Chapter 3](#) and [Chapter 6](#).

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 3.9.3 on page 46](#).

### 4.1 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 8.5 on page 159](#). 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. You can also specify functionals to use for a non-self-consistent DFT evaluation of the energy of an HF or GVB wavefunction.

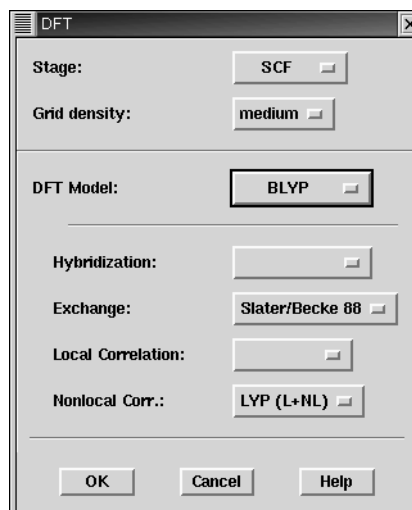


Figure 4.1. The DFT window.

### 4.1.1 Stage and Grid Density

By default, the Stage setting is SCF, meaning that Jaguar performs an SCF calculation of the Kohn-Sham orbitals and DFT energy.<sup>1</sup> If you set Stage to Post-SCF, Jaguar evaluates the DFT energy of the final wave function using the functionals you have specified.<sup>2</sup> You may also choose one set of functionals for the SCF stage and another set for a post-SCF DFT energy evaluation by making functional settings for each Stage choice in turn. If you do a post-SCF DFT energy evaluation on any wavefunction, you cannot perform a geometry optimization or calculate the solvation energy, polarizability, or any other non-default properties.

The Grid density menu determines the grid for DFT calculations. By default, DFT calculations use grids with a medium point density,<sup>3</sup> but finer density grids are also available.<sup>4</sup>

The other settings determine the functionals used, if any. Unless you select a functional or functionals, no DFT calculation is performed.

1. Keyword **dfname** in the **gen** section of the input file selects functionals for the SCF calculation.
2. Keyword **jdft** in the **gen** section of the input file selects post-SCF functionals.
3. Keywords **gdftmed** = -10, **gdftfine** = -11, and **gdftgrad** = -12 in **gen** section of input file.
4. Keywords **gdftmed**, **gdftfine**, and **gdftgrad** = -13 in **gen** section of input file.

## 4.1.2 DFT Model Options

The most commonly used functionals can be selected from the DFT Model option menu. Below this menu are four option menus that allow you to select local and nonlocal exchange functionals and local and nonlocal correlation functionals. When you make a selection from the DFT Model menu, the selections in these four menus change accordingly. By default, DFT is not used, so the default settings are none.

The DFT Model options include both pure DFT methods and hybrid methods, which include a Hartree-Fock exchange contribution as well as local and nonlocal functionals. Most of the hybrid methods employ either the parameters developed for Becke's three-parameter method [27, 28] (Becke 3) or the parameters developed for Becke's Half & Half method [26]. The option menu also contains some recently developed hybrid and non-hybrid functionals. The functionals available from the option menu are described below.

Functionals with local exchange only:

- HFS<sup>5</sup>: Slater local exchange functional [29]
- Xalpha<sup>6</sup>: X $\alpha$  local exchange functional [29]

Functionals with local exchange and local correlation:

- SVWN<sup>7</sup>: Slater local exchange functional [29], Vosko-Wilk-Nusair (VWN) local correlation functional [30]
- SVWN5<sup>8</sup>: Slater local exchange functional [29], Vosko-Wilk-Nusair 5 (VWN5) local correlation functional [30]

Functionals with local and nonlocal exchange and correlation:

- BLYP<sup>9</sup>: Exchange: Slater local functional [29], Becke 1988 nonlocal gradient correction [32]; correlation: Lee-Yang-Parr local and nonlocal functionals [33]
- BPW91<sup>10</sup>: Exchange: Slater local functional [29], Becke 1988 nonlocal gradient correction [32]; correlation: Perdew-Wang 1991 GGA-II local and nonlocal functionals [31]
- BP86<sup>11</sup>: Exchange: Slater local functional [29], Becke 1988 non-local gradient correction [32]; correlation: Perdew-Zunger 1981 local functional [34], Perdew 1986 gradient correction functional [35]

---

5. Keyword **dftname** = hfs in **gen** section of input file.

6. Keyword **dftname** = xalpha in **gen** section of input file.

7. Keyword **dftname** = svwn in **gen** section of input file.

8. Keyword **dftname** = svwn5 in **gen** section of input file.

9. Keyword **dftname** = blyp in **gen** section of input file.

10. Keyword **dftname** = bpw91 in **gen** section of input file.

11. Keyword **dftname** = bp86 in **gen** section of input file.

- BP86-VWN5<sup>12</sup>: Exchange: Slater local functional [29], Becke 1988 nonlocal gradient correction [32]; correlation: Vosko-Wilk-Nusair (VWN) local functional [30], Perdew 1986 gradient correction functional [35]
- PWP91<sup>13</sup>: Exchange: Slater local functional [29], Perdew-Wang 1991 gradient correction functional [31]; correlation: Perdew-Wang 1991 GGA-II local and nonlocal functionals [31]
- HCTH407<sup>14</sup>: Hamprecht-Cohen-Tozer-Handy functional including local and nonlocal exchange and correlation, reparametrized with a training set of 407 molecules by Boese and Handy [40]
- PBE<sup>15</sup>: Perdew-Burke-Ernzerhof local and nonlocal exchange and correlation functional [41]

## Hybrid functionals:

- B3LYP<sup>16</sup>: Exchange: exact HF, Slater local functional [29], Becke 1988 nonlocal gradient correction [32]; correlation: Vosko-Wilk-Nusair (VWN) local functional [30], Lee-Yang-Parr local and nonlocal functional [33]. See refs. 27 and 28.
- B3PW91<sup>17</sup>: Exchange: exact HF, Slater local functional [29], Becke 1988 nonlocal gradient correction [32]; correlation: Perdew-Wang 1991 local and GGA-II nonlocal functional [31]
- B3P86<sup>18</sup>: Exchange: exact HF, Slater local exchange functional [29], Becke 1988 nonlocal gradient correction [32]; correlation: Vosko-Wilk-Nusair (VWN) local functional [30], Perdew 1986 nonlocal gradient correction [35]
- BHandH<sup>19</sup>: 50% exact HF exchange, 50% Slater local exchange functional [29]
- BHandHLYP<sup>20</sup>: Exchange: 50% exact HF exchange, 50% Slater local exchange functional [29]; correlation: Lee-Yang-Parr local and nonlocal functionals [33]
- B97-1<sup>21</sup>: Reparametrization of Becke's 1997 hybrid functional [36] by Hamprecht, Cohen, Tozer, and Handy [39]

---

12. Keyword **dftname** = bp86-vwn5 in **gen** section of input file.

13. Keyword **dftname** = pwp91 in **gen** section of input file.

14. Keyword **dftname** = hcth407 in **gen** section of input file.

15. Keyword **dftname** = pbe in **gen** section of input file.

16. Keyword **dftname** = b3lyp in **gen** section of input file.

17. Keyword **dftname** = b3pw91 in **gen** section of input file.

18. Keyword **dftname** = b3p86 in **gen** section of input file.

19. Keyword **dftname** = bhandh in **gen** section of input file.

20. Keyword **dftname** = bhandhlyp in **gen** section of input file.

21. Keyword **dftname** = b97-1 in **gen** section of input file.

- B98<sup>22</sup>: Becke’s 1998 hybrid functional including the Laplacian of the density and kinetic energy density terms as well as gradient terms [37]
- SB98<sup>23</sup>: Schmider and Becke reparametrization of Becke’s 1998 functional [38]

The names of the functionals in this list are valid values of the keyword **dfname**, which you can use in the **gen** section of the input file instead of **idft** to specify the functional. The names are case-insensitive.

### 4.1.3 Custom Functionals

The four menus below the DFT Model option menu provide information on which functionals are currently selected and allow you to make functional choices not available from the DFT Model options described above.<sup>24</sup> If you make a choice from these menus that does not correspond to a DFT Model option, the DFT Model option changes to Custom.

The Hybridization menu choices are Half & Half and Becke 3 par. If you choose one of these, the Hartree-Fock treatment of the exchange and the contributions of the functional terms selected are weighted by coefficients from either Becke’s Half & Half method or his three-parameter method.

The Exchange menu choices include two local exchange functionals, Slater and  $X\alpha$  [29], and two combinations of local and nonlocal functionals, combining Slater local exchange with either the Becke 1988 [32] or the Perdew-Wang 1991 GGA-II [31] non-local exchange correction term. If you do not make a selection, Jaguar uses the exact Hartree-Fock exchange. However, if you are using a hybrid method, you should select an exchange functional from the menu.

The local correlation functional options are two functionals by Vosko, Wilk, and Nusair [30] (labeled VWN and VWN5), Perdew and Zunger’s 1981 functional [34] (labeled PZ81), and Perdew and Wang’s 1991 local correlation functional [31] (labeled PW91). Nonlocal correlation options are Perdew’s 1986 gradient correction functional [35] (labeled PW86), Perdew and Wang’s 1991 generalized gradient approximation correlation functional [31] (labeled GGA-II), and the Lee-Yang-Parr functional [33], which includes both local and non-local terms (labeled LYP (L+NL)).

As an example, to set up a calculation using Becke’s three-parameter method to weight the Slater/Becke 88 exchange functional, Perdew and Zunger’s 1981 local correlation functional, and Perdew’s 1986 non-local gradient correction functional, you could select Becke 3 par. from the Hybridization menu, Slater/Becke 88 from the Exchange menu, PZ81 from

---

22. Keyword **dfname** = b98 in **gen** section of input file.

23. Keyword **dfname** = sb98 in **gen** section of input file.

24. DFT keyword settings are extensive and complicated, so further options are not footnoted. See [Section 9.5.7 on page 173](#) for more information.

the Local Correlation menu, and Perdew 86 from the Non-local Corr. menu. As another example, you could select the Half & Half option from the DFT Model choices, then add functionals by selecting, for instance, Slater/Becke 88 to add Becke's non-local gradient correction to the exchange functional, and LYP (L+NL) for a correlation treatment.

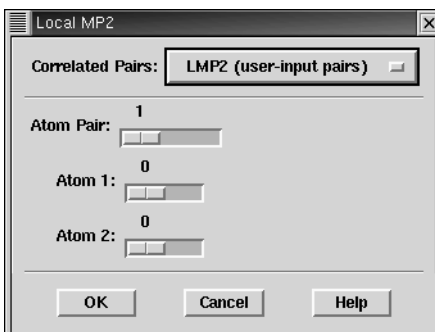
## 4.2 Local MP2 Settings

The Local MP2 button opens a window that allows you to set up a local Møller-Plesset second-order perturbation theory [42–45] calculation. The local MP2 (LMP2) method greatly reduces the basis set superposition errors that can arise from the canonical MP2 method [45]. 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 8.4 on page 156](#).

### 4.2.1 Summary of the LMP2 Method in Jaguar

For closed shell systems, you can perform LMP2 geometry optimizations, charge fitting, solvation calculations, and most other options available with HF wavefunctions. Local MP2 geometry optimizations employ analytic gradients. For calculations of LMP2 dipole moments, Jaguar computes a coupled perturbed Hartree-Fock (CPHF) term, 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 4.6.1 on page 60](#) and [Section 4.6.2 on page 62](#).

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 4.8 on page 70](#) and in several of the periodic tables of information by element beginning on [page 222](#). A warning is displayed if you choose a non-pseudospectral calculation.



*Figure 4.2. The Local MP2 window.*



The local MP2 reference wave function is produced through Pipek-Mezey localization [47] of the usual Hartree-Fock reference wave function, a procedure which involves a unitary transformation of the occupied canonical Hartree-Fock orbitals. This localization procedure, which is similar to the localization of bond pairs for the GVB representation of a molecule, *does not* change the reference energy.

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 wavefunctions 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 the Pipek-Mezey localized orbitals that are localized on the specified atom pairs. The atomic orbital coefficients for each Pipek-Mezey orbital are evaluated and 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 Pipek-Mezey orbital is considered to be localized on that single atom, and that Pipek-Mezey orbital will be included in any LMP2 calculation for which that atom is specified in any requested LMP2 atom pairs.

## 4.2.2 Setting up an LMP2 Calculation

Jaguar will not perform an LMP2 calculation unless you make a selection under Correlated Pairs to indicate which atoms should be treated at the LMP2 level. To perform an LMP2 calculation that includes all atoms, select LMP2 (all pairs).<sup>25</sup>

You can perform a “local local” MP2 calculation by selecting a subset of atoms to be treated with LMP2, while the remaining atoms are treated at the HF level. Jaguar includes a setting to treat 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 LMP2 (hetero pairs).<sup>26</sup>

To specify atom pairs yourself, select LMP2 (user-input pairs) and choose the LMP2 pairs using the three sliders in the LMP2 window.<sup>27</sup> To specify the first LMP2 pair, leave the

---

25. Keyword **mp2** = 3 in **gen** section of input file.

26. Keywords **iheter** = 1 and **mp2** = 3 in **gen** section of input file.

27. If you were editing an input file directly instead of using the GUI, you would need to set LMP2 pairs in the **lmp2** section of the input file.

slider marked Pair set at 1. Next, specify the atom numbers for the atoms in that pair using the Atom 1 and Atom 2 sliders or by clicking in the slider box. The appropriate atom labels (for example, “H2”) are displayed to the right of the Atom 1 and Atom 2 boxes and reflect the atoms selected in those boxes. Additional pairs can be entered in the same manner after specifying a new pair number with the Pair slide bar. You can also combine the user-input pairs and heteroatom pairs options, setting your own LMP2 pairs in addition to all heteroatom pairs.

### 4.3 Generalized Valence Bond (GVB) Settings

The window opened when you click GVB allows you to request a generalized valence bond (GVB) calculation and to set the GVB pairs for that calculation. You can also choose to do a restricted configuration interaction (RCI) calculation [12, 48, 49] for some or all of the pairs. The theory behind GVB and GVB-RCI calculations is explained in Chapter 8.

The default Jaguar calculation is closed shell or open shell Hartree-Fock, depending upon the number of electrons in the system. To include electron correlation with the Generalized Valence Bond Perfect-Pairing (GVB-PP) method [20], you can provide a list of GVB pairs to be used in the calculation. Specifying this list automatically enables GVB.

The GVB and GVB-RCI methods in Jaguar do not include the concept of resonance. Consequently, the GVB or GVB-RCI pair input for a molecule such as benzene, for example, should include alternate single and double bonds for its carbon ring. If you perform a GVB or GVB-RCI geometry optimization on a molecule with equal, resonating bonds (like the carbon bonds in benzene), you should force the optimizer to keep their bond distances the same, even if the input lists different bond orders for the bonds. To impose this restriction, use Z-matrix form for your geometry input and set all relevant bonds equal to the same variable. See Section 3.2.5 on page 29 and Section 3.2.6 on page 31 for more information.

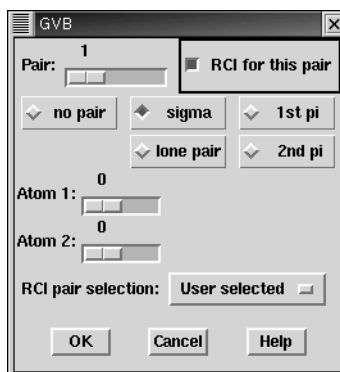


Figure 4.3. The GVB window.

### 4.3.1 GVB or GVB-RCI Pair Input

In order to describe the placement of GVB pairs for a GVB calculation,<sup>28</sup> you need to know the atom numbers for the relevant atoms. To display the atom numbers in the Maestro Workspace, choose Atom Labels from the Display menu, select Atom Number in the Composition tab, then click the All button in the Label Atoms group box. You can also use the Edit Geometry window, and identify the atom number by the order the atoms are listed in the file.

You can select GVB pairs in any order. To specify the first GVB pair, leave the Pair slider set at 1. Next, specify the desired pair type by clicking in the appropriate box. You can choose a sigma bond, pi bond, second pi bond (in a triple bond), or lone pair. If you select a lone pair, it is assigned to a single atom, so when you specify the number of the atom for the lone pair using the Atom 1 slider or by clicking in the slider box, the same number is displayed for Atom 2. If you select any other kind of pair, you must set the atom number for Atom 2 separately. The appropriate atom labels (for example, "H2") appear to the right of the Atom 1 and Atom 2 slider boxes.

Additional GVB pairs can be specified in the same way as the first pair, after changing the number shown by the Pair slider. *A particular atom should have either all or none of its lone pairs specified as GVB lone pairs.* Also, you cannot set GVB lone pairs when you are using a minimal basis set (e.g., STO-3G).

If you compute solvation energies using GVB or LMP2, as described in [Section 4.5 on page 58](#), we recommend using heteroatom pairs for the GVB calculation for the most efficient results, since solvation energy calculations often use radii optimized for calculations with heteroatom pairs set. (See [Section 10.6 on page 253](#) for more details.) Heteroatom pairs are all pairs whose atoms are of different elements, except for C–H pairs.

You can select RCI pairs in two ways. By default, RCI is off for all pairs, and the RCI pair selection option menu is set to user selected. In this case, a pair is included in an RCI calculation only if you select RCI for this pair beside the Pair slider. To set RCI on for all pairs, choose RCI on for all pairs from the RCI pair selection option menu. The RCI for this pair button is dimmed, but is automatically on for all pairs.

## 4.4 GVB-LMP2 Calculations

Jaguar's pseudospectral GVB-LMP2 module allows this multireference perturbation method to be applied to medium and large molecules with reasonable CPU, memory, and disk use. The method has been shown to give highly accurate conformational energies [18].

---

28. If you were editing an input file directly instead of using the GUI, you would need to set GVB pairs in the **gvb** section of the input file.

For GVB-LMP2 calculations, Jaguar first performs an SCF calculation of the reference GVB wavefunction using the GVB pairs specified in the input. Next, the program applies an LMP2 perturbative correction to the energy. The LMP2 calculation is performed on the entire system, even if only part of the system was treated at the GVB level.

To set up a GVB-LMP2 calculation, first specify the GVB pairs to be used in the GVB reference wavefunction, following the procedure described in [Section 4.3 on page 56](#). Next, request an LMP2 treatment on all pairs in the system by choosing LMP2 (all pairs) from the Pairs option menu in the Local MP2 window. You can perform the LMP2 calculation at either the valence only or the all electrons level.

We advise using GVB-LMP2 primarily for single-point energy calculations since Jaguar cannot compute GVB-LMP2 atomic charges or analytic gradients. For best results with GVB-LMP2, first run your calculations with the 6-31G\*\* basis set, then change the basis set in the restart file to cc-pVTZ(-f) and restart the job. (See [Section 7.2 on page 142](#) for a description of how to restart jobs.) This procedure will generally be significantly faster than running a GVB-LMP2/cc-pVTZ(-f) job from scratch.

The most effective choice of GVB reference wavefunction depends on the type of calculation being performed. For conformational energy calculations, we recommend setting all possible GVB lone pairs, all possible GVB pairs that describe multiple bonds between two carbons, and all GVB pairs for bonds between two different non-hydrogen atoms. (For information on how to make these GVB pair settings automatically, see [Section 9.5.5 on page 170](#).) For studies of bond dissociation, all bonds from the atoms involved in the dissociating bond (or bonds) should be treated at the GVB-LMP2 level. Note also that for dissociation of multiple bonds, GVB-LMP2's accuracy is limited by its inadequate treatment of spin coupling between high-spin fragments; we recommend using the GVB-RCI method instead for such cases.

## 4.5 Solvation

Jaguar can treat solvated molecular systems with a self-consistent reaction field method, using its own Poisson-Boltzmann solver [15, 135].<sup>29</sup> You can compute solvation energies and minimum-energy solvated structures or solvated transition states. To perform a geometry optimization in solution, you must make appropriate settings in the Optimization window as well. The solvation energy from a geometry optimization 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.

In the SCRf method that Jaguar uses, Jaguar first calculates the usual gas phase wavefunction and from that the electrostatic potential, and fits that potential to a set of atomic charges, as described in [Section 4.6.1 on page 60](#). These charges are passed to the Jaguar

---

29. Keyword **isolv** = 2 in **gen** section of input file.

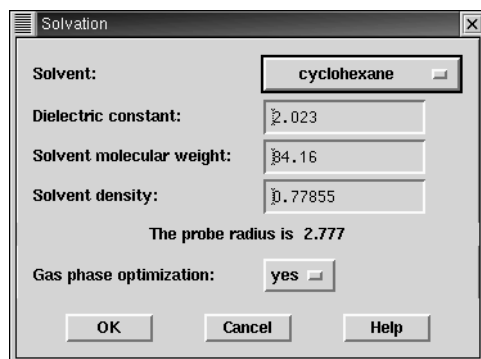


Figure 4.4. The Solvation window.

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 wavefunction 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.

Solvation energies can be computed for cases using HF, DFT, GVB, or LMP2 wavefunctions. For GVB or local LMP2 solvation energy calculations, we recommend using heteroatom pairs for the most efficient results, particularly since solvation energy calculations often use radii optimized for calculations with heteroatom pairs set. (See [Section 10.6 on page 253](#) for more details.) See [Section 4.2 on page 54](#) and [Section 4.3 on page 56](#) for information on setting LMP2 or GVB pairs.

### 4.5.1 Solvent Parameters

Solvent parameters are set in the Solvation window, which is opened by clicking the Solvation button in the Jaguar panel. If the solvent you want to use for your solvation energy calculation is listed in the set of choices available under the Solvent option menu, you can make the appropriate choice, and Jaguar performs a solvation calculation, setting the appropriate dielectric constant<sup>30</sup> and probe radius.<sup>31</sup> The dielectric constant [52] and probe radius [53] values set by Jaguar for various solvents are shown in [Table 4.1](#).

To use a solvent that is not on the list, define it by choosing other from the Solvent option menu and changing the entries for Dielectric constant, Solvent molecular weight, and Solvent density. The latter two values are used to calculate the probe radius (in angstroms), whose value is shown in the same window (see reference 53).

30. Keyword **epsout** in **gen** section of input file.

31. Keyword **radprb** in **gen** section of input file.

Table 4.1. Parameters for Various Solvents

Solvent	Dielectric Constant	Probe Radius
cyclohexane	2.023	2.78
carbon tetrachloride	2.238	2.67
benzene	2.284	2.60
chlorobenzene	5.708	2.72
1,2-dichloroethane	10.65	2.51
methanol	33.62	2.00
nitrobenzene	35.74	2.73
water	80.37	1.40

## 4.5.2 Performing or Skipping a Gas Phase Optimization

If you are computing the solvation 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.<sup>32</sup> However, if you want only an optimized *structure* in solution and do not care about the accuracy of its computed solvation energy, you can skip the gas phase geometry optimization by setting the Gas phase optimization option to no.<sup>33</sup>

## 4.6 Properties

Various molecular properties can be calculated for a particular wavefunction. These calculations are normally performed using the converged SCF wavefunction. By default, none of the properties listed below are computed, but you can compute them by changing the settings in the Properties window, which you open by clicking Properties.

### 4.6.1 Electrostatic Potential Fitting

Jaguar can fit the molecular electrostatic potential (ESP) to a set of point charges [55, 56]. These monopoles can be located either at the atomic centers<sup>34</sup> or at the atomic centers and the bond midpoints,<sup>35</sup> depending on the Fit ESP to selection. The atomic charges are

32. Keyword **nogas** = 0 in **gen** section of input file.

33. Keyword **nogas** = 2 in **gen** section of input file.

34. Keyword **icfit** = 1 in **gen** section of input file.

35. Keyword **icfit** = 2 in **gen** section of input file.

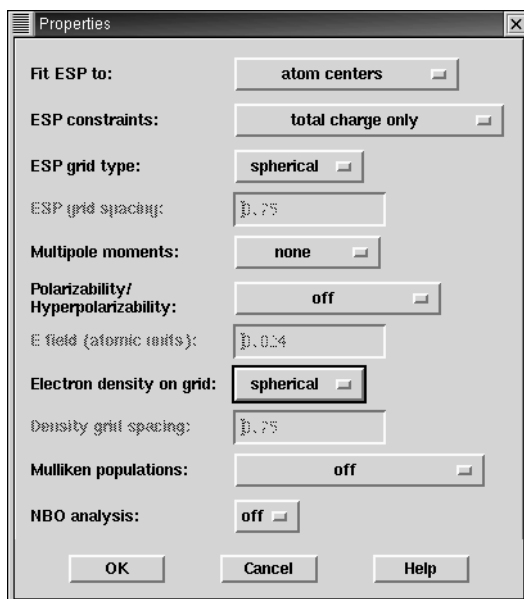


Figure 4.5. The Properties window.

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.

For electrostatic potential fitting of an LMP2 wavefunction, 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 may 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 exactly the dipole moment (and other higher moments, if specified), by choosing charge + dipole moment<sup>36</sup> (or the appropriate higher-moment choice<sup>37</sup>) from the ESP constraints option menu. (For LMP2 wavefunctions, 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 describe the Coulomb field around the molecule. The dipole moment is generally very close to the quantum mechanical dipole moment as calculated from the wave function, and constraining the charge fitting to reproduce it 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.

36. Keyword **incdip** = 1 in **gen** section of input file.

37. Keyword **incdip** = 2 or 3 in **gen** section of input file.

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.<sup>38</sup> An alternative is to use a regular lattice of grid points [56], by choosing rectangular from the ESP grid type option menu.<sup>39</sup> You can then set the spacing in bohr between points in this lattice in the Rect. grid spacing box.<sup>40</sup> 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 [57] van der Waals radii for hydrogen and for carbon through argon, and universal force field [54] van der Waals radii for all other elements. These radii are listed in Table 9.42. 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 218.

You can also print out the values of the electrostatic potential at grid points whose locations you specify. See Section 9.5.12 on page 188.

## 4.6.2 Multipole Moments

Jaguar can compute multipole moments<sup>41</sup> through hexadecapole for HF, GVB, or DFT wavefunctions, and can compute dipole moments for LMP2 wavefunctions. 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.

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 [55, 56], as described above, or by Mulliken population analysis [58], as described below, the multipole moments are also calculated from these point charges for comparison.

## 4.6.3 Polarizability and Hyperpolarizability

You can calculate polarizabilities and first hyperpolarizabilities by making the appropriate choice from the Polarizability/Hyperpolarizability options menu. To calculate second hyperpolarizabilities, you must set **ipolar** = -2 in the **gen** section of the input file.

---

38. Keyword **gcharge** = -1 in **gen** section of input file.

39. Keyword **gcharge** = -2 in **gen** section of input file.

40. Keyword **wispc** in **gen** section of input file.

41. Keyword **ldips** = 2, 3, 4, or 5 in **gen** section of input file.



The coupled-perturbed HF option<sup>42</sup> calculates both polarizability and hyperpolarizability using coupled perturbed Hartree-Fock (CPHF) techniques. In general, this option is superior to the finite field option,<sup>43</sup> but the CPHF option can be used only with closed-shell and unrestricted open-shell wavefunctions, and with basis sets that do not use effective core potentials. (In Jaguar, the basis sets with effective core potentials are CSDZ and those with names beginning with “LA.” See [Section 4.8 on page 70](#) for information on basis sets.)

The finite field option [17] uses a 5-point finite difference method, which employs the results from 13 SCF calculations: one with no field; one with an electric field of  $E$  (where  $E$  is 0.024 au by default) in the  $x$  direction; one with a field of  $-E$  in the  $x$  direction; four others with fields of  $+E$  and  $-E$  in the  $y$  and  $z$  directions; and six others using fields of  $+aE$  and  $-aE$  in the  $x$ ,  $y$ , and  $z$  directions, where  $a$  is a constant determined automatically.

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

If you want to change the electric field used for the finite field calculation or to use other finite field methods to calculate the polarizability and hyperpolarizability, see [Section 9.5.12 on page 188](#) for information on editing the input file appropriately.

#### 4.6.4 Electron Density

The electron density for the final wavefunction can be evaluated on a set of grid points. The Cartesian coordinates of these grid points and the electron density in au, respectively, for each grid point are written to the file `jobname.chdens`, where `jobname.in` is the input file for the Jaguar job.

If you select rectangular<sup>44</sup> for the electron density calculation, the grid used is rectangular with spacing in Angstroms determined by the density grid spacing<sup>45</sup> set immediately below. For spherical,<sup>46</sup> the default choice, the electron density is evaluated on the ultrafine grid used by the pseudospectral method. We recommend using the spherical grid for quantitative results, although the rectangular grid is sometimes useful for display purposes. (To use a different grid for electron density calculations, see [Section 9.5.23 on page 210](#) for information about the grid keyword **geldens** in the **gen** section of the input file.)

---

42. Keyword **ipolar** = -1 in **gen** section of input file.

43. Keyword **ipolar** = 5 in **gen** section of input file.

44. Keywords **ldens** = 1 and **geldens** = -3 in **gen** section of input file.

45. Keyword **denspc** in **gen** section of input file.

46. Keywords **ldens** = 1 and **geldens** = 4 in **gen** section of input file.

### 4.6.5 Mulliken Population Analysis

Mulliken populations [58] can be computed for each atom, giving a representation of the molecule as a set of nuclear-centered point charges.<sup>47</sup> 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.

Mulliken populations can be computed for each basis function as well as for each atom<sup>48</sup>, or for each bond between neighboring atoms, as well as by atom and basis function.<sup>49</sup>

### 4.6.6 Natural Bond Orbital (NBO) Analysis

To request a default Natural Bond Orbital (NBO) analysis [59] at the end of the Jaguar job, turn on NBO analysis<sup>50</sup> at the bottom of the Properties window. The output from the NBO analysis is included in the Jaguar output file.

Other options for NBO calculations can also be specified in the **nbo** section or in the **core**, **choose**, and **nrtstr** sections of the Jaguar input file. It is not possible to run NEDA (Natural Energy Decomposition Analysis) calculations from Jaguar, however.

See [Section 9.18 on page 236](#) and the *NBO Program Manual* for more details on NBO input and output.

## 4.7 Frequencies and Related Properties

Using the Frequencies window, you can request calculations of frequencies, infrared (IR) intensities, and thermochemical properties (heat capacity, entropy, enthalpy, and Gibbs free energy). Vibrational frequencies and thermochemical properties can be computed for HF, DFT, LMP2, or GVB wavefunctions (except that numerical frequencies cannot be computed for unrestricted HF or DFT wavefunctions). IR intensities are computed by default for frequency jobs for which either: (a) analytic HF frequencies are computed and the basis set does not have any effective core potentials (see [Section 4.8 on page 70](#) for details), or (b) HF, GVB, or DFT frequencies are computed numerically. The results of frequency calculations can be animated in Maestro.

---

47. Keyword **mulken** = 1 in **gen** section of input file.

48. Keyword **mulken** = 2 in **gen** section of input file.

49. Keyword **mulken** = 3 in **gen** section of input file.

50. Empty **nbo** section in input file (“&nbo &”).

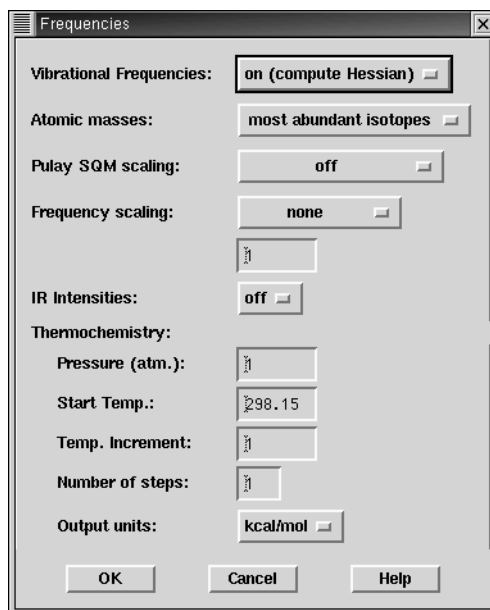


Figure 4.6. The Frequencies window.

### 4.7.1 Frequencies

To calculate vibrational frequencies<sup>51</sup>, select on (compute Hessian) in the Frequencies window. Vibrational frequency calculations are available for HF, GVB, LMP2, and DFT wavefunctions in gas phase or in solution, but are not available for GVB-RCI or GVB-LMP2 calculations.

For gas phase HF and DFT jobs with basis sets that allow pseudospectral calculations and do not include f functions, Jaguar computes analytic frequencies. (See [Section 4.8 on page 70](#) for more information on basis sets.) Otherwise, Jaguar uses energies obtained at perturbed geometries to calculate the numerical derivatives of the analytically computed forces.

Generally, analytic frequency calculations are much faster than numerical frequency calculations. However, when frequencies are calculated analytically, molecular symmetry is turned off for the job. Therefore, if you want to compute analytic frequencies for large, highly symmetric molecules, you should first run any other computationally intensive portions of the job (geometry optimization, for instance), then use the new input (restart) file generated during the job as input for an analytic frequency job. (See [Section 7.2 on page 142](#) for information on generating restart files and restarting jobs.) If you want to

51. Keyword **ifreq** = 1 in **gen** section of input file.

calculate frequencies numerically instead, make the keyword setting **nmdr**=2 in the **gen** section of the input file, as described in [Section 9.5 on page 168](#).

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), choose use available Hessian<sup>52</sup> from the Vibrational Frequencies option menu.

If you choose to generate a Molden input file after a frequency calculation, the normal modes are written to the Molden file, enabling you to visualize the frequencies with this program. See [Section 6.5 on page 129](#) or [Section 9.5.20 on page 206](#) for more information on writing a Molden input file.

## 4.7.2 Atomic Masses

For frequency calculations, by default, the atomic mass used for each element is that of its most abundant isotope.<sup>53</sup> However, you can choose to use an average of the isotopic masses, weighted by the abundance of the isotopes, by selecting average isotopic masses<sup>54</sup> from the Atomic masses option menu.

## 4.7.3 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 options are available for frequency scaling: the Pulay et al. Modified Scaled Quantum Mechanical Force Fields (SQM) method [60] 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 containing C, H, N, O, and Cl, using the 6-31G\* basis set. Jaguar permits only the SQM scaling method to be used for B3LYP/6-31G\* frequency jobs. You can turn on SQM scaling for these jobs from the Frequencies window by setting the option menu labeled Pulay SQM scaling to B3LYP 6-31G\* factors.<sup>55</sup> The method is off<sup>56</sup> by default.

---

52. Keyword **ifreq** = -1 in **gen** section of input file.

53. Keyword **massav** = 0 in **gen** section of input file.

54. Keyword **massav** = 1 in **gen** section of input file.

55. Keyword **isqm** = 1 in **gen** section of input file.

56. Keyword **isqm** = 0 in **gen** section of input file.

Alternatively, for any type of frequency job, you can multiply all frequencies by the same scale factor by changing the value in the text box marked Frequency scaling.<sup>57</sup> Table 4.2 lists recommended scale factors for various methods and basis sets. The factors in the table are from ref. 61 and are optimized for the best agreement with experiment for the frequencies themselves. Ref. 61 also includes scale factors suitable for use when low-frequency vibrations are of particular interest, for zero-point vibrational energies, and for prediction of enthalpy and entropy. Other scale factors may be available in the literature.

Table 4.2. Recommended Frequency Scale Factors for Various Combinations of SCF Method and Basis Set (taken from Ref. [61])

SCF Method	Basis Set	Scale Factor
HF	3-21G	0.9085
HF	6-31G*	0.8953
HF	6-31+G*	0.8970
HF	6-31G**	0.8992
HF	6-311G**	0.9051
MP2	6-31G*	0.9434
MP2	6-31G**	0.9370
MP2	6-311G**	0.9496
BLYP	6-31G*	0.9945
BP86	6-31G*	0.9914
B3LYP	6-31G*	0.9614
B3P86	6-31G*	0.9558
B3PW91	6-31G*	0.9573

#### 4.7.4 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. For calculations that use a project table entry as the source of input, the vibrational data is incorporated when the job finishes, and a Vib column is added to the Project Table. The Vib column has a button labeled V for each entry that has vibrational data—much like the Surf column has 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.

<sup>57</sup>. Keyword `scalfr` in `gen` section of input file.

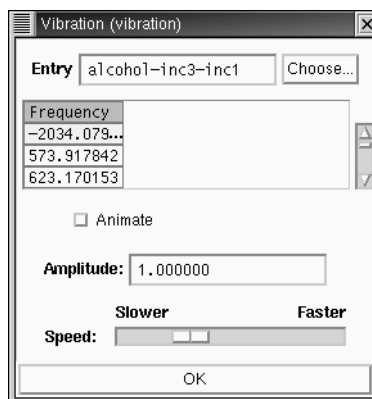


Figure 4.7. The Vibration panel.

If you did not run the frequency job from the Jaguar panel, the vibrational data is not incorporated because the molecule is not a project table entry. To view the animation, 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.

To view vibrational animations from calculations run with previous versions of Jaguar, 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. Click Edit Input to edit the restart file.
3. 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.”

4. Run the job.

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 Vibration panel.

### 4.7.5 Infrared Intensities

To calculate infrared intensities for each frequency in km/mol, select IR Intensities<sup>58</sup>. For HF jobs where frequencies are calculated analytically, the IR intensities are obtained from coupled perturbed Hartree-Fock (CPHF) calculations of the derivative of the dipole moment with respect to changes in the nuclear coordinates, and molecular symmetry is not used for the job. For calculations for which frequencies are computed numerically, the numerical derivative of the dipole can be obtained for IR intensity calculations. Analytic IR intensities are not available for open-shell molecules.

### 4.7.6 Thermochemical Properties

Thermochemistry calculations of a system's 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.

By using the thermochemistry settings, you can control the temperatures and pressure used for calculations of these quantities. The pressure<sup>59</sup> (in atm) used for thermochemical calculations is 1.0 by default, and the initial temperature<sup>60</sup> (in K) is 298.15 by default. Either of these settings can be changed. To compute thermochemical properties at more than one temperature, specify the differences between temperatures using the Temp. Increment<sup>61</sup> setting and the number of temperatures at which thermochemical properties should be computed with the Number of steps<sup>62</sup> setting, which is 1 by default.

By default, thermochemical output is in units of kcal/mol (for H and G) and cal/mol K (for  $C_v$  and S). To report the output in units of kJ/mol and J/mol K instead, select J/mol from the Output units menu<sup>63</sup>.

---

58. Keyword **irder** = 1 in **gen** section of input file.

59. Keyword **press** in **gen** section of input file.

60. Keyword **tmpini** in **gen** section of input file.

61. Keyword **tmpstp** in **gen** section of input file.

62. Keyword **ntemp** in **gen** section of input file.

63. Keyword **ip28=2** in **gen** section of input file.

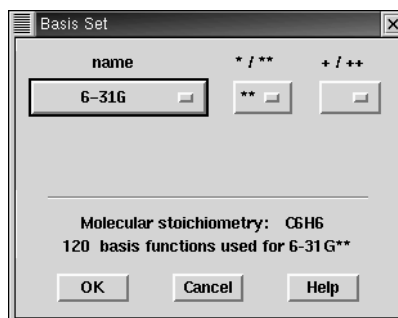


Figure 4.8. The Basis Set window.

## 4.8 Basis Set

From the Basis Set window, you can choose a basis set<sup>64</sup> from the option menu shown under the label name, and select polarization and diffuse functions from the option menus marked \*/\*\* and +/++, respectively. If an option 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). If you select an italicized basis set name, the calculation will run all-analytically, without using the pseudospectral method. All of these settings are explained further below.

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.

You can perform counterpoise calculations with Jaguar, adding counterpoise atoms that have the usual basis functions for their elements, but include no nuclei or electrons. However, counterpoise atoms should be entered through the Edit window rather than the Basis Set window. See [Section 3.2 on page 26](#) if you want to use counterpoise atoms.

For any basis set with the \*\* option, choosing \*\* places polarization functions on all atoms, unless the basis set uses effective core potentials, in which case polarization functions are placed only on atoms not described with effective core potentials. For STO-3G and 3-21G basis sets, choosing \* places polarization functions on all atoms found in the third row (Na-Ar) or higher rows (for STO-3G) of the periodic table. For effective core potential basis sets (CSDZ and those whose names begin with “LA”), choosing \* places polarization functions on all atoms not described by effective core potentials except H and He. For all other basis sets, \* places polarization functions on all atoms except H and He.

64. Keyword **basis** in **gen** section of input file.



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.

Table 4.3 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 cc-pVDZ and cc-pVTZ basis sets include polarization functions by definition. Note that in versions 2.3 and earlier of Jaguar, the cc-pVTZ basis set did not include f functions, and therefore corresponds to the current cc-pVTZ(-f).

Table 4.3 also indicates the method used for the calculation: the fast pseudospectral method or the slower analytic method, in which four-center, two-electron integrals are computed explicitly, as in conventional ab initio programs. The analytic method is used only when optimized pseudospectral grids and dealiasing function sets for one or more atoms in the molecule are not available. For molecules whose atoms are all in the range H-Ar in the periodic table, we recommend using the 6-31G\*\* basis set (the default choice), which is one basis set that permits pseudospectral calculations.

The column headed “# of d fns.” indicates whether d shells include the five 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 second-order 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 9.5.14 on page 193. The orbital coefficients are always printed out in terms of the six Cartesian functions. The full references describing the basis sets are in the References list at the back of this manual.

Table 4.3. 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	62-66
3-21G	H-Xe	* (Na-Ar), + (Li-Ar), ++ (H-Ar)	H-Ar pseudospectral (analytic with + or ++), K-Xe analytic	6	67-69
4-21G	H-Ne	*, **	analytic	6	70
6-21G	H-Ar	*, **	analytic	6	67-69
4-31G	H-Ne	*, **	analytic	6	71-76
6-31G	H-Zn	*, **, +, ++ for H-Ar	H-Ar pseudospectral, K-Zn analytic	6	72-78

Table 4.3. Available Basis Sets That Do Not Include Effective Core Potentials (Continued)

Basis Set	Atoms Included	Options	Method	# of d fns.	Refs.
6-311G	H-Ar	*, **, +, ++	H, Li, C-F, Na, Si-Cl pseudospectral, others analytic	5	79-82
6-311G(3df-3pd)	H-Ar	+, ++	analytic	5	79-82
D95V	H, Li-Ne	*, **	analytic	6	83
D95	H, Li-Ne, Al-Cl	*, **	H, Li, C-F, Si-Cl pseudospectral, others analytic	6	83
MSV	H-Ru, Pd-Xe		analytic	5	84
cc-pVDZ	H-He, B-Ne, Al-Ar	+, ++	H, C-F, Si-Cl pseudospectral, others analytic; +, ++ analytic	5	85-88
cc-pVDZ(-d) (without d functions)	H-He, B-Ne, Al-Ar	+, ++	H, C-F, Si-Cl pseudospectral, others analytic; +, ++ analytic	5	85-88
cc-pVTZ	H-Ar, Ca, Ga-Kr	+, ++	H, C-F, Si-Cl pseudospectral, others analytic	5	85-88
cc-pVTZ(-f) (without f functions)	H-Ar, Ca, Ga-Kr	+, ++	H, C-Ne, Si-Ar pseudospectral, others analytic	5	85-88
cc-pVQZ(-g) (without g functions)	H-F, Na-Ar, Ca, Ga-Kr	+, ++	H, C-O, pseudospectral, others analytic; +, ++ analytic	5	85
MIDI!	H, C-F, P-Cl		pseudospectral	5	89
TZV	H-Kr	*, **	analytic	5	90
TZV(f)	Sc-Zn		analytic	5	90

The other available basis sets, which are listed in Table 4.4, 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 4.4. 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)	91-92
LAV2D	Na-La, Hf-Bi	H, Li-Ne (D95V)	*, ** (H, Li-Ne)	91-92
LAV2P	Na-La, Hf-Bi	H-Ne (6-31G)	*, ** (H-Ne); +, ++ (H-Ne)	91-92
LAV3D	Na-La, Hf-Bi	H, Li-Ne (D95V)	*, ** (H, Li-Ne)	91-92
LAV3P	Na-La, Hf-Bi	H-Ne (6-31G)	*, ** (H-Ne); +, ++ (H-Ne)	91-92
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)	93
LACVP	K-Cu, Rb-Ag, Cs-La, Hf-Au	H-Ar (6-31G); Zn-Kr, Cd-Xe, Hg-Bi (LAV3P)	*, ** (H-Ar); +, ++ (H-Ar)	93
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)	94
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)	95
ERMLER2	K-Lr	H-Ar (6-31G)	*, **, +, ++ (H-Ar, Ga-Kr, In-Xe, Tl-Rn)	96-103

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 Cundari-Stevens ECP basis set [95], 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 [96-101], 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. Polarization and diffuse functions for the 4p, 5p, and 6p elements have been added from the relativistic all-electron double-zeta set of Dyll [103]. This addition represents a change from the previous definition of the basis set, in which the addition of a \* or + to the basis set name had no effect for the p-block elements defined by ECPs. Consequently, results obtained using the current release with polarization or diffuse functions on the ERMLER2 basis sets may differ from results obtained with previous releases.

Table 4.4 describes the ECP basis sets. 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. All ECP basis sets use five d functions, as described earlier in this section.

Currently, the LACVP, LAV3P, and LACV3P basis sets use the pseudospectral method, while all other ECP basis sets use the analytic method, computing the four-center, two-electron integrals explicitly. 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 wavefunctions. 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.

## 4.9 Methods

The Methods window includes various settings that control the type of calculation and how the calculation is performed, including the wave function type, the electronic states, the source of the initial wave function, the convergence method, the maximum number of SCF iterations, and the job's accuracy level. You should not need to change the convergence options unless you are having convergence problems. You might want to select the wave function type or the electronic state.

### 4.9.1 Wavefunction Type (Restricted or Unrestricted)

To perform an unrestricted HF or DFT calculation, you can select Unrestricted (UHF/UDFT)<sup>65</sup> from the Wavefunction type option menu. The default method for open-shell systems is restricted open-shell HF or DFT.<sup>66</sup>

---

65. Keyword **uhf** = 1 in **gen** section of input file.

66. Keyword **uhf** = 0 in **gen** section of input file.

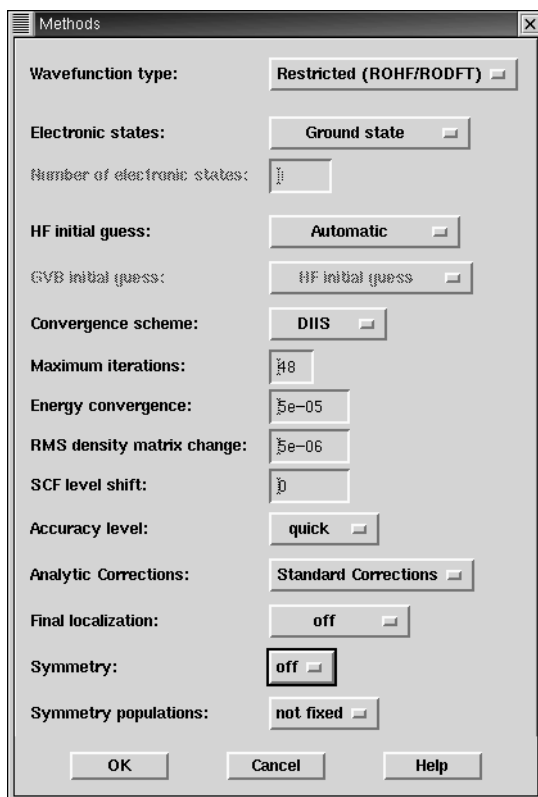


Figure 4.9. The Methods window.

## 4.9.2 Selecting Electronic States

To perform a configuration interaction singles (CIS) calculation to obtain information on excited electronic states, choose Excited States from the Electronic States list,<sup>67</sup> and enter the number of excited states for which you want information in the Number of Electronic States text box.<sup>68</sup> When you select a CIS calculation, Jaguar first performs a closed-shell Hartree-Fock calculation. The Hartree-Fock orbitals are used as input to the CIS calculation. You cannot do a CIS calculation based on an open-shell SCF wavefunction. The output includes excitation energies, transition dipole moments, and oscillator strengths from the ground state to the excited states.

67. Keyword **icis** = 1 in **gen** section of input file.

68. Keyword **nroot** in **gen** section of input file.

### 4.9.3 Choosing an Initial Guess Type

The default HF initial guess selection is Automatic, meaning that Jaguar selects the initial guess method most likely to lead to convergence. However, you can explicitly select from among several possible initial guess algorithms.

By default, for non-GVB calculations on simple closed-shell systems with no transition metals, Jaguar constructs the initial wavefunction from orbitals that give the best overlap with previously calculated orbitals from atomic calculations.<sup>69</sup> This initial guess method, which can be selected for any calculation, is identified as AO Overlap. The algorithm used is described in reference 14. This method compares well with the semi-empirical schemes that other ab initio programs use to obtain initial guesses.

Jaguar also provides a unique initial guess feature to improve SCF convergence (both HF and DFT), particularly for transition-metal-containing systems. As described in reference 19, research at Schrödinger has established that poor convergence of these systems is very often due to problems with the trial wavefunction's orbital shapes and occupations. We have therefore developed an algorithm based upon ligand field theory that creates a high-quality initial guess specifically designed for transition-metal-containing systems [19]. The HF initial guess options, labeled Ligand Field Theory<sup>70</sup> and LFT + dd repulsion,<sup>71</sup> both use this algorithm. See reference 19 for further details. Both initial guess methods can take advantage of user-provided information on charges and spins of “fragments” within the system, as described in Section 7.1 on page 139, although such information is not required.

If you restart a calculation with an input file generated during a previous run, as described in Section 7.2 on page 142, the wave function from the earlier run is read from the **guess** section and used as an initial guess, unless you change the default choice of **Read** from input.<sup>72</sup> The **guess** section is described in Section 9.10 on page 227. 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).

For GVB calculations, the GVB initial guess options menu lets you choose the method used to generate this guess. By default, the GVB initial guess is automatically constructed from the Hartree-Fock initial guess by piece-wise localization.<sup>73</sup>

Another option is to use a converged HF wavefunction as the basis for the GVB initial guess. For this option, select compute from HF converged wavefunction<sup>74</sup> for the GVB initial guess setting, and make the appropriate setting under HF initial guess for the

---

69. Keyword **iguess** = 10 in **gen** section of input file.

70. Keyword **iguess** = 25 in **gen** section of input file.

71. Keyword **iguess** = 30 in **gen** section of input file.

72. Keyword **iguess** = 1 in **gen** section of input file.

73. Keyword **ihfgvb** = 2 in **gen** section of input file, or keyword **ihfgvb** = 0 if **iguess** is not 1.

74. Keyword **ihfgvb** = 1 in **gen** section of input file.

Hartree-Fock calculation. Because this selection requires two SCF calculations, one for HF and one for GVB, it is considerably more expensive than using the GVB initial guess. You might want to try this option if you encounter convergence difficulties.

The third possibility for the GVB initial guess is to read in a GVB wavefunction from the input file's **guess** section and to use that as the initial guess for the calculation.<sup>75</sup> For instance, if you were restarting a job, as described in [Section 7.2 on page 142](#), and wanted to use the result from the previous run as an initial guess for the new run, the **Read from input** option for the GVB initial guess would allow you to do so.

#### 4.9.4 Convergence Issues

We recommend using the default Direct Inversion in the Iterative Subspace (DIIS)<sup>76</sup> or GVB-DIIS<sup>77</sup> SCF convergence schemes [11, 104] 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 and multiple GVB pairs. You can also use the OCBSE convergence scheme<sup>78</sup> [20], although it is generally much slower than DIIS.

You can change the maximum number of SCF iterations allowed.<sup>79</sup> 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 default energy convergence criterion for Jaguar,<sup>80</sup> which can also be changed, is set to  $5.0 \times 10^{-5}$  Hartrees for the total energy on consecutive iterations, except for polarizability calculations, for which it 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.

When the RMS change in density matrix elements for a polarizability, hyperpolarizability, or geometry optimization calculation is less than the RMS density matrix element change criterion,<sup>81</sup> whose default value is  $5.0 \times 10^{-6}$ , the calculation is considered to have 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.

---

75. Keyword **ihfgvb** = 0 and **iguess** = 1 in **gen** section of input file.

76. Keyword **iconv** = 1 in **gen** section of input file.

77. Keyword **iconv** = 4 in **gen** section of input file.

78. Keyword **iconv** = 3 in **gen** section of input file.

79. Keyword **maxit** in **gen** section of input file.

80. Keyword **econv** in **gen** section of input file.

81. Keyword **dconv** in **gen** section of input file.

The SCF level shift<sup>82</sup> setting determines the amount that the energies of the virtual orbitals are increased before diagonalization, in atomic units. Level shifting reduces the mixing of the real and virtual orbitals, which slows convergence, but often helps otherwise intractable cases to converge. Useful SCF level shift values are generally in the range 0.3–1.0.

### 4.9.5 Accuracy Level

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. The usual default quick setting<sup>83</sup> allows fast calculations to be performed, using several different pseudospectral grid types, and cutoffs which should generally produce well-converged energies.

The accurate setting,<sup>84</sup> which corresponds to tighter cutoffs (and therefore somewhat slower calculations), also uses a variety of pseudospectral grids. If you choose the ultrafine setting,<sup>85</sup> 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.

For more information on grids and cutoffs, see [Section 10.4 on page 248](#) and [Section 10.5 on page 252](#).

### 4.9.6 Analytic Corrections

For efficiency, Jaguar uses both numerical and analytical methods. The trade-off is that analytic methods are more accurate, but also more time-consuming. Setting analytic corrections to Fully analytic calc.<sup>86</sup> results in a non-pseudospectral calculation, which is significantly slower than the usual method. The default is Standard Corrections,<sup>87</sup> for which the exact number and type of analytically calculated two-electron integrals [105,106] are automatically determined.

---

82. Keyword **vshift** in **gen** section of input file.

83. Keyword **iacc** = 3 in **gen** section of input file.

84. Keyword **iacc** = 2 in **gen** section of input file.

85. Keyword **iacc** = 1 in **gen** section of input file.

86. Keyword **nops** = 1 in **gen** section of input file.

87. Keywords **noatcor** = 0 and **nops** = 0 in **gen** section of input file.



## 4.9.7 Final Localization of the Orbitals

By default, the final wavefunction is not localized.<sup>88</sup> You can localize the valence orbitals after the wavefunction is computed with either the Boys<sup>89</sup> procedure [46] or the Pipek-Mezey<sup>90</sup> procedure [47]. If you choose Boys as a Final localization method, Jaguar localizes the doubly-occupied orbitals by maximizing the term  $\sum_{ij} |\langle \phi_i | r | \phi_j \rangle - \langle \phi_j | r | \phi_i \rangle|^2$ . 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 6.7 on page 133](#) to find out how to print the localized orbitals resulting from either method.

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 fastest 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 7.2 on page 142](#) for information on restart files and restarting jobs.

## 4.9.8 Symmetry

By default, Jaguar takes advantage of molecular symmetry<sup>91</sup> in order to obtain CPU savings. Both Abelian and non-Abelian point groups are recognized. You can turn the use of symmetry off.<sup>92</sup> For information on how to make sure the symmetry of your input structure is treated as you expect, see [Section 3.5.2 on page 37](#).

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

## 4.10 Surfaces

The Calculate Surfaces window allows you to generate electrostatic potential, electron density, spin density, and orbital data that can be visualized using the surface options in Maestro. To generate surface data, you must first run a calculation on the molecule of interest and then read in a restart file. Once you have read in the restart file, the Surfaces button in the Jaguar panel is activated.

You can generate plot data for the electrostatic potential, electron density, electron spin density, and orbitals in the same run. To select multiple orbitals from the list, use SHIFT to

---

88. Keyword **locpostv** = 0 in **gen** section of input file.

89. Keyword **locpostv** = 1 in **gen** section of input file.

90. Keyword **locpostv** = 2 in **gen** section of input file.

91. Keyword **isymm** = 8 in **gen** section of input file.

92. Keyword **isymm** = 0 in **gen** section of input file.

select a range of items and CTRL to select or deselect a single item without affecting other items. The plot data is tabulated on a 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 text boxes at the bottom of the Calculate Surfaces window.

After you make your selection, click RUN. Jaguar launches a job to generate the plot data, and the Monitor panel is displayed. When the job finishes, the surfaces are imported into Maestro and the first surface is displayed. If your molecule is not already an entry in the Project Table, the surfaces are not automatically incorporated. To display the surfaces, select the entry in the Project Table for the chosen molecule, choose Surfaces from the Applications menu, and then choose Import Surface/Volume from the Surfaces menu.

You can view multiple surfaces for the same molecule, but they are superimposed. If you want to view multiple surfaces (e.g., plots for several orbitals) from the same molecule

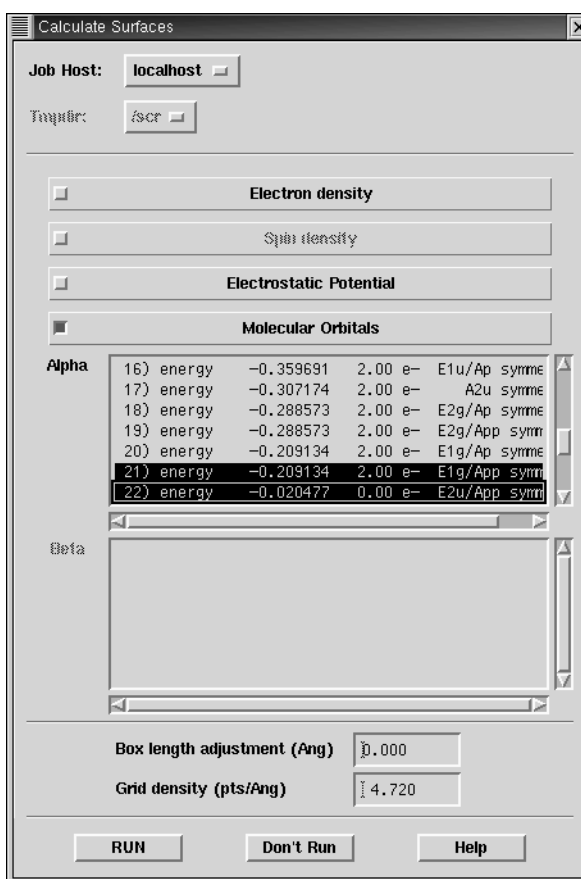


Figure 4.10. The Calculate Surfaces window.

side-by-side, you must duplicate the Project Table entry for the molecule as many times as you have orbitals to view, then create a separate orbital surface for each entry.

To run other calculations using the same input file after running a surface calculation, clear the settings in the Calculate Surfaces window, then click Don't Run to reset the input file settings and close the window without running a job.

## 4.11 J2 Theory Calculations

If you want to calculate energies accurately, you can perform J2 theory calculations [25] using a predefined batch script. The J2 batch script performs a B3LYP/6-31G\* geometry optimization and frequency calculation, followed by single point GVB/LMP2 calculations using the cc-pvtz(-f) and cc-pvtz++ basis sets at the B3LYP optimized geometries. These single-point calculations are used to determine a basis set correction energy. A parameterized electron pair correction energy is also added. The final J2 energy is an absolute enthalpy at 298K. The finite temperature effects are calculated from the B3LYP frequencies. The J2 results do not include a standard heat of formation, because the relevant calculations for the constituent atoms are not made.

To run J2 theory calculations, first create and save an input file containing the desired molecular geometry. In the Jaguar Batch window, click the Select button and select the script `j2.bat` from the list of built-in scripts. Then select the input file or files from the list in the Jaguar Batch window and click RUN. The J2 job is run on the local host.

You can also use the `jaguar j2` command to run the calculation from the command line:

```
jaguar j2 [options] input-files
```

This command executes `jaguar batch`. You can run the calculation on a remote host or in parallel by specifying the relevant command options. See [Section 11.3 on page 275](#) for details on `jaguar batch` commands, including command line options.



---

# Chapter 5: Optimizations and Scans

---

For Hartree-Fock, GVB, 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, while for GVB-RCI calculations, optimizations can be performed using numerical forces.

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 3.9.3 on page 46](#).

## 5.1 Geometry Optimization: The Basics

The Geometry Optimization window contains the settings for optimization of minimum-energy structures or transition states.

The Optimization task is set to none<sup>1</sup> by default, meaning that Jaguar performs a single-point calculation. The energy minimization option<sup>2</sup> requests a search for the molecular geometry with the lowest energy. The calculate forces only<sup>3</sup> option computes the derivatives of the energy for the input structure but does not change the geometry. Another option is transition state search.<sup>4</sup> Settings specific to transition state optimizations are described in a later section of this chapter.

### 5.1.1 Maximum Iterations

An upper limit on the number of steps taken in the geometry optimization sequence can be set in the box marked Maximum iterations.<sup>5</sup> 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 Optimization settings described below.

- 
1. Keyword **igeopt** = 0 in **gen** section of input file.
  2. Keyword **igeopt** = 1 in **gen** section of input file.
  3. Keyword **igeopt** = -1 or keyword **nmdr** = 1 in **gen** section of input file, depending on whether analytical or numerical forces are requested.
  4. Keyword **igeopt** = 2 in **gen** section of input file.
  5. Keyword **maxitg** in **gen** section of input file.

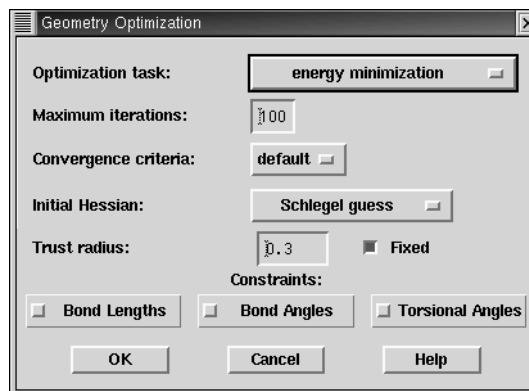


Figure 5.1. The Geometry Optimization window.

## 5.1.2 Geometry Convergence Issues

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 wavefunction is considered converged when the root mean squared change in density matrix elements is less than the RMS density matrix element change criterion,<sup>6</sup> whose default value is  $5.0 \times 10^{-6}$ . (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 Methods window, whose button appears in the Jaguar panel.

For the initial iterations of an optimization, the SCF calculations are performed at the quick accuracy level described in Section 4.9.5 on page 78 (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.

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. These criteria are all affected by the Convergence criteria choices, default<sup>7</sup> or loose;<sup>8</sup> the loose criteria are all five times larger than 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.

6. Keyword **dconv** in **gen** section of input file.

7. Keyword **iaccg** = 2 in **gen** section of input file.

8. Keyword **iaccg** = 3 in **gen** section of input file.

See [Section 9.5.9 on page 179](#) if you want more details on the geometry optimization convergence criteria or information on how to edit the input file to set them directly.

### 5.1.3 The Initial Hessian

To perform an optimization, Jaguar first needs to read or generate an initial Hessian (second derivative matrix or force constant matrix).

You can provide Jaguar with a Hessian in the **hess** section of an input file, as described in [Section 9.9 on page 226](#). For instance, if you restart a geometry optimization from a previous job, as described in [Section 7.2 on page 142](#), Jaguar automatically uses the Hessian provided in your input file.

If your input file does not contain a Hessian, you can use the Initial Hessian option menu in the Optimization window to specify what kind of initial Hessian Jaguar should generate. You can select from among several internal guesses: the Fischer-Almlöf Hessian<sup>9</sup> [50], the Schlegel Hessian<sup>10</sup> [51], or the unit matrix.<sup>11</sup> For most cases, the Schlegel or Fischer-Almlöf options are the best choices. The Schlegel guess is the default.

The final option, quantum-mechanical,<sup>12</sup> is to have Jaguar compute the initial Hessian. This calculation is the most time-consuming of the initial Hessian options. Theoretically, it can be 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.

### 5.1.4 Trust Radius

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 (bohr and/or radians). The trust radius can vary from one iteration to another.

If the trust radius is marked Fixed,<sup>13</sup> the trust radius 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 not fixed<sup>14</sup> (the default for transition state optimizations), Jaguar keeps geometry changes within the region that is well-described by the Hessian by

---

9. Keyword **inhess** = -1 in **gen** section of input file.

10. Keyword **inhess** = 0 in **gen** section of input file.

11. Keyword **inhess** = 1 in **gen** section of input file.

12. Keyword **inhess** = 4 in **gen** section of input file.

13. Keyword **itradj** = 0 in **gen** section of input file.

14. Keyword **itradj** = 1 in **gen** section of input file.

increasing the trust radius when the Hessian is correctly predicting energy changes and decreasing it when the predictions are inaccurate.

At the beginning of a job, the trust radius starts at the value in the box marked Trust radius. For gas phase optimizations to minimum-energy structures, the default initial trust radius is 0.3 atomic units; for solvation calculations or transition state optimizations, the default initial trust radius is 0.1 atomic units.

## 5.2 Constraining Coordinates

To constrain certain coordinates to stay frozen (unchanged) or equal to each other during an optimization, you can use the Geometry Optimization window settings or the Edit window. The first subsection of this section describes how to freeze an entire class of coordinates during an optimization. The other subsections describe ways to constrain individual coordinates by editing the geometry input.

### 5.2.1 Freezing All Bond Lengths, Bond Angles, or Torsional Angles

You can freeze all bond lengths,<sup>15</sup> all bond angles,<sup>16</sup> and/or all torsional angles<sup>17</sup> during an optimization. Jaguar will then keep these coordinates unchanged throughout the job.

To freeze all coordinates of a particular type, first open the Geometry Optimization window. To make Constraints settings from this window, you must first choose energy minimization or transition state search from the Optimization task option menu. You can then turn any of the constraint settings for Bond Lengths, Bond Angles, or Torsional Angles using the buttons at the bottom of the Optimization window.

### 5.2.2 Freezing Specific Coordinates

You can constrain (freeze) specific coordinates in your geometry input<sup>18</sup> to remain fixed at their original values during an optimization. From the Edit Geometry or Edit Job window, you can freeze a specific coordinate by adding a “#” sign at the end of its value in your geometry input.

For example, to fix the HOH bond angle of water at 106.0 ,you could use the following Z-matrix:

- 
15. Keyword **noopt**r = 1 in **gen** section of input file.
  16. Keyword **noopt**a = 1 in **gen** section of input file.
  17. Keyword **noopt**t = 1 in **gen** section of input file.
  18. The geometry input, including constraints (# signs), is in the **zmat** section of the input file.



```
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, if your input for an optimization of a water molecule looked like this:

```
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 would only be allowed to move 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.

### 5.2.3 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 Geometry or Edit Job window, 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.

Whenever Cartesian input with variables is used for an optimization, Jaguar performs the optimization using Cartesian coordinates rather than generating redundant internal coordinates, and the optimization does not make use of molecular symmetry.

## 5.2.4 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 complex structure whose conformation you want to change, and you know that changing a particular torsional angle would cause parts of the molecule to crash into each other 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 166](#).

## 5.3 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.

To set up a transition state search, open the Optimization window and select transition state search<sup>19</sup> from the Optimization task option menu. The Transition state search window is displayed. You should use this window to describe the sort of transition state optimization you want to perform, then return to the Optimization window to set any other optimization features that are not unique to transition state searches.

---

19. Keyword **igeopt** = 2 in **gen** section of input file.

This section describes various transition state search options. For information on general settings that are useful for all types of geometry optimizations, see [Section 5.1 on page 83](#).

### 5.3.1 Transition State Search Method

The first choice listed in the Transition state search window is the Search method, which can be set to `standard`<sup>20</sup> or `QST-guided`.<sup>21</sup> 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 `QST-guided`.

For a `QST-guided` search, you must provide either two geometries, corresponding to the reactant and the product, or three geometries, corresponding to the transition state guess, the reactant, and the product. (The distinction between reactant and product is arbitrary for Jaguar.) [Section 5.3.2 on page 90](#) describes how to specify these structures. If you provide reactant and product geometries, but not a transition state guess, Jaguar generates a transition state guess by interpolating between these two structures; see [Section 5.3.3 on page 90](#) for more details.

For the first few steps of a `QST-guided` search, the optimizer is restricted to search 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.

---

20. Keyword `iqst` = 0 in `gen` section of input file.

21. Keyword `iqst` = 1 in `gen` section of input file.

### 5.3.2 Specifying Different Structures for the Reaction

As mentioned above, for a QST-guided search, you must enter either two geometries, corresponding to the reactant and the product, or three geometries, corresponding to the transition state guess, the reactant, and the product. In either case, 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.

If you are entering three geometries, the transition state guess is the main geometry used, so the reactant and product geometries are labeled Geometry 2 and Geometry 3. If you enter two geometries, the first geometry is the main geometry used, and the second is considered to be Geometry 2. The main geometry for the job is the one used to determine constraints and coordinates. Constraints in Geometry 2 or Geometry 3 are ignored, and the atoms in these geometries must be listed in the same order in which they appear in the main geometry.

To enter these geometries, you can build them in the Maestro Workspace, read them, or create them in the Edit Geometry window. The main geometry needs no special options. To build Geometry 2 or Geometry 3 in the Workspace, click Structure in the Jaguar panel, then select Geometry 2<sup>22</sup> or Geometry 3<sup>23</sup> from the Displayed Structure menu. The Workspace is cleared and you can now build the structure. To read in a geometry as Geometry 2 or Geometry 3, set the Read as option in the Read window to Geometry 2 or Geometry 3. To create or edit one of these geometries in the Edit Geometry window, choose Geometry 2 or Geometry 3 from the Structure menu. If you want to use the same Z-matrix for Geometry 2 or Geometry 3 as you are using for the main geometry, choose Use initial geometry Z-matrix from the Z-matrix menu. You can then set the variables to the desired values.

### 5.3.3 Initial LST Guess

If you provide reactant and product geometries for a QST-guided search, but do not provide a transition state guess, Jaguar generates a transition state guess by interpolating between these two structures.

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<sup>24</sup> 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.

---

22. **zmat2** section of input file.

23. **zmat3** section of input file.

24. Keyword **qstinit** in **gen** section of input file.

### 5.3.4 Searching Along a Particular Path or Eigenvector

If you are using the standard (non-QST-guided) optimization transition state optimization method, you can use the Search along option menu to specify a path for the optimizer to follow or an eigenvector for it to minimize each iteration: the lowest Hessian eigenvector<sup>25</sup> (the default), the lowest non-torsional mode,<sup>26</sup> the lowest bond-stretch mode,<sup>27</sup> the reactant-product path,<sup>28</sup> or a user selected eigenvector.<sup>29</sup>

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. The reactant->product path option causes the optimizer to follow the Hessian eigenvector that is most similar to the direction of the linear path between the reactant and product structures (if you have provided these two structures, but no transition state guess, in your input) or the tangent of the circular curve between the reactant, transition state guess, and product structures (if you have provided all three of these structures in your input).

If you know the 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 setting Search along to user selected eigenvector and specifying the eigenvector number in the text box marked Selected eigenvector.<sup>30</sup> You can identify the eigenvector number by running one geometry optimization iteration (see [Section 5.1.1 on page 83](#) for information) and examining the output summary of the Hessian eigenvectors, which indicates the dominant internal coordinates and their coefficients for each eigenvector.

### 5.3.5 Eigenvector Following

The setting for Eigenvector following determines whether a minimization will follow a new eigenvector each iteration<sup>31</sup> (the default behavior, with eigenvector following off) or whether the transition state optimizer will follow the eigenvector that most closely correlates with the one chosen the previous iteration.<sup>32</sup>

---

25. Keyword **itrvec** = 0 in **gen** section of input file.

26. Keyword **itrvec** = -1 in **gen** section of input file.

27. Keyword **itrvec** = -2 in **gen** section of input file.

28. Keyword **itrvec** = -5 in **gen** section of input file.

29. Keyword **itrvec** > 0 in **gen** section of input file.

30. Keyword **itrvec** > 0 in **gen** section of input file, where **itrvec** is the relevant eigenvector number for the selected eigenvector.

31. Keyword **ifollow** = 0 in **gen** section of input file.

32. Keyword **ifollow** = 1 in **gen** section of input file.

### 5.3.6 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 Jaguar travels on a potential energy surface in its 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 identified 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 5.3.5 on page 91](#)), 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, first choose low-frequency modes from the Hessian refinement option menu in the Transition state search window, which should open when you select transition state search in the Optimization window's first option menu.

Next, you must specify the number of low-frequency Hessian eigenvectors to be used in the refinement, or you must specify one or more input coordinates for refinement. (You can also use both low-frequency modes *and* particular coordinates for a refinement.) [Section 5.3.7](#) explains how to select input coordinates for refinement. If you want to specify a certain number of low-frequency eigenvectors, edit the number in the box marked # of low-freq modes.<sup>33</sup> (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.

---

33. Keyword `irefhup` = 3 in `gen` section of input file.

### 5.3.7 Specifying Coordinates for Hessian Refinement

If you are optimizing a molecular structure to obtain a minimum-energy structure or a transition state, you might want to refine the Hessian used for the job. 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 5.3.6 on page 92](#). The Hessian can be refined in both ways in the same job.

If you put an asterisk (\*) after a coordinate value, Jaguar will compute 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 line at the end of the geometry input that defines the variables. For instance, if you entered either of the following two input geometries in the Edit window:

```
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*
```

they would have the same effect: a job from either input that included Hessian refinement would use both O–H bonds and the H–O–H angle in the refinement.

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 is used for the job.

## 5.4 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. For instance, if you want to perform a “relaxed scan,” finding minimum-energy geometries while holding a particular coordinate fixed to various values, you can set up a geometry optimization input file with a description of the values that coordinate should take.

If you want to vary a particular coordinate for 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, 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.

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.

To scan over a coordinate, set the coordinate with a variable in the geometry input (as described in [Section 3.2 on page 26](#)), then set the variable using one of the options above. For instance, to vary the angle HCCH over the values {0, 30, 60, 90, 120, 150, 180}, you could set it with 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 the “at” 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 the innermost on the 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.

An additional output file with the name `jobname.steps.in` 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.

For each geometry in the scan, the default initial guess is taken from the previous geometry. You can change this behavior using the **igscan** keyword in the **gen** section of the input file.



## 5.5 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 calculations can run in the forward direction (toward the products) and the backward direction (toward the reactants).

IRC scans have been implemented in Jaguar using the methods described in ref. [143]. 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.

To set up a default IRC or minimum energy path calculation, you must first perform a transition state calculation and read in the restart file, then choose intrinsic reaction coordinate<sup>34</sup> or minimum energy path<sup>35</sup> from the Optimization task menu of the Geometry Optimization window.

The calculation also requires a Hessian for the transition state. You can either precalculate the Hessian and read it in from the restart file, or make the calculation of the Hessian part of the IRC or MEP calculation by including the keyword **inhess=4** in the **gen** section of the input file (use the Edit Input window). 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 Methods window). 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.

The direction of the reaction coordinate is not defined by the transition state on its own. You can specify the reactant structure by setting Geometry 2 and the product structure by setting Geometry 3 in the Structure dialog box. You must specify both (or neither). See [Section 5.3.2 on page 90](#) for information on setting these geometries.

The default calculation generates 6 points in both forward and backward directions from the supplied transition state.<sup>36</sup> When the calculation is finished, 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 the *Jaguar Quick Start Guide*.

For information on keywords for IRC calculations, see [Section 9.5.10 on page 185](#).

---

34. Keyword **irc** = 2 in **gen** section of input file.

35. Keyword **irc** = 1 in **gen** section of input file.

36. Keyword **ircmode** = both in **gen** section of input file.



---

# Chapter 6: Output

---

The output from a Jaguar run always includes a Jaguar output file, which contains the primary output, and a log file, which contains a job summary that is updated as the job is being run. If you request other output options in the Files window, various other files can also be generated as output.

This chapter begins with a description of the Jaguar output file for a standard Hartree-Fock calculation, and continues with a discussion of the changes in the output for various other calculation options and the output options that can be set from the Standard, Files, Per Iteration, and Orbitals windows found in the Output section. The final section explains the log file, which is the file displayed in the Monitor panel as a job runs.

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.

## 6.1 Summarizing Jaguar Results

You can obtain summaries of Jaguar results in simple table form by using 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
<i>Job and Molecular Information Options:</i>	
-jobname	job name
-longjobname	job name, with wider output
-stoich	stoichiometry
-weight	molecular weight
-basis	basis set
-nbasis	number of basis functions
-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
-s2	spin: $\langle S^2 \rangle$
-sz2	spin: $S_z \langle S_z + 1 \rangle$
-method	SCF/post-SCF method
<i>Energy-related Options:</i>	
-energy	final molecular energy
-enuc	nuclear repulsion energy
-egas	gas-phase energy
-esoln	solution-phase energy
-esolv	solvation energy
-esolute	solute energy
-esolvent	solvent energy
-ereorg	solvent reorganization energy
-homo	HOMO energies

Table 6.1. Options for the jaguar results Command (Continued)

Option	Meaning
-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 (Cv) at 298.15 K
-int_energy	internal energy (U) at 298.15 K
-Utot	Total internal energy (Utot) at 298.15 K, including the SCF energy and zero-point energy
-Htot	Total enthalpy (Htot) at 298.15 K, including the SCF energy and zero-point energy
-Gtot	Total Gibbs free energy (Gtot) at 298.15 K, including the SCF energy and zero-point energy
-pka	pKa
-pkb	pKb
-dipole	total dipole moment
<i>Geometry Optimization Options:</i>	
-iterg	geopt iteration number
-stepg	geopt step number
-zvar <i>name</i>	z-variable value (must be followed by zvar name)
-grms	rms gradient
-gmax	maximum gradient component
-drms	rms displacement
-dmax	maximum displacement
-echange	energy change
<i>Timing Options:</i>	
-time	total cpu time for job
-tscf	total time in scf (cumulative)
-trwr	total time in rwr (cumulative)

Table 6.1. Options for the `jaguar results` Command (Continued)

Option	Meaning
<code>-tder1b</code>	total time in <code>der1b</code> (cumulative)
<i>SCF Information Options:</i>	
<code>-iter</code>	number of <code>scf</code> iterations
<i>Per-atom Information Options:</i>	
<code>-atoms</code>	atom names
<code>-atomnums</code>	atomic numbers
<code>-coords</code>	cartesian atomic coordinates
<code>-forces</code>	cartesian atomic forces
<code>-charges</code>	ESP atom-centered charges
<i>Options for Printing of Title and Intermediate Results:</i>	
<code>-title</code>	print column titles
<code>-titleonly</code>	print only the column titles
<code>-all</code>	report results every geometry iteration
<code>-allscf</code>	report results for each <code>scf</code> calculation
<code>-allder1b</code>	report results for each <code>der1b</code>

### 6.1.1 Reporting Final Results From One or More Jobs

By default, each row of the Jaguar results table (except the title row) corresponds to the final results from a Jaguar output file that was listed 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`, the table has column headings indicating the type of information listed. The columns appear in the table in the same 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 -exchange -gmax -grms \
-dmax -drms dftg.out
```

here produces a table showing the convergence of a BLYP geometry optimization of water:

Geopt iter	Energy [change]	Gradient [max]	Gradient [rms]	Displace. [max]	Displace. [rms]
===== 1	===== 2	===== 3	===== 4	===== 5	===== 6
1	-2.04E-03 .	3.22E-02 .	2.65E-02 .	5.53E-02 .	4.88E-02 .
2	-7.04E-05 .	3.85E-03 .	3.18E-03 .	2.79E-02 .	1.70E-02 .
3	-1.04E-06 #	4.19E-04 *	3.82E-04 .	1.45E-03 *	1.01E-03 *
4		3.05E-05 #	2.52E-05 #	6.13E-05 #	5.13E-05 #

The last section of [Table 6.1](#) lists the options that let you specify 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 106](#) describes the variations in the output file for the calculation options described in [Chapter 4](#).

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 which 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: its job name, the directory containing the executables used to run the job, the directory containing the temporary files, comments from the input file (if any), and the names and paths of any non-default data files used for the calculation (as explained in [Section 9.1 on page 161](#) and [Chapter 10](#)). Comments from the input file include any text entered in the Comment box in the Run or Save window, as well as a comment about the point group if the geometry was symmetrized as described in [Section 3.5.2 on page 37](#).

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 calcu-



lation. Canonical orbital eigenvectors with very small eigenvalues (less than  $5.0 \times 10^{-4}$ ) 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 wavefunction (initial guess) for a Hartree-Fock calculation. The output from the program `hfig` for a default calculation begins with the line, “initial wavefunction generated automatically from atomic wavefunctions.” 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 wavefunction generated automatically from atomic wavefunctions

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
```

The `probe` program, which follows `hfig` and insures 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 248](#). 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
      e d i u i
      r t s t d
      total energy
      energy change
      RMS density change
      maximum DIIS 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 the number of electrons in the molecule, the number of alpha and beta electrons, the total number of orbitals for the calculation, the numbers of core, open shell, occupied, and virtual orbitals, the number of Hamiltonians used for the calculation, the number of 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 252](#).

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 9.5.23 on page 210](#) and [Section 10.4 on page 248](#) 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  $\mathbf{FDS} - \mathbf{SDF}$  in atomic orbital space, where  $\mathbf{F}$ ,  $\mathbf{D}$ , and  $\mathbf{S}$  are the Fock, density, and overlap matrices, respectively. For open shell and GVB 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. The nuclear repulsion, one-electron, two-electron, and electronic contributions are all listed, followed by the total. 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. 107. 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 Calculation Options

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 4](#).

Generally, only the *format* changes that result from these settings are discussed below. Naturally, these settings will often change the data listed. For information on the settings themselves, see [Chapter 4](#). 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 (GVB-DFT calculations, for instance) first lists the standard output for the HF, GVB, 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 wavefunction, 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.

### 6.3.2 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 `loc_lmp2`, 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 your choice from the LMP2 window of whether to 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 9.5.19 on page 204](#).)

### 6.3.3 GVB

If a GVB calculation is performed from a Hartree-Fock initial guess, the `pre` program output lists a table of GVB pair information below the list of non-default options. The information in the table includes whether a restricted configuration interaction (RCI) calculation including that pair will be performed (Y or N for yes or no), and what the configuration interaction (CI) coefficients are for the pair. Since the use of symmetry is turned off automatically for GVB calculations, the output from the programs `pre` and `hfig` is somewhat different than for a Hartree-Fock calculation. Also, the program `gvbig` runs after `hfig`, if the GVB initial guess is being generated from the HF initial guess.

The output from the `scf` program is more extensive than for a default HF calculation. First, the number of GVB pairs and the number of GVB orbitals are added to the list of electron and orbital information preceding the table of SCF iteration information. Secondly, the summary of the SCF output is followed by a breakdown of the two-electron contribution to the energy into Coulomb and exchange parts. For each of these parts, the contribution from each GVB Hamiltonian is listed. After this information, the intra-pair exchange energies and their sum are listed. Finally, a table of GVB pair information is given. Here is an example of this GVB information in the SCF output, for a water molecule with two GVB sigma pairs:

	Total	Coulomb	Exchange
Total two-electron terms	37.90378136033	46.96140169504	-9.05762033471
Hamiltonian 1.....	25.77166631229	32.84704880440	-7.07538249211
Hamiltonian 2.....	6.02807668738	6.99023521309	-0.96215852571
Hamiltonian 3.....	6.02990515066	6.99271668375	-0.96281153309
Hamiltonian 4.....	0.03711925295	0.06576591758	-0.02864666463
Hamiltonian 5.....	0.03701395705	0.06563507622	-0.02862111917

## List of Intra-Pair K Energies

```
-0.03983705429    -0.03981442075
Sum of Intra-Pair K Energy...    -0.07965147505
```

## GVB pair information:

pair	first natural orbital				second natural orbital				overlap	ci energy lowering
	orb	ham	shl	ci coeff.	orb	ham	shl	ci coeff.		
1	4	2	2	0.995433818	6	4	4	-0.095454256	0.824997160	0.020103338
2	5	3	3	0.995443725	7	5	5	-0.095350881	0.825171705	0.020091467

```
SCFE: SCF energy: GVB    -76.06328826029 hartrees    iterations:    8
```

Each row in the GVB pair information table lists the pair number, the orbital number (after all core and open orbitals have been assigned numbers), Hamiltonian number (after the core Hamiltonian and any open Hamiltonians have been assigned numbers), and shell number (after the core shell and any open shell have been assigned numbers) corresponding to each natural orbital, and CI coefficient corresponding to each GVB natural orbital in the pair. Next, the overlap between the two corresponding non-orthogonal orbitals for that pair is listed, followed by the CI energy lowering, which is a guide to the energy change resulting from the inclusion of the second natural orbital in the calculation.

If a GVB calculation is performed from a Hartree-Fock converged wavefunction, the program `scf` runs twice, once to obtain the HF converged wavefunction, and once to perform the final GVB calculation. The SCF output from the first `scf` run will look like the SCF output from a standard HF calculation; the output from the second run will have the format described above for a GVB calculation from an HF initial guess.

### 6.3.4 GVB-RCI

For restricted configuration interaction calculations, the SCF output is the same as for non-RCI GVB calculations, but the output from the program `rci` appears after the SCF output. The RCI output first lists information on the total number of orbitals, the number of core orbitals for the RCI calculation, the numbers of open shell and GVB orbitals, the number of GVB and RCI pairs, the numbers of occupied and virtual orbitals, the numbers of Coulomb and exchange Hamiltonians, and the multiplicity. Next, the total energy of the

GVB wavefunction, which was obtained from the SCF procedure earlier, is listed and broken down into a nuclear repulsion term and terms from the core, which is treated with Hartree-Fock methods, the GVB pairs, and the open shell contribution. If you specified GVB pairs that were not also RCI pairs, a non-zero value is listed for the non-RCI GVB pair energy. The number of RCI spatial configurations and the number of RCI configuration state functions follow. Each RCI configuration state function is the product of a contracted spatial function and a spin function; the number of these functions indicates the size of the RCI expansion.

The `rci` output lists the RCI initial guess energy next, followed by the converged total RCI energy. For small cases, the initial guess and the converged energy may agree exactly, since the RCI coefficients are obtained by one diagonalization of a small matrix. For larger cases, the output includes the results of the iterative diagonalization.

### 6.3.5 Geometry or Transition State Optimization (HF, GVB, DFT, and LMP2)

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 4.9 on page 74](#) instead of the usual energy convergence criterion. Therefore, these SCF calculations often proceed for several more iterations than single point energy calculations yield.

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.

If `Optimize geometry` is set to minimum energy or transition state, 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 5.2 on page 86](#), 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, a process that 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 as well. 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 from the Optimization window, as described in [Section 5.1 on page 83](#), or, alternatively, the criteria set by the `gconv` keywords in the input file. The criteria are described in detail in [Section 9.5.9 on page 179](#). If the gradients are converged and the energy change is below  $2.5 \times 10^{-7}$ , 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  $2.5 \times 10^{-6}$ .

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.

### 6.3.6 Optimizations With GVB-RCI Wavefunctions

Geometry or transition state optimizations using GVB-RCI run in much the same way as described above for HF, GVB, or DFT optimizations, except that the forces for the optimization are computed numerically rather than analytically. Consequently, the `der1a` and `der1b` programs never run; instead, when forces are needed, the structure's energy is evaluated at  $6N_{\text{atom}}$  perturbed geometries, where  $N_{\text{atom}}$  is the number of atoms, and the forces are computed numerically. The program `nude` generates each perturbed geometry by moving an atom a small amount in the positive or negative x, y, or z direction, and also evaluates the numerical derivatives when calculations on all perturbed geometries are complete, listing them in a force table similar to the usual geometry optimization force table described for HF, GVB, or DFT systems. The program `geopt` still runs in the usual way as well, computing each iteration's new geometry using the available forces.

### 6.3.7 Solvation

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 wavefunction 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 energy for the molecule converges.

For solvation calculations on neutrally charged 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 253](#).) 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 218](#), 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 6.3.9 on page 116](#), 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  $\text{\AA}^2$ ) 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  $\text{\AA}^2$ ), 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.

After this output, 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. An example, from the first solute-with-solvent-point-charges `scf` run in a calculation of 6-31G\*\* water in cyclohexane, using the Jaguar solver, is given here:

```

start of program scf

      i u d i g
      t p i c r
      e d i u i
      r t s t d      total energy      energy      RMS      maximum
                                change      density      DIIS
                                change      change      error

etot  1  N  N  2  U  -76.03588607997      6.8E-04  6.6E-03
etot  2  Y  Y  6  M  -76.03615425936  2.7E-04  1.9E-04  1.8E-03
etot  3  Y  N  6  M  -76.03617415619  2.0E-05  0.0E+00  0.0E+00

```

Energy components, in hartrees:

```

(A) Total zero-electron terms....      9.35161183359
(B)   Nuclear-nuclear.....            9.33000672144
(C)   Nuclear-solvent.....            0.02160511215
(E) Total one-electron terms....     -123.39806065860
(F)   Electron-nuclear.....     -199.21812919134
(G)   Electron-solvent.....      -0.03443064237
(H)   Kinetic.....                75.85449917511
(I) Total two-electron terms....      38.01027466882
(L) Electronic energy.....           -85.38778598978  (E+I)
(N) Total quantum mech. energy...    -76.03617415619  (A+L)
(O) Gas phase energy.....           -76.02364072535
(P) Solution phase energy.....      -76.02607108661  (Q+R+S)
(Q) Total solute energy.....        -76.02334862596  (N-C-G)
(R) Total solvent energy.....       -0.00641276511  (C/2+G/2)
(S) Solute cavity energy.....         0.00369030447
(U) Reorganization energy.....        0.00029209939  (Q-O)
(V) Solvation energy.....           -0.00243036126  (P-O)

```

SCFE: SCF energy: HF -76.03617415619 hartrees iterations: 3

HOMO energy: -0.49985

LUMO energy: 0.22469

```

Orbital energies/symmetry label:
-20.55803 A1      -1.34624 A1      -0.71287 B2      -0.57176 A1
-0.49985 B1       0.22469 A1       0.31901 B2       1.01892 B2
 1.09275 A1       1.13045 A1       1.16509 B1       1.29393 B2
 1.41452 A1       1.80375 A2       1.82851 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. 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             -0.0024304        -1.5251
  2             -0.0027473        -1.7240
  3             -0.0027918        -1.7519

stopping: solvation energy converged

iterations:   3      sfinal:    -1.7519 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 4.5 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.

### 6.3.8 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 wavefunction 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.

### 6.3.9 Properties

If you make any non-default selections from the Properties window, the program `ch` runs and writes the results to the output file after the SCF iterations, if any.

When multipole moments are calculated, the x, y, and z components of the dipole moment and the total magnitude of the dipole moment  $\mu$  are reported in Debye, followed by information on any requested higher-order moments and the corresponding traceless higher-order moment tensors. For example, here is the output for a calculation of water's dipole and quadrupole moments:

```

Moments from quantum mechanical wavefunction:
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 of the output from such a calculation 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 octapole 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, a new moment (or moments) can be calculated from the fitted charges, as described in [Section 4.6.1 on page 60](#). The output from `ch` begins with the moment or moments calculated for the quantum mechanical wavefunction, 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 moment or moments recalculated using the electrostatic potential charges are reported.

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 4.6 on page 60](#) for details on the methods used to calculate polarizability and hyperpolarizability.) The data from the program `scf` includes the virial ratio  $-V/T$ . 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  $\alpha$  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  $\beta$  is defined by

$$\beta = \frac{3}{5\mu} (\mu_x \Sigma\beta_x + \mu_y \Sigma\beta_y + \mu_z \Sigma\beta_z)$$

where  $\mu$  is the dipole moment. The average second hyperpolarizability  $\gamma$  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  $z$ .

If you choose to calculate the electron density, the output from the program `eldden` 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.

If you calculate Mulliken populations by atom, the charge for each atom and the sum of the atomic charges will be noted under the heading “Atomic charges from Mulliken population analysis.” If you choose to calculate them by basis function, the atomic charge output will be preceded by a section labeled “Mulliken population for basis functions,” listing the atom label, function (labeled with consecutive numbers), type of basis function (S for s, X for p<sub>x</sub>, XX for d<sub>xx</sub>, etc.), and calculated population. Calculating Mulliken populations by bond yields the populations by atom and basis function as well. An example of this output for a calculation of water using the 6-31G\*\* basis set is provided below.

```
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 from the Standard window, whose button appears under the Output heading, if you want to have more information about the basis functions. More information on this output option is given in [Section 6.4 on page 124](#).

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 wavefunction, as noted in [Section 4.6.5 on page 64](#). The output lists the multipole moments from the wavefunction, 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.

Output for NBO calculations appears under the heading “Jaguar NBO 5.0.”

### 6.3.10 Frequency, IR Intensity, and Thermochemistry Output

If you calculate vibrational frequencies by making the appropriate setting in the Frequencies window, any SCF calculations during the run use the RMS density change convergence criterion described in [Section 4.9 on page 74](#) instead of the usual energy convergence criterion. Therefore, these SCF calculations often proceed for several more iterations than single point energy calculations yield.

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 4.7 on page 64](#) 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  $6N_{\text{atom}}$  perturbed geometries (unless the number of perturbed geometries needed is reduced by the use of molecular symmetry). To make each perturbed geometry, one atom is moved a small, fixed amount in the positive or negative direction along the x, y, or z Cartesian axes. 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` outputs 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 7.2 on page 142](#).)

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  $\text{cm}^{-1}$  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), and Gibbs free energy, 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      226.83   587.12   708.27   994.11  1063.07  1086.36
intensities      .69      .00      6.20     6.85    56.95    18.20
reduc. mass      5.96     4.34     4.66     7.85    4.37     4.40
force const      .18      .88      1.38     4.57    2.91     3.06
F1      X      .03713  -.07085  -.05263  .01504  .10332  .10413
F1      Y      .07738  -.08093  -.07570  .00294  -.00858  -.01121
F1      Z      .11257  .01960  .00381  .00291  .00007  .00142
O2      X      .03630  -.04451  -.02646  -.05071  -.13148  -.12893
O2      Y     -.00653  -.00113  .11356  -.16823  .01169  .04865
O2      Z     -.07806  -.12275  .08708  .00749  .00934  -.01566
O3      X      .07467  .12140  .11117  .03829  .00279  .01163
O3      Y     -.02320  -.01992  .07655  .17175  -.02320  -.01800
O3      Z     -.03681  .04285  -.05472  .00584  -.13452  .13276
F4      X     -.13055  .00611  -.01869  -.00459  .00502  -.00537
F4      Y     -.05236  .09865  -.08436  -.00590  .01826  -.01460
F4      Z     -.01586  .04767  -.03106  -.01413  .10532  -.10001

Thermochemical properties at      1.0000 atm

rotational symmetry number: 1

rotational temperatures (K):   1.075181   .283392   .256070

vibrational temperatures:
mode:      1      2      3      4      5      6
temp. (K):  326.36  844.73  1019.03  1430.30  1529.51  1563.02

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): 6.670 kcal/mol  
is not included in U, H, or G in the table below

T = 298.15 K

	U	Cv	S	H	G
	-----	-----	-----	-----	-----
trans.	.889	2.981	38.655	1.481	-10.044
rot.	.889	2.981	23.636	.889	-6.158
vib.	.558	4.662	2.907	.558	-.309
elec.	.000	.000	.000	.000	.000
total	2.335	10.623	65.198	2.928	-16.511

Total internal energy, Utot (SCFE + ZPE + U): -348.203415 hartrees  
Total enthalpy, H (Utot + pV): -348.202471 hartrees  
Total Gibbs free energy (H - T\*S): -348.233448 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 described above.

### 6.3.11 Basis Set

If your calculation uses a basis set that includes effective core potentials, the output lists the number of atoms treated with effective core potentials.

### 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 4.8 on page 70](#)), 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 Orbitals window, as described in [Section 6.7 on page 133](#).)

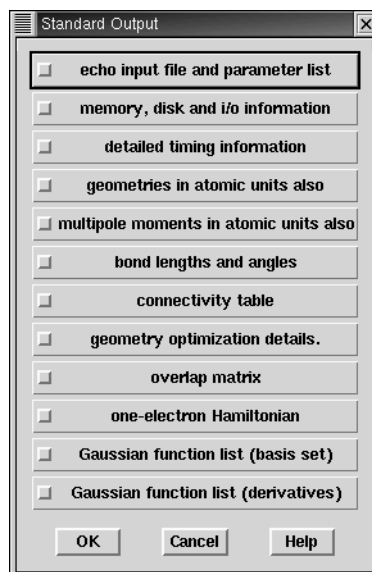


Figure 6.1. The Standard Output window.

## 6.4 Standard Output Options

The menu options from the Standard output window are described in this section. The output generated from these options appears in the output file for the job. If you make a non-default setting from the Standard output window, the output from the program `pre` prints the non-default options chosen. This output appears above the molecular geometry output from the same program, and indicates 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.<sup>1</sup> This option is likely to be useful primarily for people who have a detailed knowledge of the code itself.

### 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.”<sup>2</sup> Current and maximum

1. **echo** section constructed, and keywords **mttest** = 2 and **ip24** = 2 in **gen** section of input file.
2. Keyword **ip5** = 2 in **gen** section of input file.

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 seconds spent in various Jaguar programs is listed in the output.<sup>3</sup>

#### geometries in atomic units also

This option allows you to choose to print the geometry output in atomic units as well as in the default units, Angstroms.<sup>4</sup>

#### multipole moments in atomic units also

If you choose to calculate multipole moments by making the appropriate setting in the Properties window, this option allows you to choose to list them in the output file in atomic units as well as in the default units, Debye.<sup>5</sup>

#### 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.<sup>6</sup> The atoms are indicated with the atom labels assigned in the geometry input. When the Optimize geometry option in the Geometry Optimization window has been turned on, the bond lengths and angles standard output 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.<sup>7</sup> 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 243](#) 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.

- 
3. Keyword **ip6** = 2 in **gen** section of input file.
  4. Keyword **ip26** = 2 in **gen** section of input file.
  5. Keyword **ip25** = 2 in **gen** section of input file.
  6. Keyword **ip11** = 2 in **gen** section of input file.
  7. Keyword **ip12** = 2 in **gen** section of input file.

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 9.5.1 on page 168](#). 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 the **geometry optimization details**<sup>8</sup> option is selected, much additional information about the progress of a geometry optimization is printed. This output often helps reveal the cause of any problems with optimizations.

#### overlap matrix

The overlap matrix **S** for the basis functions is printed in five-column blocks if this option is selected.<sup>9</sup> 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.<sup>10</sup> Since the matrices are symmetric, the upper triangles are not printed.

#### Gaussian function list (basis set)

By turning this option on, you can choose to print out information about the Gaussian functions that make up the basis set.<sup>11</sup> 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  $N e^{-zr^2}$ , where  $N$  is a normalization constant and  $z$  is the exponent of the primitive. If the linear combination only includes 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 output controlled by this option gives essentially the same information about the basis functions in two different tables, after giving a list of atoms and the basis set used for each one.

The shell information table is printed first. An example, for a calculation of water with a 6-31G\*\* basis set, is given below. The first column of the table indicates which atom the

---

8. Keyword **ip192** = 2 in **gen** section of input file.

9. Keyword **ip18** = 2 in **gen** section of input file.

10. Keyword **ip19** = 2 in **gen** section of input file.

11. Keyword **ip1** = 2 in **gen** section of input file.



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. For instance, 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 two in the next two 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 should be used in the shell listed that number of rows down; a value of `-1` indicates that the exponents should be 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 one less than the lowest number corresponding to the basis function or functions being described. For example, the `nfsh=2` entries below are for p functions, so the fourth and fifth basis functions are generated in the same way as the third, but with different polynomials.

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

atom	s	j	h c i n			z	coef	rcoef
	l	t	l	l	h			
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 basis functions themselves—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

atom	l	s	h	t	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.

#### Gaussian function list (derivatives)

By turning this option on, you can choose to print out information about the derivatives of the basis set functions in terms of primitive Gaussians.<sup>12</sup> The format and information is the same as that discussed for the Gaussian function list (basis set) option immediately above.

## 6.5 File Output Options

This section describes the options in the File Output window, which you open using the File button in the Output section of the Jaguar panel. 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 from the File Output window, the output from the program `pre` lists the non-default options chosen. This output appears above the molecular geometry output from the same program, and indicates the non-default values of the keywords referred to in footnotes throughout this section.

---

12. Keyword `ip8 = 2` in `gen` section of input file.

#### Gaussian-92 input deck (.g92)

When this option is selected, a file in the format of a GAUSSIAN 92 input file is created, with the suffix `.g92`.<sup>13</sup> The file information includes the molecular geometry, the basis set name, and 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 `.g92` file.

For GVB calculations, you should specify GVB pairs; Jaguar will also generate a GVB initial guess, which will be included in the `.g92` file. For more information on setting up GAUSSIAN 92 input files, see [Section 7.3 on page 143](#).

#### GAMESS input file (.gamess)

To write out an input file for the program GAMESS, you can select this option.<sup>14</sup> 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`.<sup>15</sup>

#### Gaussian-92 basis set (.gbs)

If this option is turned on, a `.gbs` file will be generated containing the basis set in a form that can be used by GAUSSIAN 92.<sup>16</sup>

#### XYZ file (.xyz)

If you set this option, Jaguar creates a file in XYZ format with the suffix `.xyz`.<sup>17</sup> 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.

---

13. Keyword **ip160** = 2 in **gen** section of input file.

14. Keyword **ip168** = 2 in **gen** section of input file.

15. Keyword **ip165** = 3 in **gen** section of input file.

16. Keyword **ip163** = 2 in **gen** section of input file.

17. Keyword **ip175** = 2 in **gen** section of input file.

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 [108].<sup>18</sup> If you have run a frequency calculation, the normal modes are written to the .molf file.

## 6.6 Output Options Per Iteration

Some output can be printed out every SCF iteration by choosing options from the Per Iteration Output window, which you open using the Per Iter. button in the Output section of the Jaguar panel. The output described in this section appears in the output file. For each SCF iteration where the described output appears, that output is listed before the usual energy data for that iteration.

Any non-default settings from the Per Iteration Output window cause the output from the program `pre` to list the non-default options chosen. This output appears above the molecular geometry output from the same program, and indicates the keywords referred to in footnotes throughout this section.

energy components

When this output option is off, the individual components contributing to the total energy are only printed for the final, converged result of the SCF iterations. When the option is turned on, the output includes each iteration's energy components: namely, the nuclear repulsion term, the total one-electron terms, the total two-electron terms, the electronic energy, and the total energy.<sup>19</sup> The orbital energies for the occupied orbitals are also provided for each iteration.

The Coulomb and exchange contributions to the total two-electron energy are printed as well if the J and K matrices are kept separate for the calculation, as for GVB calculations and when the Core J and K option in the Methods window is turned on. In addition, for most calculations involving open shells or higher-level methods, the individual contributions from each Hamiltonian are printed for the Coulomb and exchange terms.

If the calculation involves solvation, the nuclear-electronic and kinetic terms making up the one-electron terms are also listed, as well as the term  $-V/T$  (where  $V$  is the potential energy and  $T$  the kinetic energy) and the various contributions to the solvation energy.

---

18. Keyword **ip90** = 2 in **gen** section of input file.

19. Keyword **ip17** = 2 in **gen** section of input file.

#### density matrix

If you select this option, the density matrix in atomic orbital space is printed out for each iteration.<sup>20</sup> For iterations in which Fock matrix updating is performed, the change in the density matrix from the previous iteration is printed instead of the density matrix itself. The output from the program `scf` indicates whether Fock matrix updating was performed or not in any particular iteration.

#### All J and K matrices, AO space

The Coulomb and exchange matrices in atomic orbital space can be printed out for each iteration by selecting this option.<sup>21</sup> However, by default the calculation will be performed by combining these matrices in the form  $2J - K$ , and they may not be properly separated here if this is the case. In order to print out the true J and K matrices, you must insure that the Core J and K option in the Methods window, whose button is found in the main window, specifies that the matrices be kept separate. For GVB, DFT, LMP2, and GVB-LMP2 calculations, the J and K matrices are kept separate by default.

Since J and K are symmetric matrices, the elements of the upper triangles are not printed.

#### Fock matrix in AO (HF) or MO (GVB) space

The Fock matrix in atomic orbital space (for HF or DFT calculations) or molecular orbital space (for GVB calculations) can be printed by turning this option on.<sup>22</sup> This information is only printed for iterations where the Fock matrix is not updated. Because the Fock matrix is symmetric, the elements of the upper triangle are not printed.

#### Fock matrix in CO space

The Fock matrix in canonical orbital space can be printed by turning this option on.<sup>23</sup> Because the Fock matrix is symmetric, the elements of the upper triangle are not printed.

#### GVB data: f, a, b, ci coefficients, etc.

You may print out GVB data for the initial guess and the GVB initial guess by selecting this option.<sup>24</sup>

---

20. Keyword **ip110** = 2 in **gen** section of input file.

21. Keyword **ip121** = 2 in **gen** section of input file.

22. Keyword **ip122** = 2 in **gen** section of input file.

23. Keyword **ip123** = 2 in **gen** section of input file.

24. Keyword **ip149** = 2 in **gen** section of input file.

## 6.7 Output Options for Orbitals

Orbital information can be printed to the output file as well. Several possible choices are available in the Orbitals window, whose button is found in the Output section, for what, when, and how orbitals should be printed. 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 the same program, and indicates the keywords referred to in footnotes throughout this section.

When:

The following When menu options determine the point at which orbitals are printed out.

- after HF initial guess  
Print orbitals used for the HF initial guess.<sup>25</sup>
- after GVB initial guess  
Print orbitals used for the GVB initial guess.<sup>26</sup>
- each iteration (in CO space)  
Print orbitals after each SCF iteration in canonical orbital space.<sup>27</sup> (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 4.8 on page 70](#).
- each iteration (in AO space)  
Print orbitals after each SCF iteration in atomic orbital space.<sup>28</sup>
- after SCF  
Print orbitals in atomic orbital space after the SCF converges.<sup>29</sup>
- after final localization  
Print orbitals after the localization procedure, if Boys or Pipek-Mezey localization of the wavefunction has been requested.<sup>30</sup>
- at end of job  
Print the orbitals at the end of the job.<sup>31</sup>

---

25. Keyword **ip105** in **gen** section of input file.

26. Keyword **ip106** in **gen** section of input file.

27. Keyword **ip101** in **gen** section of input file.

28. Keyword **ip103** in **gen** section of input file.

29. Keyword **ip104** in **gen** section of input file.

30. Keyword **ip107** in **gen** section of input file.

**What:**

By default, no orbitals are printed in the output file, so the selection none appears in the What: option menu.<sup>32</sup> If you select occupied orbitals, all occupied orbitals, including GVB natural orbitals, are printed.<sup>33</sup> If the all orbitals option is selected, all occupied orbitals and ten virtual orbitals are printed.<sup>34</sup> (To change the default of ten virtual orbitals, see the information on the keyword **ipvirt** in Section 9.5.22 on page 208. 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).) Selection of GVB orbitals (nonorthog.) prints only the GVB non-orthogonal orbitals.<sup>35</sup>

**How:**

The choices available for how to print the selected orbitals are:

- large elements as f5.2, labels, in list,<sup>36</sup>
- all elements as f10.5, labels, in table,<sup>37</sup>
- all elements as f19.15, in list,<sup>38</sup>
- all elements as f8.5, in list,<sup>39</sup>
- all elements as e15.6, in table.<sup>40</sup>

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<sub>z</sub>, or XX for d<sub>xx</sub>) are shown.

---

31. Keyword **ip102** in **gen** section of input file.

32. This setting corresponds to having all of the orbital output keywords set to 1.

33. Relevant orbital output keyword set to 2, 3, 4, 5, or 6 in **gen** section of input file, depending on the format setting chosen.

34. Relevant orbital output keyword set to 7, 8, 9, 10, or 11 in **gen** section of input file, depending on the format setting chosen.

35. Relevant orbital output keyword set to 12, 13, 14, 15, or 16 in **gen** section of input file, depending on the format setting chosen.

36. Relevant orbital output keyword set to 2, 7, or 12 in **gen** section of input file, depending on which orbitals are printed.

37. Relevant orbital output keyword set to 3, 8, or 13 in **gen** section of input file, depending on which orbitals are printed.

38. Relevant orbital output keyword set to 4, 9, or 14 in **gen** section of input file, depending on which orbitals are printed.

39. Relevant orbital output keyword set to 5, 10, or 15 in **gen** section of input file, depending on which orbitals are printed.

40. Relevant orbital output keyword set to 6, 11, or 16 in **gen** section of input file, depending on which orbitals are printed.



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.

Because GVB orbitals are not computed until some time after the Hartree-Fock initial guess, you cannot choose to print GVB non-orthogonal orbitals if you have selected after HF initial guess above. Also, note that in canonical orbital space, the labels indicating atom identifiers and basis function types are meaningless.

If you generate output for occupied orbitals or all orbitals 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 greater detail in [Section 9.10 on page 227](#), or for input to GAUSSIAN 92 (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, where the option requested under **When** was after SCF iterations and the option requested under **What** was occupied orbitals. Only the first two occupied orbitals are shown in each case, and not all functions are shown; these gaps are indicated by [...].

For the How option large elements as f5.2, labels, in 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
3 Orbital Energy   -0.713206 Occupation  1.000000 Symmetry B2
[...]
```

For the How option all elements as f10.5, labels, in table:

eigenvalues-		1	2	3
1 O	S	-20.55513	-1.34560	[...]
2 O	S	0.99466	-0.21055	
[...]				
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 How option all elements as f19.15, in list:

```

  1 Orbital Energy   -20.555133 Occupation  1.000000 Symmetry A1
    0.994661070265476  0.021223773328496  0.0000000000000000  0.0000000000000000
    0.001550431863529  0.004301782758377  0.0000000000000000  0.0000000000000000
  -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
  3 Orbital Energy   -0.713206 Occupation  1.000000 Symmetry B2
  [...]

```

For the How option all elements as f8.5, in 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
  3 Orbital Energy   -0.713206 Occupation  1.000000 Symmetry B2
  [...]

```

For the How option all elements as e15.6, in 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	

## 6.8 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 252](#).

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 9.5.23 on page 210](#) and [Section 10.4 on page 248](#), 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  $\mathbf{FDS} - \mathbf{SDF}$  in atomic orbital space, where  $\mathbf{F}$ ,  $\mathbf{D}$ , and  $\mathbf{S}$  are the Fock, density, and overlap matrices, respectively. For open shell and GVB 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.



---

# Chapter 7: Tips and Suggestions

---

This chapter includes information on restarting jobs, and using Maestro to help set up GAUSSIAN 9x jobs, as well as some extra suggestions for GVB calculations, geometry optimization, electrostatic potential fitting, and jobs involving transition metals.

## 7.1 Tips for Various Types of Jobs

This section contains information you may find useful for improving SCF convergence, running GVB jobs and optimizations, and fitting charges.

### 7.1.1 Organometallics and Other Difficult-to-Converge Systems

Generally, Hartree-Fock wavefunctions 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 Methods window.

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:

- Try setting `iacscf` to 1, 2, 3, or 4 (see [Table 9.27 on page 194](#) for descriptions of each number's function). You might need to increase the setting of `maxit` to 100 or more when using `iacscf` values of 1, 2, or 4.
- Select GVB-DIIS from the Convergence scheme option menu in the Methods window. Generally, DIIS is the better choice, but the GVB-DIIS convergence scheme sometimes leads to convergence when DIIS does not.
- Set the SCF level shift in the Methods window to 0.5 or 1.0. The higher the setting, the more the virtual orbitals' energies 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.

- Change the Accuracy level setting in the Methods window to ultrafine. This setting causes the job to use denser pseudospectral grids and tighter cutoffs, and generally increases computational costs by a factor of two to three.
- If the calculation is a DFT job, use finer DFT grids. You can make this setting from the Grid density option menu in the DFT window. This setting also increases the computational cost.

For transition-metal-containing systems, particularly organometallics, you can often obtain superior results by improving the initial guess wavefunction. 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.

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 window (which is accessible by clicking on the Edit Input 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 C2 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
C2     -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 218](#).

After saving the input file with the **iguess** setting and **atomic** section, you can run it in Jaguar in the usual manner.

## 7.1.2 GVB Calculations: GVB Pair Selection

For most molecules, Lewis dot structures give a reasonable idea of what GVB pairs you should consider setting. If you want to automatically assign pairs by Lewis dot structure for input files generated and submitted outside the GUI, see [Section 9.5.5 on page 170](#). You do not have to assign all possible GVB pairs. You can set GVB pairs in any order.

If you are studying a dissociating bond, you should assign all reasonable GVB pairs for that bond. For some purposes, such as for dipole moment calculations, you may find that assigning only pairs for bonds between two different atoms is sufficient. Bonds to hydrogen atoms can also be ignored for some cases.

You should not assign GVB lone pairs if you are using a minimal basis set, since the basis set does not have enough degrees of freedom to handle the lone pair. When assigning lone pairs, you should only put one GVB lone pair on atoms from the nitrogen group, two for those from the oxygen group, three for the fluorine group, and one for the carbon group. In the last case, assigning lone pairs is only reasonable when the atom is bonded to only two neighbors. If you assign one GVB lone pair for an atom, you should also assign any other possible GVB lone pairs on that atom.

## 7.1.3 Geometry Optimization

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 7.2](#) 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 window). If it did not reach convergence, you can restart the run, as described in [Section 7.2](#).

### 7.1.4 Electrostatic Potential Charge Fitting

It is probably best not to constrain electrostatic potential charge fitting to reproduce multipole moments higher than the dipole moment, because the errors in fitting the Coulomb field outside the molecule are likely to be high. Fitting to the dipole moment is usually safe; in fact, even without this constraint, the dipole moment resulting from the fitted charges is generally similar to that calculated from the wavefunction.

## 7.2 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 complete, you can usually find the new input file by following the directions at the end of this section.

A new input file, or restart file, appears in the local job directory when any Jaguar job is completed. This 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 wavefunction, the final geometry, and the like. Thus, if you want to restart the calculation with the wavefunction 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 3.4 on page 34](#), 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 from the **Methods** window, whose button



appears in the main window. 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 GVB calculation instead of Hartree-Fock. [Section 3.4 on page 34](#) 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 Jaguar Run window.

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 would need to be set to 0.) The `restart.in` file overwrites itself each time, so that the final version is written either at the end of the run or just prior to any problems encountered.

## 7.3 Suggestions for GAUSSIAN 9x Users

We recognize that some Jaguar users also use GAUSSIAN 9x for calculations. Therefore, Jaguar can generate or read GAUSSIAN 9x input files. If you plan to perform GVB calculations with GAUSSIAN 9x, you may find this feature particularly useful, since you can use Jaguar to generate a high-quality GVB initial guess automatically.

### 7.3.1 Generating GAUSSIAN 9x Input Files With Jaguar

You can use the GUI as a convenient tool to create GAUSSIAN 9x input files. The output file that is produced from the Jaguar run and whose name ends in `.g92` can be used as a GAUSSIAN 9x input file. The `.g92` 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. Details applying only to constructing an input file for a GVB calculation are discussed below.

To create a `.g92` file, turn on the Gaussian-92 input deck (`.g92`) option in the File Output window, whose button appears in the Output section of the Jaguar panel. If you are just creating a GAUSSIAN 9x input file and you do not want to use Jaguar to generate a converged wavefunction, you can save some time by using the Edit Job window to add the keyword setting **igonly**=1 (initial guess only) to the **gen** section of the input file.

The information in the `.g92` file depends on the information you have provided. The file always contains a molecular geometry (in Cartesian coordinates and ångströms); instructions for how to input geometries are available in [Section 3.2 on page 26](#). 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 Molecular State window. You can also set the name of the basis set you want to provide in the `.g92` file (for example, STO-3G) using the Basis Set window. (The default basis set choice is 6-31G\*\*.)

To actually generate the `.g92` file, you need to run the Jaguar job you have just specified. See [Section 3.6 on page 38](#) for information on running jobs.

### 7.3.1.1 Making Input Files for GVB Calculations

To set up the `.g92` file for a GVB calculation, you should use the default setting, Compute from HF initial guess, from the GVB initial guess option menu, which is in the Methods window. You should specify the GVB pairs in the GVB window, as well. See [Section 4.3 on page 56](#) for information on setting up GVB calculations.

If you have selected a GVB calculation, symmetry is automatically turned off, and the `.g92` file also specifies **nosymm**. You might want to delete this setting from the `.g92` file after it is produced.

The `.g92` file also contains a Jaguar-generated initial guess if you have selected a GVB calculation, and notes that this trial wavefunction is to be used as an initial guess for the GAUSSIAN 9x run (“guess=cards”). If you have chosen to do an initial-guess-only calculation, as described above, the initial guess is generated from Jaguar’s GVB initial guess routine. Otherwise, the initial guess provided in the `.g92` file is the final wavefunction resulting from the Jaguar SCF calculation performed starting from the GVB initial guess.

### 7.3.1.2 Other Jaguar Options for the `.g92` File

You can use a Jaguar input file to run a Jaguar job which generates a `.g92` file. See [Chapter 9](#) for a description of input files. Selecting the Gaussian-92 input deck (`.g92`) 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 Save window to save a Jaguar input file, and edit it by hand later to set other keywords.

You can generate additional information for the `.g92` 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 `.g92` file (as described for GVB calculations above) even if you are doing a non-GVB calculation. Setting it to 5 allows you to explicitly provide the basis set itself, rather than just the basis set name, within the `.g92` file. This option is useful for specifying basis sets which are included in Jaguar but not in GAUSSIAN 9x. Setting **ip160** to 4 allows you to include both the initial guess and the basis set in the `.g92` file.

### 7.3.2 Getting Basis Sets or Orbitals for GAUSSIAN 9x

The preceding subsection describes how to generate basis sets or orbitals for a GAUSSIAN 9x input file. You can also output a basis set in the format used by GAUSSIAN 9x by turning on the Gaussian-92 basis set (`.gbs`) option in the File Output window. The output is written to a file with the extension `.gbs`.

You can write orbitals from Jaguar in the format used by GAUSSIAN 9x (for its “guess=cards” option) by choosing to print the appropriate orbitals from the Orbital Output window, which is described in [Section 6.7 on page 133](#). You must choose the f19.15 or f8.5 format from the How option menu.

### 7.3.3 Using GAUSSIAN 9x Files as Jaguar Input

GAUSSIAN 9x input files can be read in the GUI, which reads the molecular geometry from them, and also turns symmetry off for the calculation or turns on electrostatic potential fitting to atomic centers if the GAUSSIAN 9x input file requests either of those options. Any other Jaguar settings take on their default values. For information on scanning in GAUSSIAN 9x input files as Jaguar input, see [Section 3.4 on page 34](#).



---

# Chapter 8: 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](#), [Section 8.3](#), and [Section 8.4](#) describe GVB, GVB-RCI, and LMP2 calculations and how the pseudospectral method improves computational scaling and efficiency for these methods. [Section 8.5](#) contains a brief description of density functional theory. [Chapter 4](#) 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 wavefunction 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 the pseudospectral method requires to compute the information needed for 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  $\mathbf{Q}$ , which is given by the equation

$$\mathbf{Q} = \mathbf{S}[\mathbf{R}^\dagger \mathbf{w} \mathbf{R}]^{-1} \mathbf{R}^\dagger \mathbf{w} \quad (1)$$

where  $\mathbf{S}$  is the analytic overlap matrix between the fitting functions and the basis set,  $\mathbf{R}$  is the matrix of fitting functions evaluated at the grid points, and  $\mathbf{w}$  is a diagonal matrix of grid weights. The fitting functions used to construct the matrix  $\mathbf{R}$  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  $\mathbf{Q}$  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  $Q_{ig}$  elements are calculated for each basis function  $i$  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  $Q$  does not depend on the wavefunction itself, it can be fully computed before the SCF procedure. However, since the  $Q$  for each grid type contains  $N_{\text{basis}} \times N_{\text{grid}}$  elements, where  $N_{\text{basis}}$  is the number of basis functions and  $N_{\text{grid}}$  the number of grid points (which is generally larger than  $N_{\text{basis}}$ ), we sometimes reduce memory demands by only computing and storing the  $N_{\text{basis}} \times N_{\text{fit}}$  matrix  $S[\mathbf{R}^\dagger \mathbf{w} \mathbf{R}]^{-1}$  in the program `rwr`, for cases where the  $Q$  for that grid type is only needed for one SCF iteration. We then assemble the full  $Q$  during the SCF iteration for which it is needed.

After the program `rwr` has generated the  $Q$  or  $S[\mathbf{R}^\dagger \mathbf{w} \mathbf{R}]^{-1}$  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

$$A_{klg} = \int \frac{\phi_k(1)\phi_l(1)}{r_{1g}} d\mathbf{r}_{1g} \quad (2)$$

where  $\phi_k$  and  $\phi_l$  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:

$$J_{ij} = \sum_g Q_{ig} \left[ \sum_{kl} A_{klg} D_{kl} \right] R_{jg} \quad (3a)$$

$$K_{ij} = \sum_g Q_{ig} \left[ \sum_n A_{jng} \sum_m D_{nm} R_{mg} \right] \quad (3b)$$

where  $D$  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  $J$  and  $K$  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  $(a|a|a|a)$ ,  $(a|a|a|b)$ ,  $(a|a|b|b)$ ,  $(a|b|a|b)$ , or  $(a|a|b|c)$ , 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  $\mathbf{J}$  and  $\mathbf{K}$  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 Implementation of the GVB Method

The pseudospectral method has also been extended to electron correlation methods, with a particular focus on Generalized Valence Bond (GVB) [20] calculations. Highly refined GVB initial guess [14] and convergence [11] algorithms have been automated within Jaguar, allowing the scaling advantages resulting from the pseudospectral method to be maintained for GVB calculations. The method yields very accurate excitation energies, rotational barriers, and bond energies for many molecules, and GVB calculations with Jaguar are typically 10 to 100 times more efficient than the best conventional GVB programs, even for molecules as small as ten atoms [6].

In the GVB approach, each bond or other electron pair is described by two non-orthogonal orbitals, whose contributions to the bond description are obtained variationally. The bond description can thus change smoothly from a description with two atomic-like orbitals at large bond distances to a description with bond-like orbitals at short distances. This improvement over Hartree-Fock, which treats bonds as having equal amounts of covalent and ionic character, allows GVB to describe charge transfer reactions and bond breaking and formation accurately, and also gives better results for other molecular properties than an HF treatment alone can provide.

The goal of a GVB calculation, then, is to obtain pairs of GVB orbitals  $\psi_{pa}$  and  $\psi_{pb}$ , where  $p$  ranges from 1 to the number of GVB pairs  $N_{\text{gvb}}$ , that lead to a minimum energy for the molecular wavefunction

$$\Psi = \prod_{p=1}^{N_{\text{gvb}}} (\psi_{pa}\psi_{pb} + \psi_{pb}\psi_{pa})(\alpha\beta - \beta\alpha) \quad (4)$$

For a given  $p$ , the orbitals  $\psi_{pa}$  and  $\psi_{pb}$  form a pair that describes a particular bond or other pair of electrons. Under the perfect pairing restriction, the GVB orbitals within a pair are not orthogonal, although they are each orthogonal to all GVB orbitals in other pairs. For computational purposes, it is useful to form orthogonal GVB natural orbitals  $\psi_{pg}$  and  $\psi_{pu}$  from the GVB orbitals  $\psi_{pa}$  and  $\psi_{pb}$  and their overlap  $S_p$ , as follows:

$$\psi_{pg} = \frac{(\psi_{pa} + \psi_{pb})}{\sqrt{2(1 + S_p)}} \quad (5a)$$

$$\psi_{pu} = \frac{(\psi_{pa} - \psi_{pb})}{\sqrt{2(1 - S_p)}} \quad (5b)$$

The  $\psi_{pg}$  orbitals generally have bonding character, while the  $\psi_{pu}$  orbitals are anti-bonding. The contribution to the GVB wavefunction from each pair is given by

$$(C_{pg}\psi_{pg}\psi_{pg} - C_{pu}\psi_{pu}\psi_{pu})(\alpha\beta - \beta\alpha) \quad (6)$$

where the GVB configuration interaction (CI) coefficients  $C_{pg}$  and  $C_{pu}$  satisfy the following equations:

$$\frac{C_{pg}}{C_{pu}} = \frac{(1 + S_p)}{(1 - S_p)} \quad (7a)$$

$$C_{pg}^2 + C_{pu}^2 = 1 \quad (7b)$$

Solving for the optimal GVB orbitals is therefore a matter of determining both the GVB natural orbitals and the GVB CI coefficients that minimize the energy of the GVB wavefunction. This energy is given by the equation



$$E = \sum_{\mu}^{2N_{gvb}} 2C_{\mu}^2 h_{\mu\mu} + \sum_{\mu\nu}^{2N_{gvb}} (a_{\mu\nu} J_{\mu\nu} + b_{\mu\nu} K_{\mu\nu}) \quad (8)$$

where  $\mu$  and  $\nu$  range over all GVB natural orbitals (bonding and anti-bonding), and where these orbitals are expanded in terms of the basis functions, as shown here:

$$\Psi_{\mu} = \sum_i^{N_{basis}} c_{i\mu} \Phi_i \quad (9)$$

The terms  $h_{\mu\mu}$ ,  $J_{\mu\nu}$ , and  $K_{\mu\nu}$  are defined by:

$$\begin{aligned} h_{\mu\mu} &= \langle \Psi_{\mu} | \mathbf{h} | \Psi_{\mu} \rangle = \sum_{ij}^{N_{basis}} c_{i\mu} c_{j\mu} h_{ij} \\ &= \sum_{ij}^{N_{basis}} c_{i\mu} c_{j\mu} \langle i | \mathbf{h} | j \rangle \end{aligned} \quad (10a)$$

$$\begin{aligned} J_{\mu\nu} &= (\mu\mu | \nu\nu) = \langle \Psi_{\nu} | J_{\mu} | \Psi_{\nu} \rangle = \sum_{ij}^{N_{basis}} c_{i\nu} c_{j\nu} J_{ij}^{\mu} \\ &= \sum_{ij}^{N_{basis}} c_{i\nu} c_{j\nu} \sum_{kl}^{N_{basis}} c_{k\mu} c_{l\mu} (ij | kl) \end{aligned} \quad (10b)$$

$$\begin{aligned} K_{\mu\nu} &= (\mu\nu | \mu\nu) = \langle \Psi_{\nu} | K_{\mu} | \Psi_{\nu} \rangle = \sum_{ij}^{N_{basis}} c_{i\nu} c_{j\nu} K_{ij}^{\mu} \\ &= \sum_{ij}^{N_{basis}} c_{i\nu} c_{j\nu} \sum_{kl}^{N_{basis}} c_{k\mu} c_{l\mu} (ik | jl) \end{aligned} \quad (10c)$$

and the quantities  $a_{\mu\nu}$  and  $b_{\mu\nu}$  obey the following rules:

$$a_{\mu\mu} = C_{\mu}^2, \quad b_{\mu\mu} = 0; \quad (11a)$$

$$a_{\mu\nu} = 0, \quad b_{\mu\nu} = -C_{\mu} C_{\nu} \quad (11b)$$

for  $\mu$  and  $\nu$  in the same pair ( $\mu \neq \nu$ ); and

$$a_{\mu\nu} = 2C_{\mu}^2 C_{\nu}^2, \quad b_{\mu\nu} = -C_{\mu}^2 C_{\nu}^2 \quad (11c)$$

for  $\mu$  and  $\nu$  in different pairs.

Examining the variation of the energy  $E$  with respect to the basis set coefficients  $c$  gives the equations for the Fock operator corresponding to each GVB natural orbital:

$$F_{ij}^{\nu} = C_{\mu}^2 h_{ij} + \sum_{\nu}^{2N_{gvb}} (a_{\mu\nu} J_{ij}^{\nu} + b_{\mu\nu} K_{ij}^{\nu}) \quad (12)$$

Each orbital's Fock operator thus depends on the other orbitals' Coulomb and exchange operators.

At the beginning of each SCF iteration, the `scf` program is provided with a set of proposed natural orbitals and a set of CI coefficients that dictate the contribution of each natural orbital to the GVB orbitals. For that set of GVB natural orbitals, the program first solves for revised CI coefficients by evaluating the Coulomb and exchange matrix elements for those orbitals and diagonalizing the two-by-two matrices  $\mathbf{Y}^p$  in the basis of the two natural orbitals in pair  $p$ , as described by these equations:

$$\mathbf{Y}^p \mathbf{C}^p = \mathbf{C}^p \mathbf{E}^p \quad (13a)$$

$$Y_{pg, pg}^p = h_{pg, pg} + \frac{1}{2} J_{pg, pg} + \sum_{q \neq p}^{N_{gvb}} C_{qg}^2 (2J_{qg, pg} - K_{qg, pg}) + \sum_{q \neq p}^{N_{gvb}} C_{qu}^2 (2J_{qu, pg} - K_{qu, pg}) \quad (13b)$$

$$Y_{pu, pu}^p = h_{pu, pu} + \frac{1}{2} J_{pu, pu} + \sum_{q \neq p}^{N_{gvb}} C_{qg}^2 (2J_{qg, pu} - K_{qg, pu}) + \sum_{q \neq p}^{N_{gvb}} C_{qu}^2 (2J_{qu, pu} - K_{qu, pu}) \quad (13c)$$

$$Y_{pg, pu}^p = Y_{pu, pg}^p = \frac{1}{2} K_{pg, pu} \quad (13d)$$

In practice, since the CI coefficients are mutually interdependent, they are determined using a self-consistent iterative procedure.

Next, holding the CI coefficients fixed, the program evaluates the energy and the Fock matrix and adjusts the basis set coefficients describing the GVB natural orbitals accordingly, in basically the same manner used for the usual HF treatment. The revised orbitals and CI coefficients are then used in the next SCF iteration, and the process continues until both the GVB natural orbitals and the CI coefficients have converged.

The GVB treatment can also be applied to open shell cases, or restricted to certain electron pairs. These variations are described in reference 20, which also provides much more detail about the GVB methods and equations. The ability to restrict the use of GVB to particular electron pairs is an important strength of the method. This feature allows computationally inexpensive correlation of critical regions in very large molecules.

## 8.3 GVB-RCI Wavefunctions

The GVB-RCI (restricted configuration interaction) wavefunction is the simplest multiterminantal reference wavefunction which properly dissociates to open shell fragments regardless of the spin multiplicity of the fragments. A critical advantage of GVB-RCI is that the GVB and RCI computations can be confined to a localized region of the molecule. The GVB-RCI method is therefore particularly useful for evaluating bond energies and bond formation and breaking, as well as for studies of open shell radicals and other systems for which it is important to avoid spin contamination problems.

The version of GVB-RCI within Jaguar uses pseudospectral numerical methods and a novel internal contraction scheme in which a GVB-PP wavefunction is used as a correlated mean field reference state [12]. This implementation of GVB-RCI can be used to generate highly accurate GVB-RCI wavefunctions, with energies within about 0.1 kcal/mole of results from all-analytical integral calculations [12]. The internal contraction scheme used restricts the number of CI coefficients in the RCI calculation to  $\sim n^3$ , where  $n$  is the number of GVB pairs, yet is in excellent agreement with a fully uncontracted CI which by contrast would contain  $2^n n^3$  CI coefficients (the number of uncontracted determinants).

The GVB-RCI program within Jaguar generates a correlated wavefunction from intra-pair excitations of the GVB reference wavefunction described in Section 8.2, using a highly effective contraction procedure to reduce the length of the CI expansions. The program employs the pseudospectral method to speed up integral evaluation, and systematically includes the most important configurations to make the calculation more practical, with minimal loss of accuracy relative to the fully uncontracted expansion.

The spatial states for an RCI pair are constructed from the same natural orbitals as those used for the GVB reference wavefunction,  $\Psi_{pg}$  and  $\Psi_{pu}$ , but in addition to the GVB spatial state from Equation (6), rewritten here:

$$\xi_{p0} = C_{pg}\Psi_{pg}^2 - C_{pu}\Psi_{pu}^2 \quad (14a)$$

the RCI spatial states include the orthogonal complements  $\xi_{p1}$  and  $\xi_{p2}$ :

$$\xi_{p1} = \Psi_{pg}\Psi_{pu} \quad (14b)$$

$$\xi_{p2} = C_{pu}\Psi_{pg}^2 + C_{pg}\Psi_{pu}^2 \quad (14c)$$

Just as the GVB method allows the user to correlate particular electron pairs for maximal efficiency, the RCI treatment can be applied to any user-specified subset of the GVB pairs. A GVB mean field procedure is then used to evaluate a Coulomb-exchange mean field operator describing the effect of the non-excited GVB pairs on the RCI pairs. This treatment effectively reduces the two-electron part of the Hamiltonian to the space of the RCI coordinates. Even for cases with many RCI pairs, the configurations are restricted to those with only a small number of excitations and use the mean field treatment for each configuration's calculation.

The RCI spatial states  $\xi_{pl}$  add an extra complication to the necessary evaluation of Coulomb and exchange matrix elements using the natural orbitals  $\Psi_{pg}$  and  $\Psi_{pu}$ . For the GVB case, it is sufficient to compute the following matrix elements (corresponding to Equation (10b) and Equation (10c)):

$$\left. \begin{aligned} J_{\nu\nu}^{\mu\mu} &= (\mu\mu|\nu\nu) = \left( \mu(1)\mu(1) \left| \frac{1}{r_{12}} \right| \nu(2)\nu(2) \right) \\ K_{\mu\nu}^{\mu\nu} &= (\mu\nu|\mu\nu) = \left( \mu(1)\nu(1) \left| \frac{1}{r_{12}} \right| \mu(2)\nu(2) \right) \end{aligned} \right\} \begin{array}{l} \mu \in \{\Psi_{pg}, \Psi_{pu}\}, \\ \nu \in \{\Psi_{pg}, \Psi_{pu}\} \end{array}$$

where the  $\mu$  and  $\nu$  can each be any natural orbital. For the RCI pairs, on the other hand, all matrix elements of the form:

$$\left. \begin{aligned} J_{\gamma\delta}^{\alpha\beta} &= (\alpha\beta|\gamma\delta) \\ K_{\beta\delta}^{\alpha\gamma} &= (\alpha\gamma|\beta\delta) \end{aligned} \right\} (\alpha, \beta \in \{\Psi_{pg}, \Psi_{pu}\}, \gamma, \delta \in \{\Psi_{pg}, \Psi_{pu}\}),$$

are needed, where the  $\alpha$  and  $\beta$  natural orbitals are from the same RCI pair  $p$  (and may be the same natural orbital), while the  $\gamma$  and  $\delta$  natural orbitals are from the same RCI pair with index  $q$ . The complicated part of the calculation of the Coulomb and exchange operators, then, is evaluating matrix elements in atomic orbital (AO) space and using the AO-space matrix elements to produce the matrix elements in the natural orbital space, a process that normally requires a four-index transformation.

By using the pseudospectral method, however, Jaguar reduces the scaling of the evaluation of each Coulomb or exchange matrix operator in basis function space from  $N^4$  to  $N^3$ , and solves for the necessary matrix elements with a two-index transformation rather than an expensive four-index transformation. For simplicity, this process is described for the Coulomb matrix elements only; the equations for  $\mathbf{K}$  are similar. First, the usual three-center, one-electron integrals  $A_{klg}$  are evaluated in spectral space (see Equation (2)). The Coulomb matrix elements  $J_{\gamma\delta g}$  are then evaluated in *physical* space for all  $\gamma\delta$  corresponding to orbital products of each RCI pair,  $\psi_{pg}\psi_{pg}$ ,  $\psi_{pg}\psi_{pu}$ , and  $\psi_{pu}\psi_{pu}$ , using the equation

$$J_{\gamma\delta g} = \sum_{kl} c_{k\gamma} c_{l\delta} A_{klg} \quad (15)$$

These matrix elements are transformed into spectral space to form  $J_{\gamma\delta}^{ij}$ , where  $i$  and  $j$  are basis function indices, using the pseudospectral method in the usual manner described in Section 8.1 on page 147:

$$J_{\gamma\delta}^{ij} = \langle i | J_{\gamma\delta} | j \rangle = Q_{ig} J_{\gamma\delta g} R_{jg} \quad (16)$$

where  $\mathbf{Q}$  is the pseudospectral least-squares operator and  $R_{jg}$  is the value of the basis function  $j$  at grid point  $g$ . A final two-index transformation,

$$J_{\gamma\delta}^{\alpha\beta} = \sum_{ij} c_{i\alpha} c_{j\beta} J_{\gamma\delta}^{ij} \quad (17)$$

is performed to obtain the matrix elements in the natural orbital basis.

When Jaguar has obtained all Coulomb and exchange operators, it performs an iterative diagonalization of the Hamiltonian to obtain the RCI coefficients. The Davidson method is used for this step.

## 8.4 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 [43, 44] formulated a version of MP2 in which the occupied orbitals are first localized (e.g., via Boys localization [46]) 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 (as for other localized correlation methods such as GVB and GVB-RCI) 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 [45]. 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  $\sim N^2$ . In fact, for calculations involving both the 6-31G\*\* and Dunning cc-pVTZ basis sets, we find a scaling  $\sim N^{2.7}$  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 [46] orbitals, for which the term  $\sum_{ij} |\langle \phi_i | r | \phi_i \rangle - \langle \phi_j | r | \phi_j \rangle|^2$  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 [43, 44, 45] to expand the first order wavefunction 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  $q$ :

$$\Psi^{(1)} = \sum_{i \geq j} \sum_{pq} C_{ij}^{pq} \Psi_{ij}^{pq} \quad (18)$$

For local MP2, we must iteratively solve the following equation, which has been derived in detail by Pulay and Sæbo, for the coefficients  $C_{ij}^{pq}$ :

$$\begin{aligned} T_{ij}^{(2)} &= \mathbf{K}_{ij} + \mathbf{F} \mathbf{C}_{ij} \mathbf{S} + \mathbf{S} \mathbf{C}_{ij} \mathbf{F} \\ &\quad - \mathbf{S} \left( \sum_k [\mathbf{F}_{ik} \mathbf{C}_{kj} + \mathbf{F}_{kj} \mathbf{C}_{ik}] \right) \mathbf{S} = 0 \end{aligned} \quad (19)$$

Here  $\mathbf{F}$  is the Fock matrix,  $\mathbf{S}$  is the overlap matrix, and  $\mathbf{T}$  is the residual matrix defined by this equation. The exchange matrix  $\mathbf{K}_{ij}$  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  $C_{ij}'$  iteratively from the equation:

$$(C_{ij}^{pq})' = C_{ij}^{pq} + \frac{T_{ij}^{pq}}{\varepsilon_i + \varepsilon_j - \varepsilon_p^* - \varepsilon_q^*} \quad (20)$$

where  $\varepsilon_i$  and  $\varepsilon_j$  are the matrix elements  $F_{ii}$  and  $F_{jj}$  in the localized molecular orbital basis and  $\varepsilon_p$  and  $\varepsilon_q$  are the eigenvalues of the Fock matrix in the local virtual basis.

From the  $C_{ij}$  coefficients and the exchange matrices  $\mathbf{K}_{ij}$ , Jaguar computes the second order energy correction  $E^{(2)}$  from the equations:

$$E^{(2)} = \sum_{i \geq j} \langle \mathbf{K}_{ij} \tilde{\mathbf{C}}_{ji} \rangle \quad (21a)$$

$$\tilde{\mathbf{C}}_{ji} = (1 + \delta_{ij})^{-1}(4\mathbf{C}_{ij} - 2\mathbf{C}_{ji}) \quad (21b)$$

where the bracket in Equation (21a) denotes a trace and  $\delta_{ij}$  is 1 if  $i = j$  and 0 otherwise. Computing the exchange matrix elements for Equation (21a) 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 and GVB calculations in Section 8.1 on page 147 and Section 8.2 on page 149. The two-electron exchange integrals needed for Equation (21a) are evaluated over grid points  $g$  as follows:

$$K_{ij}^{pq} = \sum_g Q_{ig} A_{jqg} R_{pg} \quad (22)$$

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

$$A_{jqg} = \sum_{kl} c_{kj} c_{lq} A_{klg} \quad (23)$$

The summation is performed in two steps, first summing over  $k$  to form intermediates  $A_{jlg}$ ,

$$A_{jlg} = \sum_k c_{kj} A_{klg}, \quad (24)$$

then summing over  $l$  to yield the integrals in molecular orbital space

$$A_{jqg} = \sum_l c_{lq} A_{jlg}. \quad (25)$$

Jaguar's local MP2 module also includes analytical corrections similar to those described earlier for Hartree-Fock and GVB calculations, and a length scales algorithm, both of which are explained in reference 13.



## 8.5 Density Functional Theory

Density functional theory (DFT) is based on the Hohenberg-Kohn theorem [110], 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 [111], this density is expressed in terms of Kohn-Sham orbitals  $\{\psi_i\}$ :

$$\rho(\mathbf{r}) = 2 \sum_i^{\text{occ.}} |\psi_i(\mathbf{r})|^2 \quad (26)$$

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  $V_{ij}^{xc}$ :

$$V_{ij}^{xc} = \int d\mathbf{r} \left( \left( \frac{\partial f_{xc}[\rho, \nabla\rho]}{\partial \rho} \right) \chi_i(\mathbf{r}) \chi_j(\mathbf{r}) + 2 \frac{\partial f_{xc}[\rho, \nabla\rho]}{\partial \gamma} \nabla \cdot (\chi_i(\mathbf{r}) \chi_j(\mathbf{r})) \right) \quad (27)$$

where  $f_{xc}[\rho, \nabla\rho]$  is an exchange-correlation functional and  $\gamma$  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:

$$f_{xc}[\rho, \nabla\rho] = f_x[\rho] + f_{x, NL}[\rho, \nabla\rho] + f_c[\rho] + f_{c, NL}[\rho, \nabla\rho] \quad (28)$$

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 4.1 on page 49](#) and its references.

The electronic ground state energy  $E_0$  is given by

$$E_0 = 2 \sum_i \int d\mathbf{r} \psi_i^* \left( -\frac{1}{2} \nabla^2 \psi_i + \int d\mathbf{r}' V_{nuc}(\mathbf{r}') \rho(\mathbf{r}') + \frac{1}{2} \int d\mathbf{r}'' J(\mathbf{r}'') \rho(\mathbf{r}'') + \int d\mathbf{r} f_{xc}[\rho, \nabla \rho] \right) \psi_i \quad (29)$$

(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  $\rho$  to yield a final DFT energy.

A more detailed description of density functional theory can be found in references [112](#) and [113](#).

---

# Chapter 9: 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, to submit jobs from a non-X terminal, 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 that present keyword values and definitions, the default value is 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 v50012. 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.

The input file should have the following format, where “[ ... ]” symbols denote optional entries, and entries in italics represent a character string with no spaces:

```
[comments]
{sections describing molecule & calculation}
[BASISFILE: file-path/name.basis]
[ATOMIGFILE: file-path/name.atomig]
[DAFFILE: file-path/name.daf]
[GRIDFILE: file-path/name.grid]
[CUTOFFFILE: file-path/name.cutoff]
[LEWISFILE: file-path/name.lewis]
[GPTSFILE: file-path/name]
```

The last six lines are only rarely used. Therefore, generally, your Jaguar input files will take a form as simple as

```
{sections describing molecule & calculation}
```

where only the **zmat** section, which contains the geometry and will be described later in this chapter, is actually required.

The `.basis`, `.atomig` (initial guess information), `.daf` (dealiasing functions), `.grid`, `.cutoff`, and `.lewis` 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 listed in the input file 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 9.5.23 on page 210](#).

Comments in the input file are ignored by Jaguar. If an input file was produced using the GUI, text entered in the box marked Comment in the Run or Save window generally appears on the fourth line of the input file. If the geometry was symmetrized, as described in [Section 3.5.2 on page 37](#), a comment indicating the point group to which it was symmetrized appears.

## 9.1.1 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 at all, 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 currently allowed are shown 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.

Table 9.1. Sections for Jaguar Input Files

Section	Description
<b>zmat</b>	Contains list of atomic coordinates describing molecular geometry, in Cartesian or Z-matrix format.
<b>zvar</b>	Sets values for <b>zmat</b> section variables.
<b>coord</b>	Specify particular internal coordinates to be used for optimization.
<b>connect</b>	Specify particular internal coordinates to be used when generating coordinates for optimization.
<b>tvec</b>	Specify reaction coordinate at transition state for IRC calculations.
<b>gen</b>	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.
<b>gvb</b>	Sets GVB pairs.
<b>lmp2</b>	Sets LMP2 pairs for local local MP2 calculations, and/or delocalization of LMP2 pairs.
<b>atomic</b>	Sets atom-specific properties, including atomic masses (for isotopes), van der Waals radii for solvation calculations, and basis functions for individual atoms.
<b>hess</b>	Allows input of initial nuclear Hessian.
<b>guess</b>	Allows input of initial wavefunction.
<b>pointch</b>	Adds independent point charges.
<b>efields</b>	Adds electric field or fields.
<b>ham</b>	Allows user input of Hamiltonian.
<b>orbman</b>	Allows orbitals to be reordered or linearly combined.
<b>echo</b>	One-word section indicating that the input file should be echoed in the output file.
<b>path</b>	Specifies execution path, listing order of Jaguar programs to be run.
<b>plot</b>	Allows data to be generated for plotting of orbital, potential, or densities.
<b>nbo</b>	Requests NBO (Natural Bond Orbital) calculation.

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 168](#). At least one spacing character must precede and follow each keyword and each value. 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. Note that 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 inputting a geometry through the GUI can be found in [Section 3.2 on page 26](#) and [Section 3.4 on page 34](#). 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 atomic label should begin with the one- or two-letter elemental symbol, in either uppercase or lowercase characters. Additional 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 atomic label. 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
&
```

A Z-matrix style **zmat** section should begin with the “&zmat” (or “\$zmat”) label and end with a & or \$ character, should not include a line defining any variables (which are set in the **zvar** section described in [Section 9.3 on page 166](#)), and should not contain any comment lines, but otherwise should have the same format as described in [Section 3.2.5](#), [Section 3.2.6](#), and [Section 3.2.7](#). One additional, optional feature is also available from the input file: you can orient the molecule or system according to 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 then assumes the first atom in the Z-matrix is at the origin, the second is along the *a*-axis (in the negative direction for *-a*), and the third atom is in the *ab* plane, in the quadrant determined by the positive or negative signs of *a* and *b*.

Z-matrix input is interpreted in the units specified by **unit**. [Section 3.2 on page 26](#) also includes a description of how to specify bond length or angle constraints on the Z-matrix coordinates for geometry optimizations.

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 Z-matrix format geometries.

Finally, if you are optimizing a molecular structure to obtain a minimum-energy structure or a transition state, you might want to refine the Hessian used for the job. (See [Section 5.3 on page 88](#) for information on the methods used for transition state optimizations, including 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 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 run with this **zmat** section:

```
&zmat
O1
H2  O1  1.1*
H3  O1  1.1*  H2  108.0*
&
```

that included Hessian refinement would use both O–H bonds and the H–O–H angle in the refinement.

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.

Certain types of transition state optimizations require that you enter two or three geometries (see [Section 5.3 on page 88](#) for details). For these jobs, you can input the second and/or 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 a list of equations setting the values of any variables in the geometry input 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, just 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, if used in an optimization, freeze all ycoor values to be equal to 0.753108 during the job.

Certain types of transition state optimizations require that you enter two or three geometries (see [Section 5.3 on page 88](#) 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.

## 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.



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 9.5.9 on page 179](#) for more details.)

To specify that particular bonds or angles should be included in the internal coordinates generated and 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. If you want to hold the coordinate fixed at its initial value throughout the job, add the entry “#” to the end of the line (after one or more spacing characters).

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 also 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. As a simple example of the use of a dynamic constraint, consider the following **zmat** section for a water molecule, 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 angstroms. Then you would add the following **coord** section:

```
&coord
h1 h2 # 2.0
&
```

You can use a **connect** section to specify the bonds used by Jaguar in its generation of internal coordinates. 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 4](#) and [Chapter 6](#) for details.

Throughout 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 4](#) and [Chapter 6](#). Finally, keywords relating to grids and dealiasing functions, cutoff parameters, and memory usage are described.

### 9.5.1 Geometry Input Keywords

The keywords **iunit** and **covfac** help determine how the geometry input from the **zmat** section will be interpreted. The **iunit** keyword, whose default value is 1, describes what units the geometry is assumed to have, as indicated in [Table 9.2](#).

Table 9.2. Options for the Keyword **iunit**

Keyword	Value	Description
<b>iunit</b>	0	Geometry units are bohr and radians
	<i>1</i>	Geometry units are Angstroms and degrees
	2	Geometry units are bohr and degrees
	3	Geometry units are Angstroms and radians

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 9.44](#). The default value for this variable is 1.2.

## 9.5.2 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 3.3](#) on page 33.

Table 9.3. Keywords to Describe the Molecular State

Keyword	Value	Description
<b>molchg</b>	any	Overall charge on molecule, excluding point charges set in <b>pointch</b> section (default=0)
<b>multip</b>	>0	Spin multiplicity: 1 for singlet, 2 for doublet, etc. (default=1, except for <b>ihamtyp</b> =0, when <b>multip</b> =2 by default)

## 9.5.3 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 218). The masses used are from ref. 118.

Table 9.4. Keyword to Describe the Atomic Masses Used

Keyword	Value	Description
<b>massav</b>	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

## 9.5.4 Symmetry-Related Keywords

By default, for most calculations, Jaguar takes advantage molecular symmetry to reduce computing time, as described in [Section 3.5.2](#) on page 37. 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
<b>isymm</b>	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
<b>ipopsym</b>	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)
<b>idoabe</b>	0	Allow non-Abelian point group symmetry assignment
	1	Allow only Abelian point group symmetry assignment

### 9.5.5 GVB and Lewis Dot Structure Keywords

The **ihfgvb** keyword allows you to specify the initial guess to be used for a generalized valence bond (GVB) calculation. By default, **ihfgvb** is set to 0. The **ihfgvb** keyword is described in [Section 9.5.16 on page 198](#).

GVB pairs are set in the **gvb** section, where pairs to be used in an RCI (restricted configuration interaction) calculation are also specified, and a GVB calculation will be performed any time one or more GVB pairs are described in the input file.

You can find Lewis dot structures by setting the appropriate keywords, and you can also use one of these structures to set GVB pairs automatically. The appropriate keywords are listed in [Table 9.6](#).

The Lewis dot structure code finds several alternative Lewis dot structures for resonant molecules, assigning bonds as single, double, or triple bonds unambiguously. (For instance, it finds two structures for benzene, depending on the assignment of the pi bonds.) For these cases, you might want to run Jaguar with **lewdot=-1** and **lewstr=0**, which will cause it to print out all Lewis dot structures it finds, then exit. At that point, you can figure out which structure you want to use to set the GVB pairs, set **lewstr**, **igvball**, and **igvbsel** accordingly, and set **lewdot=1**.

If you know there is only one reasonable Lewis dot structure for the molecule, you can simply set **igvball** and **igvbsel**. At that point, **lewdot** and **lewstr** will be set to 1 by default.

Table 9.6. Keywords for Evaluation of Lewis Dot Structures and Application of That Information to GVB Pair Settings

Keyword	Value	Description
<b>lewdot</b>	0	Do not find Lewis dot structure(s) or use them to set GVB pairs
	1	Find Lewis dot structure(s) and continue on with calculation ( <b>lewdot</b> =1 by default if <b>igvball</b> > 0)
	-1	Find Lewis dot structure(s) and exit without performing SCF or other later calculations
<b>lewstr</b>	0	Print all Lewis dot structures if <b>lewdot</b> =1 or -1
	>0	Use structure number lewstr for output and/or setting GVB pairs ( <b>lewstr</b> =1 by default if <b>igvball</b> > 0)
<b>igvball</b>	0	Do not select any GVB pairs based on Lewis dot structure
	1	Select GVB pairs for any atoms according to <b>igvbsel</b> and Lewis dot structure <b>lewstr</b>
	2	Select heteroatom GVB pairs only, according to <b>igvbsel</b> and Lewis dot structure <b>lewstr</b> (heteroatom pairs are all pairs whose atoms are different elements, except for C-H pairs)
<b>igvbsel</b>	1	Select only sigma GVB pairs
	2	Select only pi and second pi GVB pairs
	3	Select only sigma, pi, and second pi GVB pairs
	4	Select only lone GVB pairs
	5	Select only lone and sigma GVB pairs
	6	Select only lone, pi, and second pi GVB pairs
	7	Select sigma, pi, second pi, and lone GVB pairs (default when <b>igvball</b> > 0)

The values for **igvbsel** are easier to remember if you associate the number 1 with sigma pairs, 2 with pi pairs, and 4 with GVB lone pairs. Then, to print out any combination of these pair types, you set **igvbsel** to equal the sum of the numbers associated with the pair types you want to print.

## 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 (**mp2**=0). For more information on the local MP2 method, see [Section 4.2 on page 54](#) and [Section 8.4 on page 156](#). LMP2 keywords are given in [Table 9.7](#).

LMP2 calculations require a basis set that allows the pseudospectral method to be used. See [Table 4.3 on page 71](#) and [Table 4.4 on page 73](#) to obtain this basis set information.

Local MP2 calculations use the LMP2 method for all atoms unless the **imp2** section described in [Section 9.7 on page 217](#) 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 9.7](#).

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 9.7](#). 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.

Table 9.7. Keyword Settings for Local MP2 Calculations

Keyword	Value	Description
<b>mp2</b>	<b>0</b>	Do not run local second-order Møller-Plesset perturbation theory (LMP2) calculation
	1	Correlate core and valence electrons
	3	Run LMP2 calculation (for valence electrons only)
<b>iheter</b>	<b>0</b>	Treat all atoms with LMP2 if LMP2 is on unless <b>imp2</b> section exists; if LMP2 is on and <b>imp2</b> section exists, set atom pairs in <b>imp2</b> 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 <b>imp2</b> section at LMP2 level, other atoms at HF level
<b>ireson</b>	<b>0</b>	Do not delocalize LMP2 pairs over other atoms
	1	Calculate Lewis dot structure of molecule (by setting <b>lewdot</b> = 1), then delocalize LMP2 pairs on any bond in an aromatic ring of <7 atoms over neighboring atoms in the aromatic ring
	2	Calculate Lewis dot structure of molecule (by setting <b>lewdot</b> = 1), then delocalize LMP2 pairs on any bond in an aromatic ring of <7 atoms over all atoms in the aromatic ring

Table 9.7. Keyword Settings for Local MP2 Calculations (Cont'd)

Keyword	Value	Description
<b>idelocv</b>	<b>0</b>	Do not delocalize any pairs listed in <b>imp2</b> section (default for all calculations except those with <b>iqst</b> >0 and/or <b>ireson</b> >0)
	1	Treat all LMP2 pairs, but delocalize any pairs in <b>imp2</b> 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 <b>imp2</b> section at the LMP2 level, and also delocalize any pairs in <b>imp2</b> section as indicated there
<b>loclmp2c</b>	<b>0</b>	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
<b>loclmp2v</b>	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

### 9.5.7 DFT Keywords

To use density functional theory (DFT), you should set the **dftname** keyword. You can also use the **idft** keyword, which was the only option in versions of Jaguar prior to 5.0. If you want to evaluate the (non-self-consistent) energy of the final, post-SCF wavefunction using a particular set of functionals, you can use the **jdft** keyword. Most DFT options described here are also available from the GUI, as described in [Section 4.1 on page 49](#). For information on setting the keywords associated with grids for DFT calculations, see [Section 9.5.23 on page 210](#).

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](#). The corresponding values of **idft** are listed along with the functional name strings. For example, **dftname=bp86** specifies the BP86 functional, and is a combination of **b** for exchange and **p86** for correlation.

Table 9.8. Standard Functional Names for the **dftname** Keyword

Name	Description
hfs	Slater local exchange functional [29]
xalpha	X $\alpha$ local exchange functional [29].
hfb	Slater local exchange functional [29], Becke 1988 non-local gradient correction to exchange [32].
hfpw	Slater local exchange functional [29], Perdew-Wang 1991 GGA-II nonlocal exchange [31].
bp86-vwn5	Exchange: Slater local functional [29], Becke 1988 non-local gradient correction [32]; correlation: Vosko-Wilk-Nusair (VWN) local functional [30], Perdew 1986 gradient correction functional [35].
pwp91	Exchange: Slater local functional [29], Perdew-Wang 1991 gradient correction functional [31]; correlation: Perdew-Wang 1991 GGA-II local and non-local functionals [31].
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 [40].
pbe	Perdew-Burke-Ernzerhof local and nonlocal exchange and correlation functional [41].
b3lyp	Exchange: exact HF, Slater local functional [29], Becke 1988 nonlocal gradient correction [32]; correlation: Vosko-Wilk-Nusair (VWN) local functional [30], Lee-Yang-Parr local and nonlocal functional [33]
b3pw91	Exchange: exact HF, Slater local functional [29], Becke 1988 non-local gradient correction [32]; correlation: Perdew-Wang 1991 local and GGA-II non-local functional [31].
b3p86	Exchange: exact HF, Slater local exchange functional [29], Becke 1988 non-local gradient correction [32]; correlation: Vosko-Wilk-Nusair (VWN) local functional [30] and Perdew 1986 nonlocal gradient correction [35]
bhandh	50% exact HF exchange, 50% Slater local exchange functional [29].
bhandhlyp	Exchange: 50% exact HF exchange, 50% Slater local exchange functional [29]; correlation: Lee-Yang-Parr local and nonlocal functional [33].
b97-1	Reparametrization of Becke's 1997 hybrid functional [36] by Hamprecht, Cohen, Tozer, and Handy [39].
b98	Becke's 1998 hybrid functional including the Laplacian of the density and kinetic energy density terms as well as gradient terms [37].
sb98	Schmider and Becke reparametrization of Becke's 1998 functional [38].



Table 9.9. Functional Name Strings for Construction of the **idftname** Keyword

Name String	idft Value	Functional Description
s	1	Slater local exchange
xa	9	X $\alpha$ local exchange
b	11	Becke 1988 nonlocal exchange, Slater local exchange
pw	41	Perdew-Wang 1991 GGA-II nonlocal exchange, Slater local exchange
vwn	100	Vosko-Wilk-Nusair local correlation
vwn5	200	Vosko-Wilk-Nusair 5 local correlation
p1	300	Perdew-Zunger 1981 local correlation
p86	1300	Perdew-Zunger 1981 local correlation, Perdew 1986 non-local gradient correction
pw91	4400	Perdew-Wang GGA-II 1991 local and nonlocal correlation
lyp	2000	Lee-Yang-Parr local and nonlocal correlation

If you choose to use the **idft** keyword, you can construct a combined functional from the available local and nonlocal exchange and correlation functionals. Positive values of **idft** describe both the exchange and correlation functionals. The value of **idft** can be broken down in the form  $\text{idft} = 10000*i + 1000*j + 100*k + 10*l + m$ , or  $\text{idft} = ijklm$ , where the values of *j*, *k*, *l*, and *m* determine the exchange and correlation functionals and *i* specifies particular coefficients for the functionals. The functionals themselves are determined as described in Table 9.10 through Table 9.13.

For instance, if **idft**=1301, the DFT calculation uses the Slater local exchange functional and the Perdew-Zunger local correlation functional with Perdew's 1986 non-local correlation functional. A typical local density approximation (LDA) calculation could use **idft**=101, while **idft**=2011 sets the popular NLDA choice called BLYP. If you specify the Lee-Yang-Parr functional, which contains local and non-local terms, you may not specify a local correlation functional (i.e., if *j*=2, *k* must be 0) unless you are using the Becke 3 parameter hybrid method, as described below.

Table 9.10. Values of *m* in **idft** (Where  $\text{idft} = ijklm$ )

<i>m</i> in <b>idft</b>	Local Exchange Functional (or Exact Exchange)
<i>m</i> =0	exact exchange (Hartree-Fock)
<i>m</i> =1	Slater
<i>m</i> =9	X $\alpha$

Table 9.11. Values of  $l$  in  $idft$  (Where  $idft = ijklm$ )

$l$ in $idft$	Non-local Exchange Functional
$l=0$	none
$l=1$	Becke 1988 nonlocal term only
$l=3$	Becke 1998 (B98) nonlocal exchange functional
$l=4$	Perdew-Wang GGA-II 1991 nonlocal exchange only
$l=6$	Schmider and Becke 1998 (SB98) nonlocal exchange functional
$l=7$	HCTH407 nonlocal exchange functional
$l=8$	B97-1 nonlocal exchange functional
$l=9$	PBE nonlocal exchange functional

Table 9.12. Values of  $k$  in  $idft$  (Where  $idft = ijklm$ )

$k$ in $idft$	Local Correlation Functional
$k=0$	none
$k=1$	Vosko-Nusair-Wilk (VWN)
$k=2$	VWN5
$k=3$	Perdew-Zunger, 1981
$k=4$	Perdew-Wang GGA-II, 1991 (local correlation only)

Table 9.13. Values of  $j$  in  $idft$  (Where  $idft = ijklm$ )

$j$ in $idft$	Non-local Correlation Functional
$j=0$	none
$j=1$	Perdew 1986 nonlocal gradient correction
$j=2$	Lee-Yang-Parr local and nonlocal correlation
$j=3$	HCTH407 nonlocal correlation functional
$j=4$	Perdew-Wang GGA-II 1991 nonlocal correlation only
$j=6$	Becke 1998 (B98) nonlocal correlation functional
$j=7$	Schmider and Becke 1998 (SB98) nonlocal correlation functional
$j=8$	B97-1 nonlocal correlation functional
$j=9$	PBE nonlocal correlation functional

Table 9.14. Values of  $i$  in  $\mathbf{idft}$  (Where  $\mathbf{idft} = \mathbf{ijklm}$ )

$i$ in $\mathbf{idft}$	Hybrid Method
$i=0$	none
$i=1$	half & half (functional coefficients are all 0.5)
$i=2$	Becke 3 parameter (parameters from ref. 27)
$i=3$	Becke 1998 (B98)
$i=4$	Schmider and Becke 1998 (SB98)
$i=5$	Becke 1997 reparametrized (B97-1)

If the value of  $i$  in  $\mathbf{idft}$  is 1 or 2, the functionals given by  $j$ ,  $k$ ,  $l$ , and  $m$  are combined using coefficients determined by the appropriate hybrid method, as indicated in Table 9.14.

For the half & half hybrids, half of the exact exchange is automatically included with half of the selected exchange functional. The coefficient of any local correlation functional or non-local exchange or correlation functional is also set to 0.5. You must specify a Slater or  $X\alpha$  local exchange functional for a half & half hybrid, and if you use the Lee-Yang-Parr functional, you may not specify a local correlation functional.

For Becke 3 parameter hybrids, you need to specify a Slater or  $X\alpha$  local exchange functional, a non-local exchange functional, a local correlation functional, and a non-local correlation functional (i.e.,  $j$ ,  $k$ ,  $l$ , and  $m$  must all be non-zero if  $i$  is 2). Even when you use the Lee-Yang-Parr functional in a Becke 3 parameter hybrid, you must list a purely local correlation functional, which will be used to adjust the local correlation contribution. For Becke 3 parameter hybrids that do not include the Lee-Yang-Parr functional, the coefficients of the exact HF exchange and of the local exchange, non-local exchange, local correlation, and non-local correlation functionals are 0.2, 0.8, 0.72, 1.0, and 0.81, respectively. If the Lee-Yang-Parr functional is used in a Becke 3 parameter hybrid, its coefficient is 0.81, and the coefficient of the local correlation functional used is 0.19.

If  $\mathbf{idft}=-1$ , the values of the keywords  $\mathbf{xhf}$ ,  $\mathbf{xexl1}$ ,  $\mathbf{xexl9}$ , and  $\mathbf{xexnl}n$  determine the contributions of the exact exchange and the exchange functionals, while the keywords  $\mathbf{xcorln}$  and  $\mathbf{xcornln}$  control the contributions of the correlation functionals, as listed in Table 9.15. For example, with the keyword settings  $\mathbf{idft}=-1$ ,  $\mathbf{xhf}=0.332$ ,  $\mathbf{xexl1}=0.575$ , and  $\mathbf{xcorl1}=0.575$ , and with all other  $\mathbf{xex}$  and  $\mathbf{xcor}$  keywords set to zero, the exchange is treated with a combination of the exact exchange and the Slater local functional, while the correlation functional is pure VWN.

Finally, if you want to evaluate the energy of the final, post-SCF wavefunction using a particular functional or combination of functionals, you should use the keyword  $\mathbf{jdf}$ . This keyword can take on the same values as  $\mathbf{idft}$ , and the meanings for each value are the same as those described above for  $\mathbf{idft}$ . If  $\mathbf{jdf}=-1$ , you can set up a customized functional using

the keywords **yhf**, **yexl1**, **yexl9**, **yexnl*n***, **ycorl*n***, and **ycornl*n***, which correspond to the keywords in Table 9.15 (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.

For DFT jobs, 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 4.1 on page 49.

Table 9.15. Functional Coefficient Keywords

Keyword	Corresponding Functional (or Exact Exchange)
<b>xhf</b>	exact exchange (Hartree-Fock)
<b>xexl1</b>	Slater local exchange functional
<b>xexl9</b>	$X\alpha$ local exchange functional
<b>xexnl1</b>	Becke 1988 nonlocal gradient correction to exchange
<b>xexnl3</b>	Becke 1998 (B98) local and nonlocal exchange functional
<b>xexnl4</b>	Perdew-Wang GGA-II, 1991 nonlocal exchange functional
<b>xexnl6</b>	Schmider and Becke 1998 (SB98) local and nonlocal exchange functional
<b>xexnl7</b>	HCTH407 local and nonlocal exchange functional
<b>xexnl8</b>	B97-1 local and nonlocal exchange functional
<b>xexnl9</b>	PBE local and nonlocal exchange functional
<b>xcorl1</b>	VWN local correlation functional
<b>xcorl2</b>	VWN5 local correlation functional
<b>xcorl3</b>	Perdew-Zunger 1981 local correlation functional
<b>xcorl4</b>	Perdew-Wang GGA-II 1991 local correlation functional
<b>xcornl1</b>	Perdew 1986 non-local gradient correction
<b>xcornl2</b>	Lee-Yang-Parr local and nonlocal correlation functional
<b>xcornl3</b>	HCTH407 local and nonlocal correlation functional
<b>xcornl4</b>	Perdew-Wang GGA-II 1991 nonlocal correlation functional
<b>xcornl6</b>	Becke 1998 (B98) local and nonlocal correlation functional
<b>xcornl7</b>	Schmider and Becke 1998 (SB98) local and nonlocal correlation functional
<b>xcornl8</b>	B97-1 local and nonlocal correlation functional
<b>xcornl9</b>	PBE local and nonlocal correlation functional

## 9.5.8 CIS Keywords

The configuration interaction singles (CIS) method can be used after a closed-shell Hartree-Fock 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 used to control the CIS calculation are listed in [Table 9.16](#). You should not normally need to set **nrestart**, because the program determines how many iterations it can do with the amount of memory available.

Table 9.16. Keywords for CIS Calculations

Keyword	Value	Description
<b>icis</b>	<i>0</i>	Do not do a CIS calculation
	1	Do a CIS calculation
<b>nroot</b>	>0	Number of roots to find. Default value is 1.
<b>dconvci</b>	<i>1.0e-2</i>	Convergence criterion for the norm of the residual CI vector (default is 1e-5 for a non-pseudospectral calculation)
<b>econvci</b>	<i>1.0e-5</i>	Convergence criterion for the change in energy (default is 1e-8 for a non-pseudospectral calculation)
<b>nrestart</b>	>0	Number of CI diagonalization iterations before restarting
	<i>0</i>	Determine number of CI diagonalization iterations before restarting automatically
<b>maxciit</b>	32	Maximum number of iterations used for the diagonalization of the CI matrix

## 9.5.9 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 5](#), which also contains more details about the methods used for optimizations.

[Table 9.17](#) contains optimization keywords that take on integer values. Most default values for these 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 9.17. Integer Keywords for Geometry and Transition State Optimizations

Keyword	Value	Description
<b>igeopt</b>	<b>0</b>	Do not optimize molecular geometry
	1	Optimize minimum-energy structure
	-1	Calculate forces but do not perform geometry optimization
	2	Optimize transition state geometry
<b>iqst</b>	<b>0</b>	Perform standard (non-QST) transition state search
	1	Use quadratic synchronous transit (QST) methods to guide transition state search. Sets <b>itrvec</b> to -5
<b>igscan</b>	<b>0</b>	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 <b>iguess</b> setting
<b>nogas</b>	<b>0</b>	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 <b>esolv0</b> value (from input file) as gas phase energy (should yield same <i>structure</i> as <b>nogas=0</b> )
	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 <i>structure</i> as <b>nogas=0</b> )
<b>intopt</b>	0	Use Cartesian coordinates for optimization
	<b>1</b>	Use internally generated internal coordinates for optimization (including any from <b>coord</b> or <b>connect</b> 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, <b>intopt</b> is reset to 1)
<b>nmdr</b>	<b>0</b>	If calculating forces, compute analytic derivatives of energy
	1	If calculating forces, compute numerical derivatives of energy (obtained from calculations on $6 N_{\text{atom}}$ perturbed geometries by moving each atom <b>pertnd</b> bohr in positive or negative x, y, or z direction)
	2	Calculate frequencies numerically

Table 9.17. Integer Keywords for Geometry and Transition State Optimizations (Cont'd)

Keyword	Value	Description
<b>needgwd</b>	<b>0</b>	Do not compute DFT grid weight derivatives
	1	Compute DFT grid weight derivatives (and second derivatives if using 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)
<b>maxitg</b>	>0	Maximum number of optimization iterations (maximum number of structures generated); default is <b>100</b>
<b>iaccg</b>	<b>2</b>	Use default convergence criteria shown in <a href="#">Table 9.19</a>
	3	Perform quicker, coarser calculation by multiplying convergence criteria shown in <a href="#">Table 9.19</a> by 5
	4	Solution-phase criteria; a factor of 3 times the criteria shown in <a href="#">Table 9.19</a>
<b>nogdiis</b>	<b>0</b>	Use GDIIS method (Geometry optimization by Direct Inversion in the Iterative Subspace) [ <a href="#">117</a> ] to get new geometry
	<b>1</b>	Don't use GDIIS method
<b>ilagr</b>	<b>0</b>	Apply constraints by zeroing gradient along frozen coordinates (pre-v2.3 method; not recommended)
	<b>1</b>	Apply constraints using Lagrange multipliers
<b>nooptr</b>	<b>0</b>	Optimize all bond lengths not specifically constrained in <b>zmat</b> section
	1	Constrain (freeze) all bond lengths for optimization
<b>noopta</b>	<b>0</b>	Optimize all bond angles not specifically constrained in <b>zmat</b> section
	1	Constrain (freeze) all bond angles for optimization
<b>nooptt</b>	<b>0</b>	Optimize all torsional angles not specifically constrained in <b>zmat</b> section
	1	Constrain (freeze) all torsional angles for optimization
<b>inhess</b>	-1	Use Fischer-Almlöf guess for Hessian
	<b>0</b>	Use Schlegel guess for Hessian (default choice only if no <b>hess</b> section exists)
	1	Use unit matrix for initial Hessian

Table 9.17. Integer Keywords for Geometry and Transition State Optimizations (Cont'd)

Keyword	Value	Description
	2	Use Cartesian input Hessian found in <b>hess</b> section ( <b>inhess</b> =2 automatically if non-empty <b>hess</b> section exists)
	4	Compute and use quantum mechanical Hessian
<b>irefhup</b>	2	Refine initial Hessian using Powell updates [119]
	3	Refine initial Hessian using mixed Murtagh-Sargent/Powell updates [120]
	4	Refine initial Hessian using Murtagh-Sargent updates [121]
<b>nhesref</b>	>0	Number of lowest-frequency Hessian eigenvectors used in Hessian refinement (default is 0)
<b>ihuptyp</b>	0	Don't update Hessian
	1	Update Hessian each iteration using BFGS (Broyden-Fletcher-Goldfarb-Shanno) method [122] (default for minimum-energy structure optimizations)
	2	Update Hessian using Powell method [119]
	3	Update Hessian using mixed Murtagh-Sargent/Powell method [120] (default for transition state optimizations)
	4	Update Hessian using Murtagh-Sargent method [121] (not recommended)
	-1	Compute quantum mechanical Hessian at each geometry; sets <b>inhess</b> =4
<b>irfo</b>	0	Before using Hessian to update geometry, modify it by sign-flipping or reverting to an older Hessian [118]
	1	Before using Hessian to update geometry, modify it by RFO (rational function optimization) level shifting [123]. 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 [123]. Default for transition state searches. Automatically set for geometry optimizations that use dynamic constraints.
<b>ifollow</b>	0	For each transition state optimization iteration, select a new eigenvector to follow
	1	For each optimization iteration, follow eigenvector that most closely correlates with one followed previously



Table 9.17. Integer Keywords for Geometry and Transition State Optimizations (Cont'd)

Keyword	Value	Description
<b>itrvec</b>	0	For transition state optimization, select lowest Hessian eigenvector as transition vector
	>0	Select eigenvector number <b>itrvec</b> as transition vector (see <a href="#">Section 5.3 on page 88</a> ). Sets <b>ifollow</b> 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
<b>itradj</b>	0	Use same trust radius throughout optimization (default for minimum-energy structure optimizations)
	1	Adjust trust radius using Culot/Fletcher heuristic [ <a href="#">122</a> , <a href="#">124</a> ] (default for transition state optimizations)
	-1	Adjust trust radius using Simons' cubic potential model [ <a href="#">125</a> ] (not recommended with Jaguar)
<b>itruc</b>	0	Apply trust radius by truncating Newton-Raphson step(s)
	1	Apply trust radius by level shifting of Hessian to reduce resultant step size

The real-valued keywords that control optimizations are listed in [Table 9.18](#) and [Table 9.19](#). Note that all keyword values must be greater than or equal to zero.

The keywords shown in [Table 9.19](#) may be used to specify the geometry convergence criteria, or these criteria may be scaled to five times their default values with the keyword setting **iaccg**=3 for a quicker, coarser calculation. The first four real-valued keywords listed in [Table 9.19](#) have units of hartrees/bohr, **gconv5** and **gconv6** have units of bohr, and **gconv7** has units of hartrees.

Note also that 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.

Table 9.18. Real-valued Optimization Keywords (Except for Convergence Criteria<sup>a</sup>)

Keyword	Description
<b>pertnd</b>	Displacement (in atomic units) used for Hessian refinement or calculations of numerical forces or frequencies; default is <b>0.05</b>
<b>qstinit</b>	Distance of LST transition state initial guess between reactant and product geometries. Default is <b>0.5</b> ; range is 0.0 to 1.0.
<b>trust</b>	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 <b>itrcut</b> ( <b>trust</b> default is <b>0.3</b> , except for solvated cases or transition state optimizations, when it is <b>0.1</b> )
<b>tradmx</b>	Maximum trust radius allowed during optimization for <b>itradj</b> >0; see <b>trust</b> information ( <b>tradmx</b> default is <b>0.3</b> , except for solvated cases, when it is <b>0.1</b> )
<b>tradmn</b>	Minimum trust radius allowed during optimization for <b>itradj</b> >0; see <b>trust</b> information ( <b>tradmn</b> default is <b>0.01</b> )
<b>tremx</b>	Trust radius reduction criterion; if relative error between actual and predicted energy changes is more than <b>tremx</b> and <b>itradj</b> >0, trust radius is reduced (default is <b>0.25</b> )
<b>trgm x</b>	Trust radius reduction criterion; for <b>itradj</b> >0 and <b>trgm x</b> >0, if absolute error in a component of predicted gradient exceeds <b>trgm x</b> Hartrees/bohr, trust radius is reduced ( <b>trgm x</b> = <b>0.0</b> by default)
<b>treok</b>	Criterion for increasing trust radius; if <b>itradj</b> =2 and relative error between actual and predicted energy changes is less than <b>treok</b> , trust radius is increased ( <b>treok</b> default is <b>0.2</b> )
<b>trescal</b>	Scale factor for trust radius adjustment (default is <b>2.0</b> ); used only when <b>itradj</b> =2

a. Convergence criteria are shown in Table 9.19. All values must be set greater than or equal to 0.

Table 9.19. Geometry Convergence Criteria Keywords

Keyword	Default value	Convergence Criterion For
<b>gconv1</b>	$4.5 \times 10^{-4}$	Maximum element of gradient
<b>gconv2</b>	$3.0 \times 10^{-4}$	rms of gradient elements
<b>gconv3</b>	$1.0 \times 10^{-2}$	Maximum Newton-Raphson step (not currently used)
<b>gconv4</b>	$1.0 \times 10^{-2}$	rms Newton-Raphson step (not currently used)
<b>gconv5</b>	$1.8 \times 10^{-3}$	Maximum element of nuclear displacement
<b>gconv6</b>	$1.2 \times 10^{-3}$	rms of nuclear displacement elements
<b>gconv7</b>	$5.0 \times 10^{-5}$	Difference between final energies from previous and current geometry optimization iterations

## 9.5.10 Intrinsic Reaction Coordinate (IRC) Keywords

IRC scans have been implemented in Jaguar using the methods described in ref. [143]. 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 9.20.

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 (a new section), it defines the forward direction. An example of such a vector is as follows:

```
&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 bond stretch between atoms C2 and H3 and  $-1.0$  times the angle bend involving atoms O1-C2-H3. The forward direction is the direction that makes this coordinate larger.

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 ordinal 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.

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.

IRC calculations can be done in either Cartesian coordinates (specified with **intopt=0** in the **gen** section), or redundant internal coordinates (**intopt=1**), which is the default.

IRC in anything but the “downhill” mode requires a Hessian, which must either be entered in the **hess** section or calculated analytically before proceeding with IRC. The latter is specified with **inhess=4** in the **gen** section. Initial guess Hessians are not useful, as they do not have any imaginary frequencies.

Table 9.20. Keywords for IRC Calculations

Keyword	Value	Description
<b>irc</b>	<b>0</b>	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
<b>ircmode</b>	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
	<b>both</b>	Find IRC points in both “forward” and “reverse” direction from the transition state
<b>ircmax</b>	<b>6</b>	Maximum number of IRC points to be found in any direction. Must be a positive integer.
<b>ircmxyc</b>	<b>30</b>	Maximum number of geometry iterations to calculate each IRC point. Must be a positive integer.
<b>ircstep</b>	<b>0.1</b>	Step size taken for each IRC point. Units are bohr amu <sup>-1/2</sup> or radians amu <sup>-1/2</sup>
<b>ip472</b>	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.

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** in the **gen** section.

The restart file for an IRC job includes the geometry of the last found IRC point. This geometry is 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 proceeded far enough to have found another IRC point, no **ircmode**=downhill setting is included.

### 9.5.11 Solvation Keywords

Most of the solvation keywords correspond to GUI options described in [Section 4.5 on page 58](#). The allowed values for the integer solvation keywords are described in [Table 9.21](#). Defaults for 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.)

Table 9.21. Integer Keywords for Solvation Calculations

Keyword	Value	Description
<b>isolv</b>	0	Do not perform a solvation calculation
	2	Perform a solvation calculation using Jaguar's Poisson-Boltzmann solver
<b>icavity</b>	0	Do not include solute cavity energy term in solvation calculation
	<i>1</i>	Include solute cavity energy term (default when the solvent is water)
	2	Force calculation of cavity energy term
<b>isurf</b>	0	Do not include first shell correction factor term in solvation energy
	1	Include first shell correction factor term in solvation energy (default for most calculations in water; turns on Lewis dot keyword <b>ivanset</b> =1 by default)
<b>ivanset</b>	0	Do not set van der Waals radii according to Lewis dot structure
	1	Set van der Waals radii according to Lewis dot structure <b>lewstr</b> (1st structure by default); see <a href="#">Section 9.5.5 on page 170</a> , and <a href="#">Section 10.6 on page 253</a>
<b>kesep</b>	0	Combine terms for all one-electron matrices
	1	Keep kinetic energy terms, nuclear attraction integrals, and point charge terms separate (Note: if <b>isolv</b> =1 or 2, <b>kesep</b> =1 by default)
<b>isolvg</b>	0	Compute gradients in solvation with method used for Jaguar version 3.5 and earlier
	<i>1</i>	Compute gradients in solvation with more robust method for version 4.0 on

The real-valued parameters for solvation calculations, which are shown in Table 9.22, help describe the solvent and the solute. Section 4.5 on page 58 contains more details on these parameters. The default values for these parameters correspond to water.

For solvated geometry optimizations, the **trust** keyword, which is described in Section 9.5.9 on page 179, has a default value of 0.1 instead of its usual default of 0.3.

Table 9.22. Real-valued Solvation Keywords

Keyword	Default Value	Description
<b>epsout</b>	80.37	Outer dielectric constant of solvent
<b>epsin</b>	1.0	Inner dielectric constant of solvent [138]
<b>radprb</b>	1.40	Radius of solvent probe molecule
<b>sconv</b>	$1.5 \times 10^{-5}$	Solvation energy convergence criterion in Hartrees
<b>esolv0</b>	any number	Gas phase energy of molecule, in Hartrees; used in some restart (new input) files for solvation jobs

### 9.5.12 Properties Keywords

Various keywords are used to request calculation of molecular properties, including multipole moments and charge fitting properties. Most of the keywords listed in Table 9.23 correspond to GUI options described in Section 4.6 on page 60. Only the values listed in the table are allowed.

Table 9.23. Integer-valued Keywords for Charge Fitting, Multipole Moment, and Polarizability &amp; Hyperpolarizability Calculations

Keyword	Value	Description
<b>icfit</b>	0	Do not do electrostatic potential fitting
	1	Fit electrostatic potential to atomic centers (default for solvation calculations)
	2	Fit electrostatic potential to atomic centers and bond midpoints
<b>incdip</b>	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

Table 9.23. Integer-valued Keywords for Charge Fitting, Multipole Moment, and Polarizability &amp; Hyperpolarizability Calculations (Cont'd)

Keyword	Value	Description
	111	Use charge, dipole moment, quadrupole moment, and octapole moment as constraints in electrostatic potential (ESP) fitting
	<i>ijk</i>	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 octapole moment if $i=1$
	-1	Do all <b>incdip</b> options sequentially
<b>ldips</b>	<b>1</b>	Do not calculate any multipole moments
	2	Calculate dipole moments
	3	Calculate dipole and quadrupole moments
	4	Calculate dipole, quadrupole, and octapole moments
	5	Calculate dipole, quadrupole, octapole, and hexadecapole moments
<b>ipolar</b>	<b>0</b>	Do not calculate polarizabilities or hyperpolarizabilities
	-2	Calculate polarizabilities $\alpha$ and first and second hyperpolarizabilities $\beta$ and $\gamma$ using CPHF method
	-1	Calculate polarizabilities $\alpha$ and hyperpolarizabilities $\beta$ using CPHF method
	1	Calculate polarizabilities using 3-point finite field method
	2	Calculate polarizabilities and hyperpolarizabilities using 3-point finite field method
	5	Calculate polarizabilities and hyperpolarizabilities using 5-point finite field method
	7	Calculate polarizabilities and hyperpolarizabilities using 7-point finite field method
<b>ldens</b>	<b>0</b>	Do not calculate electron density
	1	Calculate electron density on grid (grid choice set by grid keyword <b>geldens</b> ; ultrafine grid used by default)
	-1	Calculate electron density on grid and write chdens file in a format that can be converted to a file Anthony Nicholls' program Grasp can read, using ps2grasp.f (available from Schrödinger; run with <b>geldens</b> =-3 and <b>denspc</b> =0.3 or smaller for best results)

Table 9.23. Integer-valued Keywords for Charge Fitting, Multipole Moment, and Polarizability &amp; Hyperpolarizability Calculations (Cont'd)

Keyword	Value	Description
<b>mulken</b>	<b>0</b>	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

Analytic polarizabilities  $\alpha$  and hyperpolarizabilities  $\beta$  and  $\gamma$  are available for HF, UHF, DFT, and UDFT methods. The definition of  $\beta$  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  $\alpha$ , first hyperpolarizabilities  $\beta$ , and second hyperpolarizabilities  $\gamma$ , are

$$\alpha_{ij} = -\frac{d^2 E}{dF_i dF_j} \quad \beta_{ijk} = \frac{d^3 E}{dF_i dF_j dF_k} \quad \gamma_{ijkl} = \frac{d^4 E}{dF_i dF_j dF_k dF_l}$$

If you want to calculate polarizabilities with the old definition, you must set **iopt332**=332 in the **gen** section, and you can only calculate  $\alpha$  and  $\beta$  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.

All polarizability methods are run with symmetry off—that is, the keyword **isymm** is set to 0 automatically if **ipolar**  $\neq$  0. Similarly, for any polarizability calculation, the Methods keyword **econv**, which gives the energy convergence criterion, is set by default to  $1.0 \times 10^{-6}$  (although if the calculation first satisfies the criterion dictated by the Methods keyword **dconv**, the energy convergence criterion is ignored).

The keywords **cfiterr**, **wispc**, **denspc**, and **efield**, which are listed in Table 9.24, take on real values. 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 input an electric field for finite field polarizability and hyperpolarizability calculations, as described earlier in this subsection. Its default value shown in [Table 9.24](#) applies to all cases when **ipolar** > 1. For **ipolar**=1 (3-point, polarizability-only calculations), **efield**'s 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 161](#)). The **ip172**=2 setting instructs Jaguar to write out a file named *jobname.resp* containing the electrostatic potential data (see the text under [Table 9.32](#)).

Table 9.24. Real-valued Property Keywords

Keyword	Default Value	Description
<b>cfiterr</b>	1.0 x 10 <sup>-6</sup>	Allowed error in electrostatic potential charge fitting when fitting is constrained to reproduce multipole moments
<b>wispc</b>	0.75	Spacing in bohr of rectangular grid for ESP fitting
<b>denspc</b>	0.75	Spacing in bohr of rectangular grid for electron density calculation
<b>efield</b>	0.024	Electric field for polarizability and hyperpolarizability calculations, in au (default is 0.006 for <b>ipolar</b> =1)

### 9.5.13 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 9.25](#), correspond to GUI options described in [Section 4.7 on page 64](#). 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 **ip28**=2 for output in J/mol K and kJ/mol.

When the calculation of vibrational frequencies is requested with **ifreq**=1 and the level of theory being used is Hartree-Fock, IR intensities for the IR-active vibrational modes are automatically calculated (i.e., **irder**=1 automatically). For DFT, you must explicitly set **irder**=1, and the derivatives must be calculated numerically by setting **nmdr**=2. The calculation of IR intensities involves the calculation of the dipole moment derivatives. If

Table 9.25. Keywords for Frequency-related Properties

Keyword	Value	Description
<b>ifreq</b>	<b>0</b>	Do not calculate frequencies (second derivatives)
	1	Get frequencies from Hessian of second derivatives of energy
	-1	Calculate frequencies from most recent Hessian (from end of optimization or from initial Hessian if initial Hessian was never updated)
<b>maxitcp</b>	<b>35</b>	Maximum number of CPHF iterations
<b>rmscp</b>	<b>5e-5</b>	CPHF convergence threshold
<b>imw</b>	<b>0</b>	Print normal modes in cartesian coordinates without mass-weighting
	1	Print normal modes in mass-weighted cartesian coordinates
<b>isqm</b>	<b>0</b>	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)
<b>scalfr</b>	>0	Scale vibrational frequencies by this factor (default is 1.0), and use scaled frequencies for thermochemical calculations
<b>irder</b>	<b>0</b>	Do not compute dipole derivatives or IR intensities for vibrational frequencies
	1	Compute derivatives of dipole moment and IR intensities for vibrational frequencies (see text for details)
<b>press</b>	>0	Pressure for thermochemical calculations from frequencies, in atm; default is 1.0
<b>tmpini</b>	>0	Initial temperature for thermochemical calculations, in K; default is 298.15
<b>tmpstp</b>	>0	Temperature step size (difference between consecutive temperatures) for thermochemical calculations, in K; default is 10.0
<b>ntemp</b>	>0	Number of temperatures at which thermochemical properties are computed; default is 1

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 9.16 on page 231](#)) to tell Jaguar the appropriate sequence of executables to run in order to calculate dipole derivatives only. The **path** section to use is:

```
&path pre onee hfig probe grid rwr scf ira rwr irb &
```

You must also set **irder**=1 and **isymm**=0 and **igeopt**=1. The **igeopt** setting is necessary to force tight accuracy in the SCF, but no optimization is actually performed.

To compute partial frequencies for a fragment, you must first define the fragments in the atomic section, then make the setting **freqfrag**=*fragno* in the **gen** section of the input file for the frequency calculation. These settings are made in addition to any other frequency-related settings.

### 9.5.14 Basis Set Keywords

The character string keyword **basis** allows you to override the default basis set (6-31G\*\*). This keyword should be a string describing the standard basis and any desired polarization and diffuse functions. The string describing the standard basis should be chosen from the first column of [Table 4.3 on page 71](#) or [Table 4.4 on page 73](#). Lowercase or uppercase letters can be used. The polarization and diffuse function options are described by adding \*, \*\*, +, or ++ immediately after the basis name. The meaning of these symbols is also described in [Section 4.8 on page 70](#). Neither polarization nor diffuse functions are used if none of these options are specified. The tables in [Section 4.8](#) list the basis sets and indicate which options and atoms Jaguar currently accepts for each.

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 4.8](#). Possible settings for **numd** are shown in [Table 9.26](#).

Table 9.26. Keyword to Determine the Number of d Functions

Keyword	Value	Description
<b>numd</b>	5	Use 5 d functions, regardless of basis set
	6	Use 6 d functions, regardless of basis set

### 9.5.15 Keywords for SCF Methods

Many of the keywords that control the SCF calculation can be set from the **Methods** window as described in [Section 4.9 on page 74](#). (The other keyword settings corresponding to **Methods** window settings are described in [Section 9.5.16](#) and [Section 9.5.17](#).)

The two real-valued convergence criterion keywords are **econv**, the energy convergence criterion, which dictates the maximum difference in energy between one SCF iteration and the next for convergence to be assumed, and **dconv**, the criterion for the root mean square

change in density matrix elements. The default value of **econv** is normally  $5.0 \times 10^{-5}$  Hartrees. However, for polarizability or hyperpolarizability calculations, **econv** is  $1.0 \times 10^{-6}$  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  $5.0 \times 10^{-6}$ , 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**, which has a default value of 10, sets the maximum number of Fock matrices 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**, whose units are hartrees. For standard Hartree-Fock, DFT, LMP2, or GVB-LMP2 calculations, **stdiis** is 2.0 by default; for GVB calculations (when **iconv**, listed in Table 9.27, is 3 or 4), **stdiis** is 1.0 by default; and for cases involving transition metals or open shell calculations, or when the initial guess is obtained from the one-electron Hamiltonian (**iguess**=0; see Section 9.5.16 on page 198), its default value is 0.1. In general, after GVB-DIIS starts, any density matrix averaging requested by the keywords **iteravg** and **istavg** (explained in Table 9.27) is turned off.

The last real-valued methods keyword, **vshift**, describes the amount the virtual orbitals' 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, 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 other integer SCF keywords are described in Table 9.27. Note, however, that the keyword settings for convergence are somewhat complicated, and the defaults vary somewhat depending upon the settings of other keywords.

Table 9.27. Integer Keywords for Methods Used in the SCF Convergence Procedures

Keyword	Value	Description
<b>iuhf</b>	0	Restricted open-shell (ROHF or RODFT) calculation
	1	Unrestricted (UHF or UDFT) calculation
<b>iconv</b>	0	Convergence via Fock matrix diagonalization
	1	DIIS convergence scheme (default choice for most non-GVB calculations; see <b>iconv</b> =4)
	3	OCBSE convergence

Table 9.27. Integer Keywords for Methods Used in the SCF Convergence Procedures (Cont'd)

Keyword	Value	Description
	4	GVB-DIIS convergence (default for GVB, open shell singlet calculations, and calculations whose initial guess is obtained from $H_0$ )
<b>maxit</b>	0	Calculate energy, but do not update wavefunction (i.e., do only one iteration)
	>0	Perform a maximum of maxit SCF iterations (default value is 48)
<b>newcon</b>	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
<b>iacc</b>	1	Only ultrafine grid used; “tight” cutoffs
	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
<b>iacscf</b>	0	Make no special adjustments (variable vshift, cutoff adjustments, etc.) for convergence
	1	Start with level shift ( <b>vshift</b> ) of 5.0 and decrement over 10 iterations to 0.8; keep number of canonical orbitals fixed during optimization; run at ultrafine accuracy level and with tight cutoffs
	2	Start with level shift ( <b>vshift</b> ) 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 <b>iacscf</b> =1, except with maximal analytic corrections
<b>jksep</b>	0	$2J - K$ formed for core when convenient
	1	$J$ and $K$ for core are kept separate
<b>noatcor</b>	0	Analytic corrections calculated
	1	No analytic corrections calculated
<b>nops</b>	0	Use pseudospectral method to calculate $J$ and $K$ operators
	1	Construct $J$ and $K$ from analytic four-center two-electron integrals (no grid used)
<b>noupdat</b>	0	Fock matrix updating [126] set on or off automatically
	1	No Fock matrix updating (set <b>iacc</b> =1 if you set <b>noupdat</b> =1)

Table 9.27. Integer Keywords for Methods Used in the SCF Convergence Procedures (Cont'd)

Keyword	Value	Description
<b>iteravg</b>	<b>0</b>	Do not average density matrices and adjust orbitals accordingly (unless <b>istavg</b> keyword requests averaging)
	>0	For iterations whose number is $n \cdot \mathbf{iteravg} + 1$ , where $n$ is an integer, revise orbitals so that they correspond to average of density matrices from preceding and current iterations
<b>istavg</b>	<b>0</b>	Do not average density matrices and adjust orbitals accordingly (unless <b>iteravg</b> keyword requests averaging)
	>0	For iterations whose number is <b>istavg</b> , revise orbitals so that they correspond to average of density matrices from preceding and current iterations
<b>noauto</b>	<b>0</b>	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
<b>idenavg</b>	<b>0</b>	Converge in the usual fashion
	1	Do density averaging before DIIS starts, mixing in some of old orbitals with new orbitals (default for DFT calculations)
<b>lastwv</b>	0	Skip diagonalization of Fock matrix on last iteration
	<b>1</b>	Diagonalize Fock matrix on last iteration
<b>nosuper</b>	<b>0</b>	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
<b>itwice</b>	<b>1</b>	Do $A_{mng}$ integrals once in SCF routine
	2	Do $A_{mng}$ integrals twice in SCF routine—required for GVB, optional for HF
<b>ichange</b>	$\neq 0$	Change all cutoffs (except those related to $S$ eigenvalues, bc pairs, or ab distance cutoff for exchange) by a factor of 10 to the <b>ichange</b> power

Table 9.27. Integer Keywords for Methods Used in the SCF Convergence Procedures (Cont'd)

Keyword	Value	Description
<b>ifdtherm</b>	<b>0</b>	Do not use thermal smearing in DFT or HF calculations
	1	Use fractional occupation number (FON) method for thermal smearing [127]
	2	Use pseudo-fractional occupation number (pFON) method for thermal smearing [127]
<b>fdtemp</b>	<b>10000</b>	Initial temperature in K for thermal smearing
<b>icanorb</b>	<b>0</b>	Allow number of canonical orbitals to vary during calculation
	1	Fix number of canonical orbitals during calculation
<b>nicanorb</b>	>0	Number of canonical orbitals to keep during calculation

One of the most important keywords in controlling the SCF is **iacscf**. This keyword should be employed when the SCF fails to converge under the default conditions, 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 can be directly compared to energies obtained without using **iacscf**. Energies obtained using other values of **iacscf** are not comparable because they use different grids or cutoffs.

Another method for controlling SCF convergence is thermal smearing [127]. Thermal smearing is a method for improving 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,  $n = 1/(1 + \exp((\epsilon - \epsilon_F)/kT))$ . Two methods for determining the occupation numbers have been implemented, FON (fractional occupation number) and pFON (pseudo-FON). In the first method (FON), the Fermi energy is determined so that the resulting occupations sum to the total number of electrons. In the second method (pFON), 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 (though use of symmetry is recommended), 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.

### 9.5.16 Initial Guess Keywords

Table 9.28 lists the keywords related to the initial guess and the meaning of the integer values each keyword can take on. Most of the keyword values in Table 9.28 correspond to options described in Section 4.9 on page 74.

Table 9.28. Initial Guess Keywords

Keyword	Value	Description
<b>igonly</b>	<b>0</b>	No effect
	1	Use initial guess or input wavefunction for any post-SCF calculations, skipping SCF step
<b>iguess</b>	0	Generate initial guess by diagonalizing one-electron Hamiltonian
	1	Read initial guess from <b>guess</b> section from input file or from guess file specified in WAVEFNFILE line ( <b>iguess=1</b> automatically if input file contains non-empty <b>guess</b> section)
	<b>10</b>	Construct initial guess from orbitals which give best overlap with atomic orbitals in <code>default.atomig</code> (or other <code>.atomig</code> file listed in input file) obtained by SCF calculations on atoms (note that if <b>guess</b> section exists, this is not the default choice)
	11	Construct initial guess from orbitals whose densities, when summed, best agree with the sum of the densities of the atomic orbitals in <code>default.atomig</code> (or other <code>atomig</code> file listed in input file) obtained by SCF calculations on atoms



Table 9.28. Initial Guess Keywords (Cont'd)

Keyword	Value	Description
	25	For a system that contains transition metal atoms, construct a high-quality initial guess using ligand field theory as described in reference 19. Not available for GVB calculations.
	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 reference 19. Not available for GVB calculations.
<b>ihfgvb</b>	<b>0</b>	a) Read in GVB initial guess from <b>guess</b> section if <b>iguess</b> =1, and do not run hfig or gvbig programs, or b) Compute it from HF initial guess (whose origin is determined by <b>iguess</b> ) if <b>iguess</b> is not 1
	1	Converge HF wavefunction (where the initial guess is determined by <b>iguess</b> ) and use converged HF wavefunction as input to program gvbig to get GVB initial guess
	2	Calculate a GVB initial guess from HF initial guess, whose origin is determined by setting <b>iguess</b>
<b>ihamtyp</b>	<b>0</b>	Construct Hamiltonian using standard core, open, and GVB orbitals
	2	Make highest two orbitals in initial guess an open-shell singlet pair (ROHF only)
	3	Input Hamiltonian in <b>ham</b> section ( <b>ihamtyp</b> =3 by default if a non-empty <b>ham</b> section exists)

If you want to perform an “open-shell singlet” calculation using UDFT or UHF, you must use **iopt457**=457 to set up an initial guess, and also set **isymm**=0 and **iuhf**=1. This option replaces the alpha and beta HOMO with a mixture of the HOMO and LUMO, as follows:

$$\begin{aligned}\varphi_{\text{HOMO}}^{\alpha} &= (\varphi_{\text{HOMO}} + \varphi_{\text{LUMO}}) / \sqrt{2} \\ \varphi_{\text{HOMO}}^{\beta} &= (\varphi_{\text{HOMO}} - \varphi_{\text{LUMO}}) / \sqrt{2}\end{aligned}$$

The orbitals are taken from a closed-shell starting guess. The LUMO remains the same.

**Note:** This starting guess does not correspond exactly to the open-shell singlet state, but is a mixture of singlet and triplet states. The final wave function in a UHF calculation will not necessarily correspond to what would be obtained in a 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.

### 9.5.17 Localization Keywords

For any Jaguar job, the final wavefunction 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 172](#) for details. The keywords in [Table 9.29](#) describe the available options for final wavefunction localization. See [Section 4.9.7 on page 79](#) for a description of the localization methods and the GUI settings related to localization.

Table 9.29. Keywords Related to Localization of Orbitals

Keyword	Value	Description
<b>locpostc</b>	<b>0</b>	Do not localize core orbitals of final wavefunction
	1	Perform Boys localization on core orbitals of final wavefunction
	2	Perform Pipek-Mezey localization on core orbitals of final wavefunction, maximizing Mulliken atomic populations
	3	Perform Pipek-Mezey localization on core orbitals of final wavefunction, maximizing Mulliken basis function populations
	-1	Mix the core and valence orbitals before localization, then localize according to the <b>locpostv</b> setting
<b>locpostv</b>	<b>0</b>	Do not localize valence orbitals of final wavefunction
	1	Perform Boys localization on valence orbitals of final wavefunction
	2	Perform Pipek-Mezey localization on valence orbitals of final wavefunction, maximizing Mulliken atomic populations
	3	Perform Pipek-Mezey localization on valence orbitals of final wavefunction, maximizing Mulliken basis function populations
<b>iordboy</b>	0	Do not order orbitals at end of Boys localization
	<b>1</b>	Order orbitals by their one-electron energy at the end of Boys localization
<b>ixtrboy</b>	<b>0</b>	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

When the keyword **ixtrboy** described in [Table 9.29](#) 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 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.

### 9.5.18 File Format Conversion Keywords

You can call the program Babel [24] 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 9.30](#)

Table 9.30. 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	Batchmin command file
box	DOCK 3.5 box file
cacrt	Cacao Cartesian file
cacint	Cacao Internal file
cache	CAChe MolStruct file
c3d1	Chem3D Cartesian 1 file
c3d2	Chem3D Cartesian 2 file
cdct	ChemDraw Conn. Table file

Table 9.30. Output Format Keywords and File Types for Babel File Format Conversions (Cont'd)

---

<b>Format Keyword</b>	<b>File Type</b>
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
gcart	Gaussian Cartesian 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
mm2out	MM2 Output file
mm3	MM3 file
mmads	MMADS file

Table 9.30. Output Format Keywords and File Types for Babel File Format Conversions (Cont'd)

---

Format Keyword	File Type
mdl	MDL Molfile file
miv	MolInventor file
mopert	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
pse	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

---

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 9.30. 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 a minimum-energy structure or transition state 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 made the setting **babel=gzmat.g92** in a Jaguar input file called `h2o.in`, the resulting job would create a Gaussian input file called `h2o.gzmat.g92`.

You can also convert file formats from the command line using the `jaguar babel` and `jagconvert` utilities. See [Section 11.2.5 on page 272](#) for information on these utilities.

### 9.5.19 Standard Output Keywords

The keywords listed in [Table 9.31](#) are the standard print options. They are all set to 1 by default, and the result is that none of the information that the keywords select is printed. Many of the print options can be turned on from the GUI, as described in [Section 6.4 on page 124](#).

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 9.21 on page 187](#)), causes the virial ratio,  $-V/T$ , to be printed out at the end of each SCF calculation.

Table 9.31. Output Keywords and Their Settings

Keyword <sup>a</sup>	Value	Description
<b>ip1</b>	2	Gaussian function list for basis set
<b>ip3</b>	2	Gaussian function list for dealiasing functions
<b>ip4</b>	2	Number of dealiasing functions used
<b>ip5</b>	2	Memory, disk, and i/o information
<b>ip6</b>	2	Timing information (user CPU and user+system CPU)

Table 9.31. Output Keywords and Their Settings (Cont'd)

Keyword <sup>a</sup>	Value	Description
	-2	Timing information (user cpu and wall clock)
<b>ip7</b>	2	Grid shell locations
<b>ip8</b>	2	Gaussian function list for derivatives of basis functions
<b>ip11</b>	2	Bond lengths and angles
	3	Same as setting <b>ip11</b> =2, but includes all internuclear distances (regardless of connectivity) and torsions
	4	Same as setting <b>ip11</b> =3, but includes all possible angles (regardless of atom connectivity)
	5	Same as setting <b>ip11</b> =4, but includes all possible torsions (regardless of atom connectivity)
<b>ip12</b>	2	Connectivity table
<b>ip13</b>	2	Eigenvectors and eigenvalues of overlap matrix
<b>ip18</b>	2	Overlap matrix
<b>ip19</b>	2	One-electron Hamiltonian
<b>ip23</b>	2	Additional RWR information and DFT grid information
<b>ip24</b>	2	All keyword settings, including internal ones
<b>ip25</b>	2	Multipole moments in atomic units (and Debye)
<b>ip26</b>	2	Geometries in bohr as well as Angstroms
<b>ip70</b>	2	Extra geometry optimization details
<b>ip170</b>	2	Localized orbital locations and LMP2 pair energies for local LMP2 calculations (full local LMP2 energy correction is sum of pair energies)
<b>ip173</b>	2	Fock matrix in Boys-localized orbital space
<b>ip192</b>	2	Extra optimization-related information, such as the quadratic energy error
	3	Same as setting <b>ip192</b> =2, but includes more detailed information such as the Hessian
<b>ip193</b>	2	Numerical Hessian in freq output
<b>ip194</b>	2	Diagonal force constants in internal coordinates

Table 9.31. Output Keywords and Their Settings (Cont'd)

Keyword <sup>a</sup>	Value	Description
	3	Same as setting <b>ip194</b> =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 <b>opt194</b> keyword
	4	All diagonal and off-diagonal force constants are printed

a. When any of the keywords is set equal to 1, the corresponding output is not generated.

## 9.5.20 File Output Keywords

The file output keywords are the options that cause files other than the usual log and output files to be created. 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.

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 9.32. Many of these options can be turned on from the GUI, as described in Section 6.5 on page 129.

Additional settings are available for **ip160** and **ip165**. When **ip165** is 3, the SPARTAN 4.0 archive file is written to the local job directory as `jobname.arc`. When **ip160** equals 3, an initial guess is included in the `.g92` file generated by the run (by default, `.g92` files generated for GVB calculations include initial guesses, but those for other calculations do not). If SCF iterations are performed, the initial guess for the `.g92` file will be obtained from the resulting wavefunction; otherwise, it will be generated from the appropriate Jaguar initial guess routine. When **ip160** equals 5, the basis set is included explicitly within the `.g92` file, rather than just the basis set name. When it equals 4, the trial wavefunction and the basis set are included.



Table 9.32. Effect of Setting Output Keywords for Files to 2

Keyword <sup>a</sup>	Description of What Is Printed When <code>ipi = 2</code>
<b>ip90</b>	Molden orbitals file ( <code>.mol.f</code> file)
<b>ip160</b>	GAUSSIAN 92 input file ( <code>.g92</code> file) (see text for <b>ip160</b> =3, 4, or 5)
<b>ip163</b>	GAUSSIAN 92 basis set ( <code>.gbs</code> file)
<b>ip164</b>	MQM basis set ( <code>.bas</code> file)
<b>ip165</b>	SPARTAN 4.0 archive file; appears in temp directory as <code>spart.arc</code> (to write <code>.arc</code> file to local job directory instead, use <b>ip165</b> =3)
<b>ip168</b>	GAMESS input file ( <code>.gamess</code> file)
<b>ip172</b>	RESP (Restrained Electrostatic Potential [128]) file ( <code>.resp</code> file) (set to 3 to include grid weights)
<b>ip175</b>	XMol file ( <code>.xyz</code> file) with geometries generated during optimization
<b>ip177</b>	AIMPAC ( <code>.wfn</code> file) which works with RHF/ROHF but not UHF

- a. See text in this subsection for information on **ip151** and information on other options for **ip160**.

The format of the `.resp` file created with the **ip172** keyword is as follows. The first line contains the number atoms in the molecule and the number of grid points at which the electrostatic potential was evaluated, respectively. Then the cartesian coordinates of the atoms, in bohrs, are given. Each of the remaining lines contains the electrostatic potential (in hartrees), the coordinates of the grid point (in bohrs) at which the electrostatic potential was evaluated, and, if **ip172**=3, the grid weights.

### 9.5.21 Output Keywords for Each Iteration

The information in Table 9.33 concerns output which can be printed out every SCF iteration if the keyword is set to 2. Section 6.6 on page 131 describes how to turn on these settings from the GUI. 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 9.33. Effect of Setting Output Keywords for Each Iteration to 2

Keyword <sup>a</sup>	Description of What Is Printed When <code>ipi = 2</code>
<b>ip15</b>	DIIS coefficients
<b>ip17</b>	Energy components
<b>ip110</b>	Density matrix (if Fock matrix updating was not performed during that iteration) or density difference matrix (if Fock matrix updating was done)
<b>ip121</b>	All <b>J</b> and <b>K</b> matrices, in atomic orbital space
<b>ip122</b>	Fock matrix in atomic orbital space (HF) or molecular orbital space (GVB)
<b>ip123</b>	Fock matrix in canonical orbital space
<b>ip149</b>	GVB data: f, a, b, etc.
<b>ip188</b>	Debug printing for automatic cutoff/convergence scheme
<b>ip201</b>	Total electronic density integrated on the DFT grid

a. See text in this subsection for information on **ip152**.

## 9.5.22 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 in which the orbital output appears.

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 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 9.34](#), 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 6.7 on page 133](#).

[Table 9.35](#) 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, including GVB natural orbitals, are printed. If **n** is 5, all occupied orbitals and **ipvirt** virtual orbitals are printed (or all virtuals if **ipvirt**=-1). Setting **n** to 10 causes only the GVB non-orthogonal orbitals to be printed.

Table 9.34. Keywords to Specify When to Output Orbitals

Keyword	Prints Orbitals
<b>ip100</b>	For initial guess from before SCF (generally redundant with <b>ip105</b> )
<b>ip101</b>	In canonical orbital space (each SCF iteration)
<b>ip102</b>	At end of job
<b>ip103</b>	In atomic orbital space (each SCF iteration)
<b>ip104</b>	In atomic orbital space after SCF
<b>ip105</b>	For HF initial guess
<b>ip106</b>	For GVB initial guess
<b>ip107</b>	After Boys or Pipek localization

Table 9.35. Dependence of the Format and Type of Orbital Output on the Value of **ipx**

Value of <b>ipx</b> <sup>a</sup>	<b>2 + n</b>	<b>3 + n</b>	<b>4 + n</b>	<b>5 + n</b>	<b>6 + n</b>
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 9.34).

For example, “**ip106=10**” would mean that all orbitals were to be printed in FORTRAN f8.5 format after the GVB initial guess was created. The options **ip105**  $\geq$  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 results are suitable for input in the **guess** section or for input to GAUSSIAN 92 (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 **ipx** = **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  $p_z$ , or XX for  $d_{xx}$ ) are shown. When **ipx** = **4 + n** or **ipx** = **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 6.7 on page 133](#). 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 [...].

### 9.5.23 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 243](#) and [Section 10.4 on page 248](#), 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:

```
ndfgrdX1=nr
ndfgrdX2=na
idfgrdX=pqq
```

where “X” is **m**, **f**, **g**, **u**, **d**, or **c**, signifying “medium,” “fine,” “gradient,” “ultrafine,” “second derivatives,” and “CPHF,” and correspond to grids -10, -11, -12, -13, -8, 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 [139], the default for medium, fine, and gradient grids), 2 (Becke’s Gauss-Chebyshev distribution [140]), 3 (described in ref [141]) and 4 (the Mura-Knowles distribution [142],

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 **ndfgrdX2** 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 9.36](#) 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 9.37](#), where *name* corresponds to one of the grid types listed in [Table 9.36](#). As an example, “gmedium=2” indicates that the medium grid to be used for the calculation is the second one listed in the `.grid` file, while “geldens=-3” indicates that an electron density calculation should use a cubic grid.

*Table 9.36. Pseudospectral, Charge-Fitting, and Electron Density Grid Types*

<i>name</i> <sup>a</sup>	Description
<b>coarse</b>	Least expensive, least accurate level
<b>medium</b>	Used for most SCF iterations
<b>fine</b>	Sometimes used for a limited number of iterations
<b>ufine</b>	Ultrafine; most accurate level
<b>grad</b>	Used in gradient computation
<b>lmp2</b>	Grid used for LMP2 energy calculations
<b>lmp2der</b>	Grid used for LMP2 gradient calculations
<b>charge</b>	Grid used for charge fitting
<b>eldens</b>	Used for electron density calculations

a. These names are used in the grid-related keywords described in [Table 9.37](#).

You can read in your own set of grid points and weights by using the **gname=-6** option and the GPTSFILE line of the input file, which is described in [Section 9.1 on page 161](#).

Table 9.37. Keywords for Specification of Length Scales for Sorting of Basis Functions, Grid Usage, and Dealiasing Function Usage

Keyword	Value	Description	Default for
<b><i>lname</i></b>	1	Only one length scale used for calculation	<b>lcoarse</b>
	2	Basis functions are sorted into short- and long-range	<b>lmedium, lfine, lufine, lgrad</b>
<b><i>gname</i></b>	>0	Specifies which parameter set from <code>.grid</code> file should be used for grid (e.g., 2 for second)	<b>gcoarse</b> (1), <b>gmedium</b> (2), <b>gfine</b> (3), <b>gufine</b> (4), <b>ggrad</b> (4), <b>gmp2</b> (4), <b>gmp2der</b> (2), <b>geldens</b> (4)
	-1	Use spherical charge fitting grid generated within Jaguar for grid listed by <b><i>name</i></b>	<b>gcharge</b>
	-2	Use cubic charge fitting grid generated within Jaguar for grid listed by <b><i>name</i></b>	none
	-3	Use cubic electron density grid generated within Jaguar for grid listed by <b><i>name</i></b>	none
	-6	Use grid and weights from file specified by GPTSFIL line in input file for grid listed by <b><i>name</i></b>	none
<b><i>dname</i></b>	>0	Specifies which dealiasing function from the <code>.daf</code> file should be used	<b>dcoarse</b> (1), <b>dmedium</b> (2), <b>dfine</b> (3), <b>dufine</b> (4), <b>dgrad</b> (5)

### 9.5.24 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, as described in the troubleshooting information in [Section 12.2 on page 286](#).

Memory use keywords are listed in [Table 9.38](#), 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 variables **mxpr** or **mxrwr**.

Finally, the **iq** keywords allow you to choose when to compute the full least-squares fitting matrix  $\mathbf{Q}$  from the smaller matrix  $\mathcal{S}[\mathbf{R}^\dagger \mathbf{wR}]^{-1}$  and whether to store it on disk. Names and default values (in bold italics) for these keywords are indicated in [Table 9.39](#). If a grid is

Table 9.38. Keywords Related to Memory and Disk Use

Keyword	Default	Description
<b>mxstrip</b>	200	Information for matrix elements evaluated on basis functions stored in core in strips of <b>mxstrip</b> *N words, rather than N <sup>2</sup> words at a time (where N is the number of basis functions).
<b>mxpage</b>	1000	For pseudospectral evaluation of <b>J</b> and <b>K</b> on grid points in program scf, memory is allocated ngblok* <b>mxpage</b> words at a time as needed, where ngblok is a parameter currently set to 128.
<b>nbuck</b>	64	Gridblocks are split up into sub-gridblocks whose points are all on the same atom and in the same region of space, with at most <b>nbuck</b> points, where <b>nbuck</b> ≤ ngblok (ngblok is the maximum number of grid points per gridblock, currently set to 128).
<b>nbcmx</b>	1000000	Maximum memory (in words) used by overlap and kinetic energy integral package, excluding final matrices themselves.
<b>ndisk</b>	1500	Atomic strips of <b>J</b> and <b>K</b> are kept in core rather than on disk if (# basis functions) x (# Hamiltonians) < <b>ndisk</b> . #Hamiltonians=1 for closed shell and 2 for open-shell.
<b>mxpr</b>	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 ≤ <b>mxpr</b> .
<b>mxrwr</b>	100	Maximum number of dealiasing functions evaluated at a time in subroutine rwrcalc.
<b>zmpmem</b>	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 <b>zmpmem</b> .

used only once per calculation, as the fine, ultrafine and gradient grids generally are, setting its **iqname** value to 0 saves disk space and costs no CPU time. Setting the **iqname** 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 9.39. Keywords to Determine When to Compute the Full Least-squares Fitting Matrix  $Q$ 

Keyword	Value	Description
<b>iqcoarse</b>	0	For coarse grid, compute $Q$ on the fly in the program <code>scf</code>
	1	For coarse grid, compute $Q$ in the program <code>rwr</code> and store on disk for later use
<b>iqmedium</b>	0	For medium grid, compute $Q$ on the fly in the program <code>scf</code>
	1	For medium grid, compute $Q$ in the program <code>rwr</code> and store on disk for later use
<b>iqfine</b>	0	For fine grid, compute $Q$ on the fly in the program <code>scf</code>
	1	For fine grid, compute $Q$ in the program <code>rwr</code> and store on disk for later use
<b>iqufine</b>	0	For ultrafine grid, compute $Q$ on the fly in the program <code>scf</code>
	1	For ultrafine grid, compute $Q$ in the program <code>rwr</code> and store on disk for later use
<b>iqgrad</b>	0	For gradient grid, compute $Q$ on the fly in the program <code>scf</code>
	1	For gradient grid, compute $Q$ in the program <code>rwr</code> and store on disk for later use

### 9.5.25 Plotting Keywords

You can generate a plot file, using keywords in the **gen** section, that contains the values of the 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 molecular surfaces. The length units for the grid are set with the **iunit** keyword.

The possible values of the plotting keywords are given in Table 9.40. An alternative to using these keywords is to include a plot section in the input file. See Section 9.17 on page 234 for information on the **plot** section and its keywords. See Section 4.10 on page 79 for information on setting up plot data using the GUI.

The generation of plot data is incompatible with NBO calculations. You must run NBO calculations in a separate job.



Table 9.40. Keywords for Generating Plot Data

Keyword	Value	Meaning
<b>iplotden</b>	<b>0</b>	Do not generate electron density data
	1	Generate electron density data
<b>iplotspn</b>	<b>0</b>	Do not generate electron spin density data
	1	Generate electron spin density data
<b>iplotesp</b>	<b>0</b>	Do not generate electrostatic potential data
	1	Generate electrostatic potential data
<b>iorb1a</b>	-3	Generate electrostatic potential data
	-2	Generate electron density data
	-1	Generate data for all alpha orbitals
	<b>0</b>	Do not generate any alpha orbital data
	>0	Index of first alpha orbital for which to generate data
<b>iorb2a</b>	>0	Index of last alpha orbital for which to generate data. Ignored unless <b>iorb1a</b> is positive.
<b>iorb1b</b>	-1	Generate data for all beta orbitals
	<b>0</b>	Do not generate any beta orbital data
	>0	Index of first beta orbital for which to generate data. Ignored for restricted wave functions.
<b>iorb2b</b>	>0	Index of last beta orbital for which to generate data. Ignored unless <b>iorb1b</b> is positive.
<b>plotres</b>	2.5	Number of points per unit length. The length units are defined by the <b>iunit</b> keyword. The default given here is in points/bohr.
<b>xmaxadj</b>	<b>0.0</b>	Amount to adjust the box boundary on the +x-axis. Can be positive or negative.
<b>xminadj</b>	<b>0.0</b>	Amount to adjust the box boundary on the -x-axis. Can be positive or negative.
<b>xadj</b>	<b>0.0</b>	Amount to adjust the x dimension of the box. Half the adjustment is added to each boundary. Can be positive or negative.
<b>ymaxadj</b>	<b>0.0</b>	Amount to adjust the box boundary on the +y-axis. Can be positive or negative.
<b>yminadj</b>	<b>0.0</b>	Amount to adjust the box boundary on the -y-axis. Can be positive or negative.

Table 9.40. Keywords for Generating Plot Data (Cont'd)

Keyword	Value	Meaning
<b>yadj</b>	<b>0.0</b>	Amount to adjust the y dimension of the box. Half the adjustment is added to each boundary. Can be positive or negative.
<b>zmaxadj</b>	<b>0.0</b>	Amount to adjust the box boundary on the +z-axis. Can be positive or negative.
<b>zminadj</b>	<b>0.0</b>	Amount to adjust the box boundary on the -z-axis. Can be positive or negative.
<b>zadj</b>	<b>0.0</b>	Amount to adjust the z dimension of the box. Half the adjustment is added to each boundary. Can be positive or negative.
<b>plotfmt</b>	<b>[.]vis</b>	Set the format and file extension for plot files to <code>.vis</code> (the default Maestro format).
	<b>[.]plt</b>	Set the format and file extension for plot files to <code>.plt</code> .

## 9.6 The `gvb` Section

The `gvb` section, whose GUI equivalent is described in [Section 4.3 on page 56](#), is not keyword based. The section should contain the pair settings, in any order, unless you are using the Lewis dot structure keywords described in [Section 9.5.5 on page 170](#). Each line describing a bond pair should contain three integers, which specify the type of bond (1 for sigma, 2 for pi, 3 for a second pi in a triple bond) and the atom number labels of the two atoms in the GVB pair. Each line describing a lone pair should contain a number identifying the lone pair, followed by the number or atom label of the atom associated with the lone pair, and the same atom number or label repeated once more. Either all or none of the lone pairs on an atom should be specified as GVB lone pairs, and these GVB lone pairs should be identified by consecutive numbers starting with 101. Thus, if the molecule had one lone pair on atom 2 and two on atom 5, the lines describing them would contain the numbers “101 2 2,” “101 5 5,” and “102 5 5,” respectively.

Three more entries may be added onto the ends of all of the lines specifying the pairs; these entries are present in new input files generated during or after calculations. The first value, if it is present, is either 0 or 1, where a 0 entry is a place holder, and a 1 entry indicates that a restricted configuration interaction (RCI) calculation including that pair will be performed. (By default, the pair will not be included in an RCI calculation.) The next two values, if they exist, indicate the CI coefficients for the first and second GVB natural orbitals in each pair. The first coefficient should always be positive, and its magnitude should always be greater than that of the second coefficient, which should always be negative. These coefficients are included in new input files so that if you restart the calculation with the new input file, the contributions of each GVB natural orbital will be known.

The sample **gvb** section which follows sets a sigma bond pair with RCI on between atom 1 and atom 2 and two lone pairs on atom 1.

```
&gvb
1 1 2 1
101 1 1
102 1 1
&
```

## 9.7 The **Imp2** Section

The **Imp2** section, whose GUI equivalent is described in [Section 4.2 on page 54](#), is not keyword based. The 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. Pairs can be listed in any order.

The sample **Imp2** section which follows instructs Jaguar to treat the atoms listed sixth, ninth, and tenth in the **zmat** section at the LMP2 level and all other atoms at the Hartree-Fock level. Atom 9 is bonded to atoms 6 and 10.

```
&lmp2
6 9
9 10
&
```

You can also use the **Imp2** section of the Jaguar input file to list particular LMP2 pairs and request that they be delocalized over listed atoms. With LMP2 delocalization, the space of correlating virtual orbitals for an LMP2 occupied orbital is extended to include orbitals on nearby atoms.

To delocalize a bond pair on two particular atoms over a space including orbitals on a set of other atoms, add a line to the **Imp2** section listing the atom labels or numbers of the two atoms upon which the bond pair is located by default, followed by the atom numbers or labels of the atoms over which the pair is to be delocalized. Next, set the keyword **idelocv** in the **gen** section to 1 (to treat all LMP2 pairs in the system) or 2 (to perform a “local local” MP2 calculation with only the pairs listed in the **Imp2** section treated at the LMP2 level). For example, the following **gen** and **Imp2** sections request a local local MP2 calculation with the C2–C3 bond pair delocalized over C1 and C4 as well as over C2 and C3:

```
&gen
mp2=3
idelocv=2
&
&lmp2
C2 C3 C1 C4
&
```

For QST-guided transition state searches with LMP2 wavefunctions, 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**.

## 9.8 The atomic Section

The **atomic** section allows you to specify data for different atoms in a molecule. This data can include basis sets for each individual atom, or atomic masses, a feature that allows 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 can then be used to generate high-quality initial guesses for these systems. See [Section 7.1.1 on page 139](#) for more information on using **atomic** sections in this manner.

### 9.8.1 General Format of the atomic Section

After the “&atomic” (or “\$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 also the column heading row. Subsequent rows list the corresponding 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 a solvation calculation), and the lines for the atoms H1 and H2 specify that these atoms have a nuclear mass of 2.00 amu (deuterium) and van der Waals radii of 1.2 Å for solvation purposes, 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 which are to be treated in the usual, 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 corresponding atom identifiers can be either atom labels (such as “H1” or “O” in the example above) or atom 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 may use a “?” or “-” to indicate that the default value should be used. Alternatively, you may leave the values blank for values at the end of the row. For instance, either the section

```
&atomic
atom mass vdw2
H1 2.00 1.20
H2 2.00 1.20
O ? 1.55
&
```

or the section

```
&atomic
atom vdw2 mass
H1 1.20 2.00
H2 1.20 2.00
O 1.55
&
```

has the same result as the first **atomic** section example listed above.

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. For example, 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 job’s 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 a job’s 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 9.41](#). Values for these keywords can appear in restart files.

Table 9.41. Keywords for Physical Properties in the *atomic* Section

Keyword	Description
<b>isotope</b>	Isotopic number (integer, e.g., 2 for deuterium); overridden by atom's <b>mass</b> setting if it exists
<b>mass</b>	Nuclear mass in amu
<b>esp</b>	Electrostatic potential fitted point charge (or request to fit charge to dummy atom; see text)
<b>formal</b>	Formal charge (integer value) on atom
<b>multip</b>	Spin multiplicity of atom (or fragment containing atom)
<b>2spin</b>	Number of unpaired alpha or beta electrons on atom; positive value for alpha spin, negative value for beta spin.
<b>mulk</b>	Mulliken population
<b>vdw</b>	van der Waals radii (in Å) for charge fitting
<b>vdw2</b>	van der Waals radii (in Å) for solvation
<b>cov</b>	Covalent radius in Å (used to determine bonding and other properties)

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 7.1.1 on page 139](#) for more information on using this improved initial guess method.

The **esp** keyword can be used to tell Jaguar 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 will be 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 will not be included in charge fitting. If the **esp** column entry for a dummy atom is “y” or “yes,” it will be 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 atom(s) 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 [57] van der Waals radii for hydrogen and for carbon through argon, and universal force field [54] van der Waals radii for all other elements. These radii are listed in Table 9.42, and can be changed using the **vdw** keyword.

The van der Waals radii for solvation calculations are listed in Table 9.43, 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 253 for more information on how Jaguar uses these radii in solvation calculations.

The covalent radii used to determine which atoms are bonded are given in Table 9.44. 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 keyword with a default value of 1.2. These radii can be changed using the **cov** keyword. See page 125 and Section 9.5.1 on page 168 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 4.8. 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 4.8.

*If you use an atomic section to specify different basis sets for one or more atoms than the basis set used for the other atoms in the input, you should not change any basis set assignments if you later restart that job.* For instance, if you run a job whose input file, `mixmol.in`, contains an **atomic** section that specifies different basis sets for different atoms, you can generate a new input file (restart file) called `mixmol.01.in` from the job, but if you use this input file for a second job (restarting the old calculation), you may not change the **atomic** section at all; otherwise, the program misinterprets the initial guess specified in the **guess** section in `mixmol.01.in`. Alternatively, you can delete the **guess** section completely and then change the atomic section.

Three other keywords shown in Table 9.45 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 9.45). 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. Keywords for Listing Basis, Grid, Dealiasing Function, and Charge Information for Individual Atoms in an **atomic** Section

Keyword	Value	Description
<b>basis</b>	n, no, or none	Use no basis functions on atom
	<i>basis-name</i>	Use basis functions from specified basis set on atom
<b>grid</b>	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
<b>daf</b>	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
<b>charge</b>	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, make the keyword setting **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 set **iopt420**=420 in the **gen** section, then manually assign ALL atoms to fragments using the **frag** column of the **atomic** section. The bonded metals must be assigned to separate fragments. All atoms must be assigned because all unassigned atoms will be assumed to be in the same fragment. 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  $H$ ,  $H_{ij} = H_{ji}$ , only the elements with  $j \leq i$  are read in, and the program symmetrizes the matrix itself later.

Since the Hessian has dimensions of  $3N \times 3N$ , 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 92 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 of these rows containing five or fewer matrix elements starts with an integer which is read first; this integer is not used in the program, but can be used to label the matrix elements for convenience in looking over the file. When the relevant matrix elements from that entire five-column block have been read in, the next block is read in in the same way, until all of the matrix elements for the bottom triangular half 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 91
21 22 32 42 52 62 72 82 92
31 32 33 43 53 63 73 83 93
41 42 43 44 54 64 74 84 94
51 52 53 54 55 65 75 85 95
61 62 63 64 65 66 76 86 96
71 72 73 74 75 76 77 87 97
81 82 83 84 85 86 87 88 98
91 92 93 94 95 96 97 98 99

```

you would need to enter the elements from the bottom triangle of the Hessian (shown in bold) 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
i 91 92 93 94 95
j
i 66
i 76 77
i 86 87 88
i 96 97 98 99
&

```

where  $i$  and  $j$  indicate integer labels not actually used by the program. In fact, the lines containing  $j$  can contain more than one integer, as described above.

## 9.10 The guess Section

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. From the Orbitals window in the GUI, you could select occupied orbitals or all orbitals from the What option menu and all elements as f19.15, in list or all elements as f8.5, in list from the How option menu for the original run, as described in [Section 6.7 on page 133](#), 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 9.5.22 on page 208](#), 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<sub>x</sub>, 2p<sub>y</sub>, and 2p<sub>z</sub> 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
&
```

## 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 bohr if the **iunit** keyword in the **gen** section is set to 0 or 2; see [Section 9.5.1 on page 168](#) 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 wavefunctions 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  $\mathbf{E} \cdot \mathbf{r}$  to the no-field Fock matrix, where  $\mathbf{E}$  is the electric field and  $\mathbf{r}$  is the electron position. The contribution due to the interaction between the field and each nucleus of position  $\mathbf{r}_i$  and charge  $q_i$  is  $-q_i(\mathbf{E} \cdot \mathbf{r}_i)$ .

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 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 [20]. 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.

## 9.14 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, alpha   k, l, beta   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.,  $\chi_i, \chi_j, \chi_k$ , and  $\chi_l$ ), and  $\alpha$  and  $\beta$  are angles (in degrees) indicating the degree of mixing. The command `hfiglcmo` mixes the orbitals to form orbitals  $\chi_i^{new}$ ,  $\chi_j^{new}$ ,  $\chi_k^{new}$ , and  $\chi_l^{new}$  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  $90^\circ$  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,  $\alpha$    k, l,  $\beta$    end
hfiglcmob   p, q,  $\gamma$    r, s,  $\delta$    end
&
```

## 9.15 The echo Section

The **echo** section, when it is included in input files, does not contain anything but its own label:

```
&echo &
```

Its purpose is to signal Jaguar to include a copy (echo) of the input file in the output file. If your input file does not contain an **echo** section, the output file will not contain an echo of the input file.

## 9.16 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 9.46 gives a brief description of each Jaguar program.

Table 9.46. 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	Insures orthogonalization
grid	Generates grids
rwr	Generates $Q$ operators
gvbig	Calculates GVB initial guess
scf	Performs self-consistent field calculation
rci	Performs RCI calculation
ch	Evaluates electrostatic properties (multipole moments, electrostatic potential fitting, Mulliken populations)
lmp2dip	Calculates dipole moments for LMP2 wavefunctions
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 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 wavefunctions
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

Table 9.46. Individual Programs Included in Jaguar (Cont'd)

Program	Description
geopt	Performs geometry optimization
pbf	Solves Poisson-Boltzmann equations for solvation calculation
solv	Performs solvation calculation (using results from Jaguar Poisson-Boltzmann solver)
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
machid	Utility program; returns machine information (note: not used in Jaguar calculations)

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 will only exit when convergence is reached, the program before a “goto” or “jump” label must have such a criterion. The three programs which 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 input the **path** section in such a way that each line is assumed to contain a single command. To input the path this way, you must include the word “line” after the “&path” (or “\$path”) label at the beginning of the **path** section.

## 9.17 The plot Section

The **plot** section allows you to generate data for one or more orbitals, the potential, or the density in a form that allows you to plot one of these properties on a cubic grid. In order to request this data, you must include a **plot** section with settings to describe what to plot, the dimensions of the box in which it is to be plotted, and the number of points in each direction in the box. Since the **plot** section requests a plot of information from the initial guess, unless a **guess** section is included in the input file you should use a restart file resulting from a completed job if you want to plot final properties from a job. See [Section 7.2 on page 142](#) for information on restart files. You can also generate plot data using **gen** section keywords. See [Section 9.5.25 on page 214](#) for information on this option.

The **plot** section should contain settings for **iorb1a**, **npts**, **origin**, **extentx**, **extenty**, and **extentz**, and may also contain a setting for **iorb2a**, **iorb1b**, **iorb2b**, and **ipltunit**. The keywords **iorb1a**, **iorb2a**, **iorb1b**, and **iorb2b** control the data that is generated; the keywords **npts**, **origin**, **extentx**, **extenty**, and **extentz** define the grid; and the keyword **ipltunit** sets the units of length. The possible values of the keywords are given in [Table 9.47](#).

Table 9.47. Plot Section Keywords

Keyword	Value	Meaning
<b>iorb1a</b>	-3	Generate electrostatic potential data
	-2	Generate density data
	-1	Generate data for all alpha orbitals
	0	Do not generate any plot data
	>0	Index of first alpha orbital for data generation
<b>iorb2a</b>	>0	Index of last alpha orbital for data generation. Ignored unless <b>iorb1a</b> is positive.

Table 9.47. Plot Section Keywords (Cont'd)

Keyword	Value	Meaning
<b>iorb1b</b>	-1	Generate data for all beta orbitals
	0	Do not generate any plot data for beta orbitals
	>0	Index of first beta orbital for data generation. Ignored for restricted wave functions.
<b>iorb2b</b>	>0	Index of last beta orbital for data generation. Ignored unless <b>iorb1b</b> is positive.
<b>npts</b>	$nx,ny,nz$	Number of points in each Cartesian direction. The three values can be separated by commas or spaces
<b>origin</b>	$Ox,Oy,Oz$	Coordinates of box origin. The three values can be separated by commas or spaces
<b>extentx</b>	$x,y,z$	Coordinates relative to the origin of the maximum extent of the box in the x direction. Only $x$ should be nonzero.
<b>extenty</b>	$x,y,z$	Coordinates relative to the origin of the maximum extent of the box in the y direction. Only $y$ should be nonzero.
<b>extentz</b>	$x,y,z$	Coordinates relative to the origin of the maximum extent of the box in the z direction. Only $z$ should be nonzero.
<b>ipltunit</b>	0	Use atomic units (bohr)
	1	Use angstrom units

The settings for **origin**, **extentx**, **extenty**, and **extentz** describe the box containing the grid points. The edges of the box start at the origin and form vectors in the three directions determined by the coordinate values of **extentx**, **extenty**, and **extentz**, whose three coordinates should be separated by commas or spaces. The number of points in each direction in the box is then given by **npts**, another three-dimensional setting. The default units are atomic units (bohr), but you can change the default units to angstroms by setting **ipltunit=1**.

Here is a sample **plot** section that generates plot information for orbitals 2 through 5:

```
&plot
iorb1a=2
iorb2a=5
npts=22,22,22
origin=-3.150000,-3.150000,-3.150000
extentx=6.300000,0.000000,0.000000
extenty=0.000000,6.300000,0.000000
extentz=0.000000,0.000000,6.300000
&
```

When the job is run, each type of output requested by the **plot** section shows up in a file whose name depends on *jobname*, the name for the job (for example, “h2o” for a job run from the input file `h2o.in`), and the type of information being plotted. The file name is *jobname\_density.plt* for a density plot or *jobname\_potential.plt* for a potential plot. Orbital plot information is written to separate files for each orbital, whose names depend on the four-digit orbital number *orbnum*, which is 0005 for the fifth orbital, for instance. The orbital file names are of the form *jobname\_aorbnumMO.plt*; for instance, the tenth orbital from the job `h2o` would be written to the file `h2o_a0010MO.plt`.

A `.plt` file always begins with an echo of the **plot** section used to generate it. The rest of the lines in the `.plt` file contain values of the relevant property to be plotted on the grid described by the **plot** section. The first line gives the value at the origin; next, the values along the vector described by **extentz** are given. The next values correspond to the grid points given by **extentz** but also displaced from the origin along **extenty**. The loop over the points along **extenty** continues, and the outer loop, generating points for displacements along **extentx** for the above square grids, provides the rest of the points in the file.

The generation of plot data is incompatible with NBO calculations. You must run NBO calculations in a separate job.

## 9.18 NBO Sections

To request a Natural Bond Orbital (NBO) analysis at the end of the Jaguar job, include an **nbo** section in your input file. If the section is empty, as it is here:

```
&nbo &
```

a default NBO analysis is performed. Other options for NBO calculations can also be specified in the **nbo** section or in the **core**, **choose**, and **nrtstr** sections of the Jaguar input file. See the NBO documentation for more details on NBO input and output. Jaguar’s interface to NBO 5.0 does not support the `$DEL` keylist, which means that Natural Energy Decomposition Analysis (NEDA) is not supported. The `$DELH` keylist is also not supported.

The generation of plot data is incompatible with NBO calculations.

---

# Chapter 10: 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 `default.lewis`. 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 create a new data directory and put files in it whose names and formats match those described above. When you run a job, you can edit the input file and add `BASISFILE`, `ATOMIGFILE`, `DAFFILE`, `GRIDFILE`, `CUTOFFFILE`, or `LEWISFILE` lines with the paths and names of the files you want to use. See [Section 9.1 on page 161](#) for more details. If you specify a `.cutoff` file called `accurate.cutoff`, `quick.cutoff`, or `solvent.cutoff`, the program assumes you are using an outdated file and will reset the name to `default.cutoff`, so be careful about using these names for new files.

The rest of 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.

### 10.1.1 Format of the Basis Set File

The basis sets are described in turn. Basis sets at the top of the file do not contain effective core potentials, and will be described first here. The basis sets with effective core potentials, whose names begin with “LA,” will be described later.

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 4.3 on page 71](#) or [Table 4.4 on page 73](#), 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 4.8 on page 70](#).

“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’s 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 being described (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 a 1, it indicates a polarization function which is included in the basis set if the basis set name ends in a \*, as described in [Section 4.8 on page 70](#). If the number is a 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 [reference 13](#). These range values are only used in pseudospectral calculations, so if your basis set



will be used for non-pseudospectral calculations, use a 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
****

```

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



```
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
****
```

## 10.1.2 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 4.9.6 on page 78](#) or the information on the input file `gen` section keyword `nops` in [Section 9.5.15 on page 193](#)).

To use the file in a Jaguar calculation, you must add a line in the form

```
BASISFILE: <basis file path and name>
```

to the input file for the job. 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`.

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

<http://www.emsl.pnl.gov:2080/forms/basisform.html>

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 input-filename 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 238](#). 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, because they do not have 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 wavefunctions in this file. When the basis set to be used for the calculation is 6-31G, MSV, LAV2P, LAV2D, LAV3P, LAV3D, LACVP, or LACVD (or any variant of these sets involving polarization or diffusion functions (e.g., 6-31G\*)), the initial guess is formed from the wavefunctions obtained from individual calculations on the atoms in the molecule which were calculated using that same 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 wavefunctions used to construct the initial guess are obtained by projecting either the appropriate atomic wavefunction in `default.atomig` onto the basis set actually being used for the molecular calculation. The 6-31G wavefunction is used whenever possible; when a 6-31G atomic wavefunction is not listed for a particular atom, the MSV one is used for that atom. For atoms beyond Xe in calculations using the LAV1S basis set, the LAV2P atomic results are used.

As in the `default.basis` file, the basis sets listed in the `default.atomig` file are listed in turn, and for each basis, the information for each atom is listed. Each basis set section begins with a blank line, which is followed by one or more lines reading "BASIS,"

followed by one space, and ending with the name or names (separated by a space and/or comma) of all basis sets for which the atomic calculations listed immediately after that line apply. The basis set names are listed in [Table 4.3 on page 71](#) and [Table 4.4 on page 73](#).

Next, the information for each atom follows. The first line lists the atomic symbol for the atom, followed by information which is simply a comment and is not read in. The next line lists two numbers. The first of these numbers gives the number of basis functions for that atom and basis set, as listed in the `default.basis` file, and the second gives the number of electrons for that atom included in an effective core (0 for the basis sets whose names do not start with “LA”). The line after that lists the orbital number (1 if it is the first orbital listed for that atom, 2 if it is the second, and so on), the orbital occupation (i.e., the number of electron pairs in that orbital), and the orbital energy in Hartrees. That orbital’s coefficients for each basis function for the given atom and basis set(s) follow on the next line(s).

When all of the orbitals for that atom have been specified, a line with 4 \* characters indicates the end of the information for that atom, and the data for the other atoms is listed. Similar information for each other basis set follows.

If you want to set up your own `.atomig` file, you can do so, if you use the format described above. To use the file in a Jaguar calculation, you must add a line saying

```
ATOMIGFILE: filename
```

to the input file for the job. 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:filename`, or `user@host:filename`.

## 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.<sup>1</sup> 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 simple polynomials multiplied by Gaussian functions, and are s-type, p-type, and so on, depending on the polynomial. Uncontracted dealiasing functions are simply formed by specifying the exponent of the 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<sub>x</sub>, etc., depending on the molecule and the basis set). Thus, a dealiasing uncontracted function can be specified by dictating the type (s, p, d, etc.) and the exponent desired for the Gaussian, while a contracted Gaussian function can be specified by dictating the type and referencing which set of contraction coefficients and exponents are desired.

[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](#) on [page 247](#).

### 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 next line is made up of two integers. The first integer dictates 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

---

1. To see this connectivity information for a system, set **ip12** = 2 in the **gen** section.

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 4.3 on page 71](#) and [Table 4.4 on page 73](#)), 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  $N_1 e^{-.040634r^2}$ , while the p-type uncontracted basis function for the same exponent would be  $N_2 r e^{-.040634r^2}$ .  $N_1$  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. Lead zeros 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 5.5 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 321](#).

*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 [ <a href="#">129</a> ]
2	8	3	Un 3-2 (Stroud), p.294 [ <a href="#">129</a> ]
3	12	3	U3 3-1 (McLaren), p.296 [ <a href="#">129</a> ]
4	14	5	Un 5-2 (Albrecht & Collatz), p.294 [ <a href="#">129</a> ]
5	18	5	Un 5-1 (Albrecht & Collatz), p.294 [ <a href="#">129</a> ]

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
6	18	5	Un 5-1 (Albrecht & Collatz), p.294 [129]
7	24	5	Un 5-4 (Stroud), p.295 [129]
8	26	7	Un 7-1 (Albrecht & Collatz), p.295 [129]
9	38	9	9.1 (Lebedev) [130]
10	38	9	9.1 (Lebedev) [130]
11	42	9	9.2 (Lebedev) [130]
12	44	9	9.3 (Lebedev) [130]
13	44	9	9.4 (Lebedev) [130]
14	50	11	U3 11-1 (McLaren), p.301 [129]; 11.1 (Lebedev) [130]
15	54	11	11.2 (Lebedev) [130]
16	56	11	U3 11-2 (Stroud), p.301 [129]
17	60	11	11.3 (Lebedev) [130]
18	60	11	11.3 (Lebedev) [130]
19	78	13	13.2 (Lebedev) [130]
20	78	13	13.3 (Lebedev) [130]
21	86	15	15.1 (Lebedev) [130]
22	90	15	15.2 (Lebedev) [130]
23	90	15	15.2 (Lebedev) [130]
24	110	17	17.1 (Lebedev) [130]
25	116	17	17.2 (Lebedev) [130]
26	146	19	19 (Lebedev) [131]
27	146	19	19 (Lebedev) [131]
28	194	23	23 (Lebedev) [131]
29	302	29	29 (Lebedev) [132]
30	434	35	Lebedev [133]

## 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 161](#). If the input file has no such line, 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. Lead zeros 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 4.9.5 on page 78](#), or to the keyword setting `iacc = 1` in the `gen` section of the input file, as described in [Section 9.5.15 on page 193](#). 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 wavefunction 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 9.5.23 on page 210 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.

## 10.6 The Lewis File

The Lewis file determines how van der Waals radii for calculations using the Jaguar solvation module are set according to chemical functional groups. By default, for neutral molecules in water, the program calculates a Lewis dot structure for the molecule or system, scans the Lewis file for radius information for each atom and sets radii for relevant atoms,

Table 10.2. Determination of Calculations of Analytic Corrections for SCF Iterations

Variable	Value	Description <sup>a</sup>
<i>jcor</i>	0	No Coulomb terms calculated analytically
	1	Atomic analytic corrections of the form <aalaa> calculated for <b>J</b>
	3	Analytic corrections of the form <aalaa> and <aalbb> calculated for <b>J</b>
	4	Analytic corrections of the form <aalaa>, <aalab>, <aalbb>, and <aalbc> calculated for <b>J</b>
	5	Analytic corrections of the form <aalaa>, <aalab>, <aalbb>, <abalab>, and <aalbc> calculated for <b>J</b> (diatomic + <aalbc>)
<i>kcor</i>	0	No exchange terms calculated analytically
	1	Atomic analytic corrections of the form <aalaa> calculated for <b>K</b>
	2	Diatomic analytic corrections of the form <aalaa>, <aalab>, <aalbb>, and <abalab> calculated for <b>K</b>

a. a, b, and c refer to distinct atoms.

then sets any radii not determined by the Lewis file according to the **atomic** section or the standard, default value. Settings for radii not included the Lewis file are described in [Section 4.5 on page 58](#) and [Section 9.8 on page 218](#) and are listed in [Table 9.43 on page 223](#). If you do not want the atomic radii that determine the dielectric continuum boundary to change according to the chemical environment of the atom, set the solvation keyword **isurf** to 0 in the **gen** section. Otherwise, Jaguar will alter some radii for neutral molecules by using the `default.lewis` file from the data directory, unless you specify your own `.lewis` file in a LEWISFILE line in the input file, as described in [Section 9.1 on page 161](#).

If radii are set according to a Lewis file, Jaguar first computes a Lewis dot structure for the input geometry to determine each atom's bonds and hybridization type. The element and chemical environment of each atom determine its atom type. When Jaguar reads the Lewis file, it sets the atom's van der Waals radius to the value dictated by the *first* atom type description in the Lewis file that matches that atom. For instance, if the atom were a methyl carbon and the first atom type description in the file was of a carbon bound to a hydrogen, the radius would be set to the radius matching that description, even if a later line in the Lewis file described a carbon bound to three hydrogens.



Atom types are determined by an atom's element and by any combination of the following other properties:

- Hybridization (for example,  $sp^2$ )
- Bonding type, which is determined by the bond orders of the bond(s) the atom forms and the element(s) to which the atom is bonded
- Hybridization type, which describes the hybridization and element of atoms to which the original atom is bonded
- Ring size: the size of the ring the atom is in (for instance, 6 for a carbon in benzene)
- Aromaticity of the ring the atom is in, if any. An aromatic ring is defined here by the Huckel Rule: if the ring contains  $4n + 2$  pi electrons, where  $n$  is any non-negative integer, it is considered to be aromatic.

The Lewis file first determines the bonding types and hybridization types that will be recognized, then lists atomic radii for various atom types. The file contains different versions of this information for LMP2 calculations than it does for other wavefunction types. Therefore, the first non-blank line of the file should begin

```
CALCULATION TYPE 01
```

(with any comment allowed after this string), indicating that the information following that line is for HF, DFT, or GVB wavefunctions. After all the information in the file for these calculations, the file should contain this line:

```
CALCULATION TYPE 02
```

followed by information for LMP2 wavefunctions.

### 10.6.1 Describing Bonding Types in the Lewis File

The bonding type information for HF, DFT, or GVB wavefunctions should follow the first line describing the calculation type. The first line of this information should begin

```
BONDING TYPE 01
```

and the rest of the bonding type information should not contain any blank lines except the last line, which signals the end of bonding type information.

Bonding type information should be listed for each relevant element in turn. The information for the first atom should follow immediately after the `BONDING TYPE 01` label. The first character of the information for that atom should begin with the atom's atomic number. The following lines should describe up to five "groups" of bonds for that atom. Each group must begin with the word

```
Group
```

(with no leading spaces) and must contain information for bond orders 1, 2, and 3, with a comment identifying each bond order. The group is a list of bonded atoms and bond orders for the element being described—for example, Group 2 for carbon could describe C=C and C=O bonds by specifying that for bond order 2, Group 2 contains two elements with atom numbers 6 and 8. The first line under each bond order label must list the number of elements in the group for that bond order (2 for the example); if this number is nonzero, the next line must list the atomic numbers for those elements (6 and 8 in the example).

Here is the beginning of a sample `.lewis` file illustrating a list of bonding type information for carbon, including some comments to further explain the file format:

```
CALCULATION TYPE 01 (HF/DFT/GVB)

BONDING TYPE 01 INFORMATION
6 CARBON
Group 1: C-H bonds (only "Group" must be here; the rest is a comment)
  Bond order: 1 (this should be a non-blank comment line)
    1 element
    1 (the atomic number of H)
  Bond order: 2 (this should be a non-blank comment line)
    0 elements
  Bond order: 3 (this should be a non-blank comment line)
    0 elements
Group 2: C=C and C=O bonds
  Bond order: 1
    0 elements
  Bond order: 2
    2 elements
    6 8
  Bond order: 3
    0 elements
```

The number of spaces at the beginning of the lines described above is irrelevant for all lines except the “Group” lines.

After all the groups have been specified for a particular atom, the file should contain a line containing three asterisks (\*\*\*) to indicate the next element’s bonding types are about to be described (in the same format). After all desired bonding types are described for all appropriate elements, the bonding type information should end with a blank line.

## 10.6.2 Describing Hybridization Types in the Lewis File

The hybridization type information in the Lewis file includes up to five groups for each element described, where each group indicates a set of elements and hybridizations for those elements. The hybridization applies to the atom to which the original element is bonded. The information for hydrogen’s first group, for instance, could list C (atomic number 6) with  $sp^2$  hybridization, allowing a later line in the Lewis file to set a particular radius for hydrogen atoms bonded to  $sp^2$  carbons.

The format of the hybridization type information is very similar to that of the bonding type information. The first line of this information (for HF, GVB, or DFT calculations) should begin

```
HYBRIDIZATION TYPE 01
```

and the rest of the hybridization type information should not contain any blank lines except the last line, which signals the end of hybridization type information.

Hybridization type information should be listed for each relevant element in turn. The information for the first atom should follow immediately after the “HYBRIDIZATION TYPE 01” label. The first character of the information for that atom should begin with the atom’s atomic number. The following lines should describe up to five hybridization “groups” for that atom. Each group must begin with the word

Group

(with no leading spaces). The group is a list of bonded atoms for all relevant hybridization types of those bonded atoms—for instance, Group 2 for hydrogen could describe hydrogens bonded to sp carbons by listing carbon’s atomic number under an sp hybridization label. Because there is no default number of hybridizations described for each group (unlike for the bonding type information, where each group contained sets for three bond orders), the first line under each group label must begin with the number of hybridizations described for that group (after any number of spaces).

The next line dictates a hybridization for the bonded elements about to be described. Hybridization labels *must start with five spaces*, followed by one of the following character strings:

```
s hybridization
p hybridization
d hybridization
sp hybridization
sp2 hybridization
sp3 hybridization
sp3d hybridization
sp3d2 hybridization
```

For each hybridization, the bonded elements with that hybridization are then listed in two lines, the first indicating the number of elements and the second indicating the elements themselves, as for the bonding type information.

Information for any following atoms should be preceded by a line with three asterisks, and a blank line indicates the end of the hybridization type information, as for the bonding type information.

The beginning of the hybridization information in a sample `.lewis` file, illustrating a list of hybridization type information for hydrogen and carbon, is shown below, with some comments to further explain the file format:

---

```

HYBRIDIZATION TYPE 01 INFORMATION
1 HYDROGEN
Group 1: H-C(sp2) bonds
  1 hybridization
    sp2 hybridization ("sp2 hybr..." MUST have 5 spaces before it)
      1 element
      6
Group 2: H-O(sp3) bonds
  1 hybridization
    sp3 hybridization
      1 element
      8
***
6 CARBON
Group 1: C-C(sp3) bonds
  1 hybridization
    sp3 hybridization
      1 atom
      6

```

The number of spaces at the beginning of the lines described above is irrelevant for all lines except the “Group” lines and the hybridization labels.

After all desired hybridization types are described for all appropriate elements, the hybridization type information should end with a blank line.

### 10.6.3 Setting van der Waals Radii From Lewis File Data

The Lewis file can be used to make non-default choices for van der Waals radii of atoms in particular chemical environments, or even to reset the default radii for particular elements. After Jaguar’s lewis program analyzes an input geometry’s Lewis dot structure, it sets the atom’s van der Waals radius to the value dictated by the *first* atom type description of element and chemical environment in the Lewis file that matches that atom with no contradiction. If no such matching description exists in the Lewis file, the atom is assigned the default radius for that element.

Atom type descriptions in the Lewis file should be preceded by a heading beginning

```
RADII TYPE 01
```

for information applying to HF, GVB, or DFT wavefunctions, or

```
RADII TYPE 02
```

for information for LMP2 wavefunctions. After that, each atom type description is listed. Blank lines are allowed in an atom type description list, and as long as some spacing exists between numbers and comments on each line, the number of spacing characters is irrelevant. However, keep in mind that the order of the atom type descriptions is important since the first matching description will always be used.

Each line describing an atom type has six integers, one real number, and an optional comment, in that order. The integers describe the atom type, while the real number sets the radius in angstroms for that atom type. The six integers describe the following characteristics, in turn:

- Atomic number (for instance, 6 for carbon)
- Hybridization of the atom itself
- Bonding type of the atom (elements it is bound to and order of those bonds)
- Hybridization type of the atom (hybridization and elements of atoms to which it is bound)
- Size of ring (if any) the atom is in
- Aromaticity of that ring according to Huckel Rule (aromatic rings have  $4n + 2$  pi electrons, where  $n$  is a non-negative integer)

All six integer values and a corresponding radius value must always be listed in an atom type description line, and the atomic number must correspond to an actual element. However, any or all of the other five integer values can be set to  $-1$ , a wild card entry indicating that any value for that characteristic matches that atom type description. To reset a default radius for hydrogen, for instance, you could put the following line before any other descriptions of hydrogen atoms:

```
1 -1 -1 -1 -1 -1 1.10 H all 1.1, ignoring chemical environment
```

and the van der Waals radius for all hydrogen atoms would be set to  $1.10 \text{ \AA}$ .

To describe the hybridization of the atom itself, the atom type description line's second integer should take on one of the values indicated in [Table 10.3](#).

The description of the atom's bonding type uses the groups listed in the bonding type information described in [Section 10.6.1 on page 255](#) (unless it is  $-1$ ). Any positive integer for bonding type describes the number of bonds the atom has in each of the bonding type groups for its element and/or the number of all other bonds the atom has. A bonding type group describes elements of bonded atoms and orders of those bonds, as described in [Section 10.6.1](#). The third integer in an atom type description line determines how many bonds the atom forms of each bonding type group  $g$  for an atom of a particular element, where  $g$  indicates the order of the bonding type groups listed for that element. The number of bonds from group  $g$  is indicated by the  $10^g$  digit in the integer.

For example, if  $g$  were 1 and the atom being described were carbon,  $g$  would correspond to the first bonding type group listed for carbon, and a bonding type integer value of 40 ( $4 \times 10^1$ ) would indicate that that carbon atom had four bonds from carbon's Group 1 bonding type information. If the Lewis file contained the bonding type information example provided in [Section 10.6.1](#), which included the lines:

Table 10.3. Lewis File Hybridization Numbers and Corresponding Hybridization Types

Hybridization Number	Corresponding Hybridization Type
1	s hybridization
2	p hybridization
3	d hybridization
5	sp hybridization
6	sp <sup>2</sup> hybridization
7	sp <sup>3</sup> hybridization
8	sp <sup>3</sup> d hybridization
9	sp <sup>3</sup> d <sup>2</sup> hybridization

```
6 CARBON
Group 1: C-H bonds (only "Group" must be here; the rest is a comment)
  Bond order: 1 (this should be a non-blank comment line)
    1 element
    1 (the atomic number of H)
```

the integer value of 40 would describe a methane carbon. The same sample Lewis file information, whose key Group 2 information for carbon appears in these lines:

```
Group 2: C=C and C=O bonds
  Bond order: 1
    0 elements
  Bond order: 2
    2 elements
    6 8
```

would mean that this radius information line

```
6 -1 120 -1 -1 -1 2.00 C in H2-C=C or H2-C=O
```

would describe a carbon atom (6) with one bond from carbon's Group 2 (a double bond to either C or O) and two bonds from carbon's Group 1 (single bonds to H), and would set such an atom's radius to 2.00 Å (unless another matching description preceded that line).

The rightmost digit in the integer describing bonding type specifies the number of bonds formed by the atom which are not of any of the forms described in the groups for that atom's bonding type information. A double or triple bond counts as one bond, not two or three, and lone pairs should not be included in the bond count.

The digits of the bonding type integer must describe *all* of an atom's bonding in order to match the atom information. For example, if the Lewis file described above contained no group for C-C bonds in the bonding type information, the integer "200" would only

describe a carbon atom with one double bond to another C or O and no other bonds, while the integer “202” would adequately describe a carbon with one double bond to another carbon and two single bonds to other carbon atoms.

The fourth integer in an atom type description, which describes hybridization type, or the elements and hybridization of the atoms to which an atom is bound, works almost the same way as the integer describing bonding type. As it does for bonding types, the digit *g* places from the rightmost digit in the integer represents the *g*th group in the hybridization type information for that element (see [Section 10.6.2 on page 256](#) for more information), while the rightmost digit specifies the number of bonds to elements and hybridization types that do not fit into any of the groups described for the element of the atom being evaluated. For example, suppose only one hybridization group were described for carbon in the sample Lewis file, as follows:

```
6 CARBON
Group 1: C-C(sp3) bonds
  1 hybridization
    sp3 hybridization
      1 atom
        6
```

Then this atom type description line in a Lewis file would accurately match the middle carbon in methylethylene ( $\text{H}_2\text{C}=\text{CH}-\text{CH}_3$ ):

```
6 -1 -1 12 -1 -1 2.00 C in H2-C=C or H2-C=O
```

as would the following line, which also contains the proper settings for the middle carbon’s hybridization and bonding type:

```
6 6 111 12 -1 -1 2.00 C in H2-C=C or H2-C=O
```

As for the integer describing bonding type, the total of the digits in the fourth integer should be the same as the number of bonds (three for this example, remembering that the double bond counts as one bond)—that is, all bonds should be accounted for (unless, of course, the integer is  $-1$ ).

The fifth and sixth integers describe the ring the atom is in, if any. If the fifth integer is a positive number *n*, it indicates that the atom description corresponds to an atom in a ring of size *n*. For example, a benzene carbon is in a ring of size 6. If the fifth number is a negative number  $-n$ , the description corresponds to an atom in a ring of size *n* or smaller, unless the fifth integer is  $-1$ , in which case the question of the atom’s ring environment is ignored completely. The size *n* should not be more than 20.

The sixth integer indicates whether the description corresponds to an atom in an aromatic ring as defined by the Huckel Rule ( $4n + 2$  electrons in ring, where *n* is a non-negative integer). If the sixth integer is 1, the description corresponds to an aromatic ring; if it is 0, the description corresponds to a non-aromatic ring; and if it is  $-1$ , the aromaticity of the ring is irrelevant. Note, however, that aromaticity is *not* evaluated if the fifth integer

(describing ring size) is  $-1$ . To describe aromaticity without regard to ring size, you should generally set the fifth integer to  $-20$  and the sixth to  $1$ , corresponding to atoms in aromatic rings of size 20 or less.

### 10.6.4 Default Behavior for Setting Radii

The radius settings contained in Jaguar's `default.lewis` file are used for any relevant atoms in all default solvation calculations in water with Jaguar's solvation module, except for calculations on ions or on molecules containing atoms with atomic numbers greater than 18. By default, the program uses the first Lewis dot structure generated to evaluate the radii, and the solvation calculation also includes a correction term (the first shell correction factor) that depends on that Lewis dot structure. If the Lewis dot structure does not correspond to that desired for the molecule, the keyword **lewstr** should be changed to correspond to a better structure, as described in the **gen** section description in [Section 9.5 on page 168](#). To avoid using Lewis dot structures for either correction terms or radius settings, set the **gen** section keyword **isurf** to 0. To use Lewis dot structures to set radii but not for correction terms, **isurf** should be 0 but the keyword **ivanset** should be 1. All Lewis dot keywords are explained in [Section 9.5.5 on page 170](#).

The radius settings in the file `default.lewis`, which appears in the standard data directory, were optimized for HF, GVB, 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 134. All calculations used a 6-31G\*\* basis set. Geometries were obtained from gas phase optimizations at the HF, GVB, and LMP2 levels. For both the geometry optimizations and the solvation energy calculations, the GVB and LMP2 treatment was restricted to heteroatom pairs.



---

# Chapter 11: Running Jobs

---

Running, monitoring and controlling jobs is done by the Maestro 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 *Maestro User Manual*. Some of this information is repeated here.

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 the next section.

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 Customizing Host Configurations

The installation directory (`$SCHRODINGER`) contains a file named `schrodinger.hosts` that identifies hosts on which Jaguar can be run and provides some information about the use of the host. When you start Maestro, the settings in the `schrodinger.hosts` file are used to determine the available options in the Jaguar Run window. You can copy and edit the `schrodinger.hosts` file to customize its settings. *You shouldn't need to update any schrodinger.hosts files when you later install other versions of Jaguar.* Information on the `schrodinger.hosts` file is given in the *Schrödinger Product Installation Guide*, and includes instructions for batch queue configuration. No information on batch queue configuration is given in this chapter.

Maestro searches the following directories for a `schrodinger.hosts` file, in the order given, and uses the first one that it finds.

- The directory in which you started Maestro
- `$HOME/.schrodinger`
- `$SCHRODINGER`

You can always determine which `schrodinger.hosts` file is being used by clicking the About button in the Jaguar panel, then clicking the Schrödinger button in the About window. The `schrodinger.hosts` file currently used by Jaguar is listed near the bottom of the window.

If you want to change entries in the `schrodinger.hosts` file, you should copy and edit your own `schrodinger.hosts` file. If there is no `schrodinger.hosts` file in your `$HOME/.schrodinger` directory or the directory from which you start Maestro, you should identify the `schrodinger.hosts` configuration file currently used by Maestro, copy this `schrodinger.hosts` file to your `$HOME/.schrodinger` directory or the directory where you want to start Maestro, and edit it there.

The following is an example of a `schrodinger.hosts` file:

```
# Schrodinger hosts file
#
name:      localhost
schrodinger: /software/schrodinger
#
name:      ahost
#
name:      bhost
#
name:      old_bhost
host:      bhost
schrodinger: /software/schrodinger_old
#
name:      another_host
processors: 2
tmpdir:    /scr
schrodinger: /usr/bin/share/schrodinger
#
# End of Schrodinger hosts file
```

The hosts file consists of one or more entries that describe a host on which jobs can be run. Typically, there will be a single entry for each machine on which you want to run jobs. For each entry in the `schrodinger.hosts` file, the following settings can be made:

```
name: entry-label
host: machine-name
user: userid
tmpdir: tmpdir
processors: number of processors
schrodinger: installation-path
```

The settings are described in the following sections. A full list of settings, including settings for batch queue configuration, is given in the *Schrödinger Product Installation Guide*.

A setting in the `schrodinger.hosts` file can be formatted with any combination of spaces and tabs, but the entire setting must be on one line. Comments may be included in the `schrodinger.hosts` file, and should start with a hash sign (#).

If you have installed Jaguar on multiple machines, you might need to edit the `schrodinger.hosts` file on each machine to add entries for the other machines.

### 11.1.1 The name and host Settings

The `name` setting must be the first line for each machine. This is the name that is displayed in the list of known hosts in the Job Host menu of the Jaguar Run window. Usually, *entry-label* is the name of a machine (a host) that can be used to run a Jaguar calculation, but if it is not, you must include a `host` setting that supplies the machine name. The `host` setting is only needed if the name line does not give the machine address. You might, for example, want to provide an alias in the `name` setting and define the host name in a `host` setting if the host name is long. Another possible use of multiple entries for a single machine is to specify different settings on a machine, such as different scratch directories or different software installations. You can also use the `name` and `host` settings to specify a batch queue name and the host on which the batch system is available.

The host name does not need to include the full Internet address unless the host on which you plan to run (the calculation host) is not on the same local network as the host from which you plan to submit jobs (the submission host).

The value `localhost` is a special name setting that means the host from which the job was submitted. In addition to this function, the `localhost` entry sets the default values of settings for all other entries. In the `schrodinger.hosts` file example above, the host entries `ahost` and `bhost` inherit the `schrodinger` setting from the `localhost` entry.

### 11.1.2 The user Setting

If you have different user IDs on the submission and calculation hosts, you must include a `user` setting for the calculation host in the `schrodinger.hosts` file on the submission host. This `schrodinger.hosts` file must be a local copy.

### 11.1.3 The tmpdir Setting

The `tmpdir` setting specifies a directory where scratch directories can be created, such as `/scr` or `/temp`. From this setting, Jaguar creates a scratch directory named `/tmpdir/userid/jobname` to store temporary files, where *userid* is your account name on the calculation host and *jobname* is the name of the Jaguar job. For example, if the user `erwin` ran a job named `h2o` on the host `withi` using the `schrodinger.hosts` file above, the temporary directory used for the job would be `/temp/erwin/h2o`. By default, Jaguar removes this subdirectory when the job is completed, after copying back all important files to the output directory, unless the subdirectory already existed when the job started.

You can override the `tmpdir` setting in the `schrodinger.hosts` file by setting the `SCHRODINGER_TMPDIR` environment variable. For example, if the directory designated by `tmpdir` becomes full with files that you don't have permission to delete, you can set `SCHRODINGER_TMPDIR` to a different directory and continue to run Jaguar jobs.

Instead of using `tmpdir` or `SCHRODINGER_TMPDIR`, you can directly specify the full path to the scratch directory in the `JAGUAR_SCRATCH` environment variable.

### 11.1.4 The processors Setting

For stand-alone computers with multiple processors, set `processors` to the number of processors in the computer. For computer clusters, set `processors` for each node to the *total* number of processors in the entire cluster.

## 11.2 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, with any installed version of Jaguar
- Kill a Jaguar job that you started on any machine at your site
- List the machines on which Jaguar is installed
- List the jobs that are running on a particular machine

If Jaguar is installed on more than one machine at your site, you can use the `jaguar` command on one machine to run, kill, or list 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](#). The options may be given in any order, and may precede any options specific to the command.

The *jobnames* argument to the `jaguar` command is a list of names. Each name in the list is the name of a Jaguar job that is run, and each name also specifies an input file. The name can be given with or without a `.in` extension. If the `.in` extension is given, Jaguar removes it to form the job name. If the `.in` extension is not given, Jaguar adds it to form the input file name. For example, the commands

```
jaguar run h2o
jaguar run h2o.in
```

both run a Jaguar job with the job name `h2o` and the input file `h2o.in`.

Table 11.1. Commands for the `jaguar` Command

Command	Description
<code>run</code> [ <i>version-args</i> ] [ <i>runargs</i> ] <i>jobnames</i>	Start the Jaguar jobs whose job names are listed, using the specified version information and run time options.
<code>batch</code> [ <i>batch-options</i> ] <i>script</i> [ <i>jobnames</i> ]	Start a Jaguar batch job using the specified script. The optional job names specify input files for the script. See <a href="#">Section 11.3 on page 275</a> for more information on this command.
<code>pka</code> <i>jobname</i>	Start a Jaguar pKa calculation.
<code>j2</code> <i>jobname</i>	Start a Jaguar J2 calculation.
<code>babel</code> [ <i>babel-options</i> ]	Perform a file format conversion using Babel.
<code>nbo</code>	Run an NBO calculation.
<code>results</code> <i>options</i>	Summarize results from the output file using the options specified. See <a href="#">Section 6.1 on page 97</a> for more information on this command.
<code>jobs</code> [ <i>jobnames</i>   <i>jobids</i>   <i>status</i>   <code>all</code> ]	Show the status of the specified running Jaguar jobs or list the jobs that have the specified status. The <code>all</code> option shows the status of all jobs, including completed jobs. The output lists the job ID, the job name, the status.
<code>kill</code> { <i>jobnames</i>   <i>jobids</i>   <i>status</i> }	Kill the specified Jaguar jobs or all jobs that have the specified status. This command is processed immediately.
<code>purge</code> [ <i>jobnames</i>   <i>jobids</i> ]	Remove records for the specified jobs from the job database. If no <i>jobname</i> is given, all completed jobs are purged.
<code>stop</code> [ <i>jobnames</i>   <i>jobids</i>   <i>status</i> ]	Stop the specified Jaguar jobs or all jobs that have the specified status when the currently running executable has finished.
<code>machid</code>	Report the hardware and software configuration. This command gives the same output as the <code>\$SCHRODINGER/machid</code> command.
<code>platform</code>	Report information on the hardware platform. This command gives the same output as the <code>\$SCHRODINGER/platform</code> command.
<code>scripts</code>	List the available batch scripts.
<code>sysreq</code>	Report any system requirements for Jaguar and whether they are met.
<code>help</code>	Display a command syntax summary including a list of valid commands.

The job control functions of the `jaguar` command (`jobs`, `kill`, `stop`, and `purge`) are now interfaces to the `jobcontrol` command with `Jaguar` selected as the program. For instance, `jaguar jobs` actually executes

```
jobcontrol -list program=jaguar
```

In addition to running the commands listed in [Table 11.1](#), you can use the `jaguar` command with the options listed in [Table 11.2](#) to obtain information about Jaguar versions and hosts available. Some of these options can be qualified by arguments that limit or define the list of versions displayed. These version arguments are listed in [Table 11.3](#). To find out about versions available on remote hosts, you can add the qualifier `-HOST hostname`. For example, to check whether version 4.2 of Jaguar is installed on the host `freda`, you could use the following command:

```
jaguar -WHICH -HOST freda -REL v42
```

Further examples are given in the next few sections.

**Note:** The options listed in [Table 11.2](#) and [Table 11.3](#) apply to all Schrödinger programs, not just to Jaguar.

Table 11.2. Information Options for the `jaguar` Command

Option	Description
<code>-WHICH [version-args]</code>	Show which version of <code>jaguar</code> and of the <code>mmshare</code> library would be used for the given <i>version-args</i> .
<code>-LIST [version-args]</code>	List the available versions of Jaguar that can be run on the specified host. If no host is specified, the local host is used. If <i>version-args</i> is <code>-ALL</code> , list all available versions of Jaguar, even if not compatible with the specified host.
<code>-HOSTS</code>	List the hosts that are available for Jaguar calculations.
<code>-ENTRY</code>	Show the section of the <code>schrodinger.hosts</code> file that will be used for this job.
<code>-WHY [version-args]</code>	Gives information about why the specified version was selected.

Table 11.3. Version Options

Option	Description
<code>-REL version</code>	Release version number: <code>v42</code> , <code>v4.2</code> , <code>42</code> , <code>v42062</code> , <code>41059</code> , <code>v4.1.049</code> are all acceptable forms.
<code>-VER pattern</code>	Pattern to match in the path to the executable. Replaces <code>-v</code> .
<code>-ARCH platform</code>	Platform code, e.g., <code>Linux</code> , <code>IRIX-mips4</code> .

## 11.2.1 Selecting a Calculation Host

If Jaguar is installed on several machines at your site, you can use the `jaguar` command to help determine which host you should use to run your job. To determine which local 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 11.1 on page 263](#) for a description of the `schrodinger.hosts` file.

## 11.2.2 Selecting Particular Jaguar Executables

By default, Jaguar looks for the executables available for the machine upon which you want to run a Jaguar job, then uses the most recent Jaguar executables for that machine type. However, if you have several differing sets of Jaguar executables at your site, such as different versions of Jaguar or executables for different machine types, you can choose to run your Jaguar job with a non-default choice of executables.

To determine which sets of Jaguar executables are available, enter the command

```
jaguar -LIST
```

to find out about executables on the current host, or

```
jaguar -LIST -HOST hostname
```

to find out about executables on another machine.

## 11.2.3 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 11.4](#) and described below. The first three options are common to all Schrödinger programs. You can also use the version options listed in [Table 11.3](#).

**Note:** The single-letter options `-h` and `-v` are no longer supported. The options `-F`, `-n`, `-p`, `-s`, and `-w` are still supported, but we cannot guarantee that they will continue to be supported. You should use the new equivalents.

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 3.7 on page 44](#) 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](#).

Table 11.4. Options for the `jaguar run` Command

Option	Effect	Default Behavior
<code>-HOST</code> <i>hostname</i>	Run a Jaguar job on the specified host or submit a Jaguar job to the specified batch queue. (Replaces <code>-h</code> .)	Run a Jaguar job on the local host
<code>-USER</code> <i>username</i>	Specify the user name to be used for remote jobs. Must be used with <code>-HOST</code> .	Use the same user name as on the job submission host.
<code>-WAIT</code>	Wait for the Jaguar job to finish before returning to the command prompt. (Replaces <code>-w</code> .)	Return to the command prompt immediately.
<code>-SAVE</code>	Save temporary files and temp directory for job at end of job. (Replaces <code>-s</code> .)	Temporary files are cleaned out of temp directory and temporary directory is removed at end of job
<code>-PROCS</code> <i>nprocs</i>	Use <i>nprocs</i> processors for a parallel job. (Replaces <code>-p</code> .)	Run a serial job
<code>-NICE</code>	Run Jaguar executables with nice -19. (Replaces <code>-n</code> .)	Jaguar executables are run without nice
<code>-FORCE</code>	Force the scratch directory to be overwritten if it exists. (Replaces <code>-F</code> .)	Abort the job if a scratch directory named for the job already exists
<code>-t</code>	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
<code>-DEBUG</code>	Print debug information in the terminal window. This information is useful if you need to contact technical support.	Do not print debug information.

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*. Jaguar supplies the `.in` extension if you omit it. With this command, the job runs on the machine upon which you have submitted the command, and uses the most recent version of Jaguar.

To run a Jaguar job on another machine, use a command in this form:

```
jaguar run -HOST hostname jobname
```

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 named `ch4` on a machine named `beta`, you would enter

```
jaguar run -HOST beta ch4
```



To run a Jaguar job on the machine *hostname* with a particular, non-default set of executables, you can use the command

```
jaguar run -HOST hostname -VER version jobname
```

where *version* is any string that appears in one of the executable directories listed for that host by the command.

```
jaguar -LIST -HOST hostname
```

The string must be unique to ensure that the desired executables are selected.

The `jaguar run` command has several other command line options, as shown in [Table 11.4](#). For example,

```
jaguar run -NICE -SAVE jobname
```

causes executables to be run with a lower CPU scheduling priority (see 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* 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 processors to use to the host name, separated by a colon, as in the following example.

```
jaguar run -HOST beta:2 ch4 nh3
```

You can also use `jaguar batch` to run multiple jobs. See [Section 11.3 on page 275](#) for more information.

### 11.2.4 Killing a Jaguar Job

The `jaguar kill` command lets you kill any Jaguar job you are running on any host. When you use the `jaguar kill` command, the temporary directory for your job still exists and contains *all* files generated during the job, and no output files are copied back to your output directory.

To kill one of your Jaguar jobs, enter the command

```
jaguar kill jobname
```

This command checks all hosts for the specified job and kills all instances of the job with the name *jobname*. To kill a specific job, use the job ID (which is unique) instead of the job name. You cannot kill stranded jobs with `jaguar kill` because the job control facility does not have the necessary information about those jobs.

## 11.2.5 Converting File Formats

Jaguar uses the Babel program [24] 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 9.5.18 on page 201](#) 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] -i input-file [-h|-d] [range]
              -o [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 9.30 on page 201](#); the input format keywords are listed in [Table 11.5](#).

Table 11.5. 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

Table 11.5. Input Format Keywords and File Types for Babel File Format Conversions (Continued)

<b>Format Keyword</b>	<b>File Type</b>
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
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

Table 11.5. Input Format Keywords and File Types for Babel File Format Conversions (Continued)

Format Keyword	File Type
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
unxyz	UniChem XYZ file
xyz	XYZ file
xed	XED file

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 range 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. To convert between Maestro format and some other formats, a file utility called `jagconvert` has been provided. For conversions between various Schrödinger file formats that are not recognized by Babel, there is a file conversion utility, `jagconvert`. This utility reads and writes Jaguar input (`.in`) files, BioGraf (`.bgf`) files, XMol (`.xyz`) files, and Maestro (`.mae`) files. It also reads Gaussian9x (`.g9x`) files and MacroModel (`.dat`) files but doesn't write them. The utility is located in `$(SCHRODINGER)/utilities`. The command syntax is as follows:

```
jagconvert [intype] infile outtype outfile
```

where *intype* is one of `-ijag`, `-ixyz`, `-ibgf`, `-ig92`, or `-imae`, and *outtype* is one of `-ojag`, `-oxyz`, `-obgf`, or `-omae`. 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.

## 11.3 Running Multiple Jobs: jaguar batch

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 batch` 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 use `jaguar batch`, you need a batch input file (whose name should end in `.bat`) and at least one template Jaguar input file. The batch input file tells `jaguar batch` how to modify the 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 batch` command then generates the input files and runs the corresponding jobs, either consecutively if only one host has been specified, or by distributing the jobs over the specified hosts.

### 11.3.1 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 11.6. The directives apply to *all* jobs described below them, unless a later line of the same type replaces them. Any 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 11.7.

Table 11.6. Batch Input File Directives

Directive	Action
EXEC = <i>directory</i>	Set the directory for the Jaguar executable. This directory can be any directory listed by <code>jaguar -LIST</code> .
SCRATCH = <i>directory</i>	Set the scratch directory. Equivalent to specifying the JAGUAR_SCRATCH environment variable.
TMPDIR = <i>directory</i>	Set the scratch directory root. Equivalent of <code>tmpdir</code> setting in <code>schrodinger.hosts</code> .
WORKDIR <i>directory</i> [ <i>files</i> ]	Create the specified directory and use it as the working directory for input and output. Copy the specified files into the directory.
FLAGS = <i>options</i>	Specify <code>jaguar run</code> command line options
OPTIONS {= +} <i>options</i>	Set options to apply to subsequent jobs. Options can be specified over multiple lines by using = on the first line and + on subsequent lines.
OUTFILES {= +} <i>files</i>	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.
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.

The syntax for job specifications is as follows:

```
template-name [new-name [options]]
```

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 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 batch` 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 batch` with the job name *h2o* 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, or the **gvb** section if you are comparing GVB results to HF, LMP2, or DFT results
- A substitution of a specified number or character 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 11.7](#). 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. After the examples, directions on how to submit a batch job follow in the final subsection.

*Table 11.7. Definition of Options That Are Applied to a Template File to Generate an Input File*

Change	Format	Examples
set keywords	<i>keyword=new-value</i> , or <i>keyword=NONE</i> to remove a setting	<code>basis=lav3p**</code> <code>dftname=b3lyp</code> <code>igeopt=NONE</code>
specify a data file path and name	<i>filetype=fullpathname</i> , or <i>filetype=NONE</i> to return to default choice for that file type	<code>BASISFILE=/usr/es/my.bas</code> <code>ATOMIGFILE=NONE</code> <code>DAFFILE=NONE</code> <code>GRIDFILE=NONE</code> <code>CUTOFFFILE=NONE</code> <code>GPTSFILE=NONE</code> <code>WAVEFNFILE=NONE</code>
remove a section	<code>RMSECTION=section-name</code>	<code>RMSECTION=guess</code> <code>RMSECTION=gvb</code>
clear the <b>gen</b> section except for the <b>multip</b> and <b>molchg</b> settings	<code>RESETGEN</code>	



Table 11.7. Definition of Options That Are Applied to a Template File to Generate an Input File (Continued)

Change	Format	Examples
insert a file at the top of the input	ADDTOP= <i>filename</i>	ADDTOP=guess.in
append a file to the input	ADDEND= <i>filename</i>	ADDEND=guess.in
substitute a value for a variable	<i>old-pattern</i> == <i>new-pattern</i> . Do not use any of the characters <code>*\$!\&lt;&gt;?</code> in either pattern	bond==1.5 torang==170.0

### 11.3.2 Batch Input File Example

As an example, 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. You can create a batch file that automates this process. The batch file would read in each molecular geometry from an existing input file, make the necessary keyword changes, and perform the calculations. Here is an example of such a batch file:

```
# 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
```

### 11.3.3 Running jaguar batch

The syntax of the `jaguar batch` command is:

```
jaguar batch [command-options] batchfile[.bat] [joblist]
```

If the batch script `batchfile.bat` uses `$JOB` in job specifications, you must supply the list of jobs to substitute in `joblist`. In the command, the suffix `.bat` is optional: if it is missing, it is added to the stem, `batchfile`.

The `jaguar` command options `-REL`, `-VER`, `-HOST`, `-USER`, `-WAIT`, and `-PROCS`, described in [Table 11.3](#) and [Table 11.4](#), can be used in the `jaguar batch` 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 select multiple processors either by repeating the host name or by appending a colon and the number of processors to the host name, e.g. `cluster:32`.

There are also some unique command options for the `jaguar batch` command, which are summarized in [Table 11.8](#).

*Table 11.8. The jaguar batch Command Line Options*

---

<b>Option</b>	<b>Description</b>
<code>-c</code>	Create input files, but don't run the batch job.
<code>-r</code>	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.
<code>-l</code>	Lists jobs that would be run if <code>jaguar batch</code> were called without options, but does not generate any files or run any jobs
<code>-s</code>	Lists jobs that would be run and shows the contents of the input files that would be generated if <code>jaguar batch</code> were called without options, but does not generate any files or run any jobs

---

The `-r` option is a restart option, which prevents `jaguar batch` from overwriting existing Jaguar input and output files and from running the jobsteps that created them. The `-l` and `-s` options permit you to see what `jaguar batch` would do, but do not actually allow it to generate any new input files or run any Jaguar jobs.

If you run remote batch jobs, you should ensure that the input and output directories are on a disk system that is available to both the submission and the calculation host, such as a cross-mounted disk or an NFS file system.

---

# Chapter 12: 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. Also feel free to contact us, as described on [page 319](#).

For problems with settings, you might find the information you need in the online help. You can obtain help from any window by clicking its Help button. The Help window is displayed with the appropriate topic selected. You can also open the Help window using the Help button in the Jaguar panel. To choose a topic, double-click the topic in the Help Topics list, or click the topic and then click Select. The title of the topic is displayed in the Selection text box, and the text of the topic is displayed in the text box above.

## 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 12.1.4 on page 284](#) and [Section 12.1.5 on page 285](#).

### 12.1.1 The SCHRODINGER Environment Variable

Before running Jaguar on any particular machine, you must set the environment variable `SCHRODINGER` to point to the installation directory on that machine. This is the directory

containing Jaguar version 5.0, which is in a subdirectory called `jaguar-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:

```
cshtcsh:      setenv SCHRODINGER installation-directory
```

```
sh/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 Including the `jaguar` Command in Your Path

The command `jaguar` is actually a short script that finds the appropriate version of Jaguar to run and passes on any relevant options to the main Jaguar program. If you have set the `SCHRODINGER` environment variable, you can run Jaguar jobs using the command `$SCHRODINGER/jaguar`. It is usually more convenient to include the installation directory in your `PATH` (or `path`) environment variable, so that you do not need to type `$SCHRODINGER/`.

To determine whether `jaguar` is in your path, enter the command

```
jaguar help
```

If the output from this command is a description of how to use the `jaguar` command, Jaguar is already in your path, and you can skip the rest of this subsection. Otherwise, if your machine responded with an error message like

```
jaguar - Command not found
```

you can add the installation directory to your path as follows.

```
cshtcsh:      setenv PATH installation-directory:$PATH
```

```
sh/bash/ksh: export path = (installation-directory $path)
```

### 12.1.3 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*.

```
cshtcsh:      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 “*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 12.1.5 on page 285](#).

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.

If you are using an SGI and you get an error message like this:

```
dgl error (getservbyname): unknown service sgi-dgl/tcp
```

it means Jaguar is unable to find the SGI Distributed Graphics Library. The file `/etc/services` should contain this information in a line beginning `sgi-dgl`. If this line is commented out (that is, if it begins with a “#” character) you can try uncommenting it. If you continue to have this problem and it is affecting the GUI performance you should ask your system administrator for help.

### 12.1.4 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 Temp directory setting in the Jaguar Run window. 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 Temp directory listing of `/scr/erwin` for a job called `h2o` (with an input file called `h2o.in`), 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 Run window 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.

### 12.1.5 Problems Running Jaguar Calculations on Other Nodes

In order to launch jobs on other nodes and for these nodes to copy files back to the host from which they were submitted, the nodes must be able to run `rsh` (remote shell) and `rsh` (remote copy) commands on each other. If you get a “Permission denied” error when trying to start a job (by selecting OK in the Run window, as described in [Section 3.1 on page 23](#) and [Section 3.6 on page 38](#)), the `rsh` command is not being allowed. This problem may occur even if the job submission host (the “local host”) and the host where the calculation is to be performed (the “calculation host”) are the same. The best method to test whether this problem is occurring is to issue individual `rsh` commands at a local host command line prompt, such as

```
rsh calculation-host who
```

where you substitute the name of the host where you want to perform the calculation for *calculation-host*.

If both the local and calculation hosts are on the same local network, ask your system manager about allowing `rsh` commands between the two, which could be done in several ways, depending on your system. One way is to list hosts which are allowed to connect using `rsh` to a given host in its `/etc/hosts.equiv` file. It may be necessary to include the name of the local host in its own `/etc/hosts.equiv` file if the calculation is to be done on the local host. See your system manager or your UNIX documentation concerning trusted hosts, NIS domains, or networking for more information.

If you get an error which 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 Temp directory bar in the Run window, and that you have permission to create that directory if it does not already exist.

If you are unable to allow `rsh` commands as described above (e.g., your local and calculation hosts are not local to each other), you must include the local machine in a `.rhosts` file in your home directory on the calculation host, and vice versa. If you have the same user name on both nodes, a line in the `.rhosts` file only needs to contain the entire host name. For more information, see the `rhosts` man page on your machine.

One further complication can result if you have distinct user names on the local and calculation hosts. In this case, you may get an error like one of the following:

```
Login incorrect
remshd: Login incorrect
rshd: xxxx-xxxx The remote user login is not correct.
```

This problem generally occurs only when the local and calculation hosts are on separate local area networks. To handle these distinct sites, 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:

```
host:   sgi username@calculation-hostname
```

where the name of the machine in the `host:` field matches that in the `uname -n` command output for that machine, `username` is replaced by your user name, and `calculation-hostname` is replaced by the name of your calculation host. See [Section 11.1 on page 263](#) 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.

- *You cannot read in a particular file as input.* Make sure you are specifying the right file type in the text box under the directory and file name lists. The File type selection alters both the extensions for the file names listed (which can be changed by editing the Filter text box), but also determines the format the file is expected to have. Also, make sure the file name, and not just its directory, is really showing up in the Selection text box before you click OK. See [Section 3.4 on page 34](#) for more information.
- *The molecular structure for the calculation is not what you expected it to be.* If you read in a Jaguar input, `.bgf`, `.hes`, `.dat`, or GAUSSIAN 92 input file, or if you read in a file containing only a geometry, the geometry is obtained from that file, unless you edit the geometry after reading the file. Any geometry you entered before reading the file is erased. Also, if you symmetrize the geometry or set symmetry on for the calculation, as described in [Section 3.5.2 on page 37](#), 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 or a GAUSSIAN 92 input file, some of the settings in the file take precedence over settings previously made in the GUI. See [Section 3.4 on page 34](#) 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  $1.0 \times 10^{-6}$ .
- *For a GVB job, the program exits early and the output states you need a different number of lone pairs on a particular atom.* As described in [Section 4.3.1 on page 57](#)



and [Section 7.1.2 on page 141](#), you must specify lone pairs for either all or none of the lone pairs on any particular atom. Change the lone pair information and try running the calculation again.

- *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. If the structure is reasonable, convergence problems should not occur, and we would appreciate it if you would describe them to us at the address given on [page 319](#), preferably by e-mailing us 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 and/or setting the accuracy level to ultrafine, both of which are described in [Section 4.9.4 on page 77](#) and [Section 4.9.5 on page 78](#). The calculation will be slower, but convergence may be better for problem cases.
- *The settings available in the Read, Save, or Jaguar Run windows are not what you expected them to be.* Many of the options for these windows are determined by the `schrodinger.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 local host is used. You can find out which `schrodinger.hosts` file is being used by clicking About in the Jaguar panel, then clicking Schrödinger and looking at the configuration file listed. If you are using a different `schrodinger.hosts` file than you expect, or if you are working with a new version of Jaguar and a new `schrodinger.hosts` file has been installed on your system, you should examine the `schrodinger.hosts` file for the job and make sure it is in the same form as the file for the system for the version of Jaguar that you are using, and that the settings are appropriate. See [Section 11.1 on page 263](#) 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 9.5.24 on page 212](#) 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 319](#), if you would like any tips for setting memory use for your machine.



---

## Chapter 13: Parallel Jaguar

---

The parallel implementation of Jaguar is based on MPI (Message Passing Interface). Jaguar can run on SMP (symmetric multi-processing) shared-memory architectures, such as workstations that contain multiple processors, and it can run on distributed-memory architectures, such as IBM SP clusters or Linux Beowulf clusters. Jaguar can also run on clusters in which each node contains multiple processors. The development of parallel Jaguar is discussed in references [136] and [137].

The following kinds of jobs can be run in parallel:

- HF and DFT single-point calculations (in gas phase or in solution)
- HF and DFT geometry optimizations (in gas phase or in solution)
- Closed-shell LMP2 single-point calculations

If you want to compute analytic frequencies, you probably should run your Jaguar energy calculation or geometry optimization first in parallel, then use the restart job for a frequency calculation in serial mode.

Jobs that cannot be run in parallel mode include:

- Jobs that use all-analytic SCF methods
- LMP2 jobs other than closed-shell single-point calculations
- LMP2 jobs with more processors than LMP2 orbitals
- GVB, GVB-RCI, and GVB-LMP2 jobs
- CPHF (hyper)polarizability jobs
- Jobs with more processors than atoms
- pK<sub>a</sub> jobs
- jobs that use the Jaguar batch facility

### 13.1 Installing Parallel Jaguar

Parallel Jaguar is currently available for the SGI, Linux, and IBM platforms. The parallel Jaguar executables are installed by default when you install Jaguar. After installation, the hosts on which you will run Jaguar may need to be configured for parallel execution. Installation and configuration instructions are given in the *Schrödinger Product Installation Guide*, and are repeated below.

In addition to host configuration, the file `$(SCHRODINGER)/schrodinger.hosts` must be edited to add entries for the parallel hosts. Each parallel host entry must include a `processors` line that indicates how many CPUs are available on that computer. This information is used in the GUI to display the maximum number of processors available for a host, and to check that this limit is not exceeded. For computer clusters that do not use

queuing software, an entry must be included for each node, and the value of `processors` for *each node* should be the total number of processors available in the cluster. For computer clusters that do use queuing software, host entries must be included for each queue that is to be used, and value of `processors` for each entry should be the total number of processors available in the cluster. See [Section 11.1 on page 263](#) for details of the format for the `schrodinger.hosts` file.

For all platforms, you should use local disks for scratch space. Performance is significantly reduced if an NFS-mounted scratch disk is used. Also, avoid using scratch directories which are actually symbolic links. Using symbolic links for scratch directories is known to prevent Jaguar jobs from running, especially under Linux. Thus, if `/scratch` is actually a symbolic link to `/scr`, specify `/scr` in the `schrodinger.hosts` file rather than `/scratch`.

### 13.1.1 SGI Installation

There are two system requirements for SGI: a version of Message-Passing Toolkit (MPT) no earlier than 1.2.1.1, and a version of Array Services no earlier than 3.1. If you are using the PBS batch queue system, you will need a version of MPT no earlier than 1.5.0.0. These packages must be installed by the system administrator for your computer because the installation requires root permission. Following are installation instructions:

1. Check to see if the required MPI (Message Passing Interface) files are already installed with the command

```
showprods | grep MPI
```

If MPI is not installed, you can install it from the MPT package, which can be downloaded from <http://www.sgi.com/products/evaluation/>

2. If necessary, install Array Services. You can check to see if Array Services are present with the command

```
showprods | grep arraysvcs
```

If you have Array Services 3.2, Patch 3532 is required. You can check if Patch 3532 is already installed with the command

```
showprods | grep 3532
```

If it is not, you can get it from <http://support.sgi.com/surfzone/patches/>

Array Services allows your SGI to run MPI applications like parallel Jaguar. Start the array services daemon with the following command:

```
/etc/init.d/array start
```

The `arrayd` daemon can be configured to start automatically at system startup with the command

```
chkconfig array on
```

Finally, note that a CERT advisory (<http://www.cert.org/advisories/CA-99-09-arrayd.html>) has been posted about the default Array Services installation on IRIX versions 6.2 – 6.5.4. A simple fix is available at

<ftp://patches.sgi.com/support/free/security/advisories/19990701-01-P>

## 13.1.2 LINUX Installation

For Linux, parallel Jaguar requires the MPICH package. Jaguar is now supported under Red Hat Linux 7.3, which is based on a Linux 2.4 kernel. Earlier versions might run but are not supported. If Jaguar is to run in parallel on a multiprocessor machine, the kernel must be compiled for SMP (symmetric multiprocessing).

### 13.1.2.1 Installing MPICH

We recommend building MPICH from the source code. The latest source code is always available from <http://www-unix.mcs.anl.gov/mpi/mpich>. Instructions for building and installing MPICH are included with the source code.

When you build MPICH from the source code, include the following configure options:

```
--with-comm=shared --with-device=ch_p4
```

The directory in which you installed MPICH is referred to below as *MPICH\_install*.

### 13.1.2.2 Configuration

1. Add the MPICH `bin` directory to the `PATH` environment variable. This is necessary for Jaguar to find the `mpirun` launch script.

**csh/tcsh:** `setenv PATH MPICH_install/bin:$PATH`

**sh/ksh/bash:** `export PATH=MPICH_install/bin:$PATH`

2. Edit the file `MPICH_install/share/machines.LINUX` and list the names of the hosts available for parallel calculations. Each line of this file should specify the name of a host and the number of processors on that host, separated by a colon. The host name should match the output of the `hostname` command. For example:

```
homer.mynet.edu:2
marge.mynet.edu:2
bart.mynet.edu:1
```

3. Edit the `schrodinger.hosts` file in the directory where Jaguar was installed, and list in it the names of the hosts in the `machines.LINUX` file. The host names in `schrodinger.hosts` need not include the domain name. See [Section 11.1 on page 263](#) for details on the format of the `schrodinger.hosts` file. For the above example, the `schrodinger.hosts` file would look like:

```
host:          homer
schrodinger:   /apps/Schrodinger
tmpdir:        /scr
processors:    2
!
host:          marge
schrodinger:   /apps/Schrodinger
tmpdir:        /scr
processors:    2
!
host:          bart
schrodinger:   /apps/Schrodinger
tmpdir:        /scr
processors:    1
```

4. Ensure that `rsh` is enabled. By default, Jaguar uses `rsh` to communicate with remote nodes (even if you are running on a stand-alone SMP workstation with 2 CPUs). To enable `rsh`, each user must create a file called `.rhosts` in his or her home directory. The `.rhosts` file should contain the name of each host listed in the file `machines.LINUX`, followed by the user's login name, e.g.,

```
homer.mynet.edu username
marge.mynet.edu username
bart.mynet.edu username
```

The `.rhosts` file must be owned by the user (not by root) and must not be writable by anyone except the user, or authentication fails. To ensure this, enter the command

```
chmod 644 ~/.rhosts
```

We strongly recommend that you test `rsh` connections by using the shell script `tstmachines`, which is in `MPICH_install/sbin`. This script attempts to run several `rsh` commands on each of the hosts listed in the file `machines.LINUX`, and lists any problems. If the command is successful it returns with no output. You can also use the `-v` option on the command line to see exactly what the script is doing.

**Note:** Because MPICH uses `rhosts` authentication, you must set up the `.rhosts` file even if you are using `ssh` for communication.

### 13.1.2.3 Launching the Secure Servers

Jaguar relies on the MPICH secure server, `serv_p4`, to transport the environment to all nodes used in a parallel calculation. The secure server must be running on all computers on which Jaguar is to run in parallel, which is normally all hosts listed in the `machines.LINUX` file. The secure server uses a communication port that is specified by the user (or by root).

To launch the MPICH secure server, enter the command

```
$SCHRODINGER/utilities/mpich start -p port
```

The port number (*port*) should be a four-digit number greater than 1023. If `-p port` is not specified, the value of `MPI_P4SSPORT` is used for the port number. If `MPI_P4SSPORT` is not set, the default value of 1234 is used. Although each user may launch the secure server and select a port number for private use, we recommend that the system administrator launch the server as root so that all users can use the same port number. The port number should be different from the default 1234, to avoid conflicts with other uses of the secure server ports. The `mpich start` command launches the secure servers on all of the hosts listed in the `machines.LINUX` file.

To use the secure servers, the following environment variables must be set:

```
cshtcsh:      setenv MPI_USEP4SSPORT yes
                setenv MPI_P4SSPORT port
sh/ksh/bash: export MPI_USEP4SSPORT=yes
                export MPI_P4SSPORT=port
```

The port number assigned to `MPI_P4SSPORT` must match the port number used to launch the secure server. These environment variables can be set up by root in the default environment, or they can be set up in a login script to avoid having to set them manually at each session. The last strategy does not work for `ksh`, which does not execute a login script.

The `mpich` script can be used to manage the secure servers. The syntax for the script is:

```
$SCHRODINGER/utilities/mpich command [options]
```

The available commands are listed in [Table 13.1](#), and the available options are listed in [Table 13.2](#). This script allows you to start and stop the servers on all or some of your machines and to check on their status through a single, consistent interface. The command acts on the hosts specified with the `-h` option, if any are given. Otherwise it acts on the hosts listed in the `machines` file. By default this file is the `machines.LINUX` file from your MPICH installation. You can override this default by specifying a file in the `SCHRODINGER_NODEFILE` environment variable, or by using the `-m` option. The port on which the servers listen can be specified using the `-p` option or the `MPI_P4SSPORT` environment variable, otherwise the standard port number 1234 is used.

Table 13.1. Commands for the `mpich` Script

Command	Action
<code>start</code>	Start servers
<code>stop</code>	Kill servers
<code>restart</code>	Kill and restart servers
<code>status</code>	Report server status
<code>pid</code>	Report server PID
<code>sems</code>	Report semaphore sets in use
<code>rmsems</code>	Delete all semaphore sets
<code>config</code>	Describe the MPICH configuration.

Table 13.2. Options for the `mpich` Script

Option	Meaning
<code>-p port</code>	Specify the port number for servers. The default port is 1234.
<code>-m hostfile</code>	Specify a file listing the MPICH host machines. The default file is <code>machines.LINUX</code> .
<code>-h host-list</code>	Act just on the specified hosts. The default is to act on all hosts specified in the host file.
<code>-u user</code>	Connect to remote machines as the specified user
<code>-v</code>	Report the version number of the <code>mpich</code> script.
<code>-d</code>	Provide debugging output.

The secure server can also be launched manually on each machine with the command:

```
/usr/lib/mpich/bin/serv_p4 -o -p port
```

You can also launch the secure servers within a job by setting the environment variable `SCHRODINGER_MPISTART` in the shell in which the job is launched, in addition to `MPI_P4SSPORT` and `MPI_USEP4SSPORT`. The value “yes” requests a single attempt to launch the secure servers. An integer value specifies the time limit in seconds for attempts to launch the servers, which are made every 10 seconds. By default, the secure servers are not launched within the job. The servers continue to run after the job is finished.



### 13.1.2.4 Selecting Nodes for a Job

Job queuing software such as PBS is often used on computer clusters to assign nodes and manage the load. If you are not using job queuing software, you can select the nodes that a job will run on in the following ways:

- Select the nodes using the `-HOST` option of the `jaguar run` command. The list of nodes must be enclosed in quotes.
- Create a local `machines.LINUX` file listing the nodes you want to use, and set the environment variable `SCHRODINGER_NODEFILE` to point to this file.
- Create a local `machines.LINUX` file listing the nodes you want to use, and set the `SCHRODINGER_MPI_FLAGS` environment variable to:

```
"-machinefile filename"
```

The file name must be the full path to the file. Use of the `-HOST` option overrides the use of the environment variables.

If the local host is in the list of available nodes, the controlling MPI process runs on the node from which you execute the `jaguar run` command; otherwise it runs on the first node in the node list.

### 13.1.2.5 Troubleshooting Parallel Job Problems

If the `jobname.log` file contains the error message

```
error while loading shared libraries: libhdf5.so: cannot open shared
object file: No such file or directory
```

then `SCHRODINGER` is not set on at least one of the nodes. The `serv_p4` process transports the environment variables, including `SCHRODINGER`, to all job nodes. If `serv_p4` is not running, or if it is using the wrong communication port (which must match your `MPI_P4SSPORT` value), or if `MPI_USEP4SSPORT` is not set (or is misspelled), then `SCHRODINGER` is not set correctly and this error message results.

If the `jobname.log` file contains the error message

```
p4_error: OOPS: semop lock failed: -1
```

then all available semaphore sets are taken on at least one of the nodes. This error can only occur when using shared memory. If you have dual-CPU machines and you compiled the Linux kernel for SMP, this error can occur if you did not build MPICH with the `--with-comm=shared` option. It can also occur if other users are running jobs that use shared memory, or if many previous Jaguar jobs failed in such a way that the semaphores could not be freed. To check for semaphores that are not freed, enter the command

```
$SCHRODINGER/utilities/mpich sems
```

If the output indicates that you are still using semaphores for jobs that are no longer running, enter the command

```
$SCHRODINGER/utilities/mpich rmsems
```

to free them. You should also check for running processes that are associated with failed Jaguar jobs with the command

```
$SCHRODINGER/jaguar jobs
```

You can kill all processes associated with a job whose status is "running" by entering the command

```
$SCHRODINGER/jaguar kill jobcontrol-id
```

The processes associated with any jobs listed as "stranded" must be killed manually.

If the *jobname.out* file contains an error message like the following:

```
p1_28583: (19.611297) xx_shmalloc: returning NULL; requested 3251504 bytes
p1_28583: (19.611409) p4_shmalloc returning NULL; request = 3251504 bytes
You can increase the amount of memory by setting the environment variable
P4_GLOBMEMSIZE (in bytes); the current size is 4194304
p1_28583: p4_error: alloc_p4_msg failed: 0
```

set the environment variable `P4_GLOBMEMSIZE` (whose default value under Linux is 4 MB) to a larger number, as the message suggests, even if the amount of shared memory requested is less than the current setting. The maximum value of the `P4_GLOBMEMSIZE` setting should be no larger than the maximum shared memory for the kernel. For RedHat Linux 7.3 this is usually 33554432 bytes. To see the maximum amount of shared memory your kernel can allocate, enter the command

```
cat /proc/sys/kernel/shmmax
```

### 13.1.3 IBM Installation

For IBM, you need to install the Parallel Operating Environment (POE) package, which includes the MPI libraries. Jaguar requires a version of POE no earlier than 2.4. Be sure to check the README file in `/usr/lpp/ppe.poe` and the man page for details on POE. If you use LoadLeveler, it must be a version that is no earlier than 2.1.

You may need to set an environment variable in order to use multiple processors for a job. The variable to set depends on how your machine has been configured; specifically whether you are running the Job Manager or not. The Job Manager manages pools of nodes, and assigns specific parallel jobs to specific nodes. To test whether you are using the Job Manager, type

```
ps aux | grep jmd
```

If you see `jmd` processes listed, you are running the Job Manager. In this case, you need to tell Job Manager the pool from which you want to have nodes assigned to you. The command `jm_status -P` lists the available pools and their member nodes. The environment variable that sets your job pool is called `MP_RMPOOL`, and it should be set to the appropriate pool number:

```
csh/tcsh:    setenv MP_RMPOOL 1
sh/ksh/bash: export MP_RMPOOL=1
```

If your machine does not use the Job Manager, you can set the environment variable that `MP_HOSTFILE` to the file that contains the host list. If `MP_HOSTFILE` is not set, then the hostfile is assumed to be called `host.list` and to reside in the current directory (see the `poe` man page). The host file should contain the names of the nodes on which parallel jobs can be run. The node name is listed once for each processor in that node. For example, if you have a workstation called “bobcat” with four processors and you want to be able to use all four processors, the host file should contain the following four lines:

```
bobcat
bobcat
bobcat
bobcat
```

If you call this host file `my.hostfile`, then you should set `MP_HOSTFILE` as follows:

```
csh/tcsh:    setenv MP_HOSTFILE /home/userid/my.hostfile
sh/ksh/bash: export MP_HOSTFILE=/home/userid/my.hostfile
```

If you are unsure of your system configuration, contact your system administrator for more information.

Ensure that the `schrodinger.hosts` file is properly configured for your cluster. See [Section 11.1.4 on page 266](#) for more information on this file.

Finally, you must also make sure you have `rsh` access to the host, even if you are on it. To do this, add a line to your `~/ .rhosts` file that specifies the node that you need access to, and your login name:

```
bobcat.schrodinger.com userid
```

This gives user *userid* `rsh` access to host `bobcat.schrodinger.com`.

## 13.2 Running Jobs in Parallel

To run Jaguar jobs in parallel, you need only specify the number of processors to use for the job at the time you launch it. You do not need to launch `mpirun` or `poe`: this is done automatically by Jaguar. If you launch the job from the command line, set the `-PROCS` option to the number of processors to be used. For example,

```
jaguar run -PROCS 8 -HOST mysmp jobname
```

If you launch the job from the GUI, type the number of processors to be used into the # of Processors: window in the Jaguar Run panel. By default, the maximum number of processors that you can request is shown in this window. The number is read from the `processors` line for the selected host in the `schrodinger.hosts` file.

If you need to pass additional parameters to the POE or MPI launch commands, you can set the environment variable `SCHRODINGER_POE_FLAGS` on IBM platforms or `SCHRODINGER_MPI_FLAGS` on all other platforms to the arguments that you want to pass in. For example, for verbose output from `mpirun` on an SGI, set the following:

```
cshtcsh:      setenv SCHRODINGER_MPI_FLAGS "-v"  
sh/ksh/bash: export SCHRODINGER_MPI_FLAGS="-v"
```

When a parallel job is run on an IBM host, the following POE flags are automatically set:

```
-euilib ip -shared_memory yes -wait_mode poll
```

When you are choosing the number of processors to use for a parallel Jaguar job, divide the number of basis functions for the job by 100 for HF or DFT jobs or 80 for LMP2 jobs, then discard any portion of this number after the decimal place. This number is the maximum number of processors advised for an efficient parallel run. For instance, if your molecule had 486 basis functions, the maximum number of processors advised for an HF or DFT calculation is 4, and the maximum number of processors for an LMP2 job is 6.

You can tell whether a job is running in parallel by looking at its log file (*jobname.log*). If the job is running in parallel, the third line of the log file will contain, for example,

```
Running on 2 processors
```

If there is no such line, the job is running in serial mode.

If you are using a queueing system for your parallel jobs, note that the number of processes created by Jaguar is the number of processors for the job plus one, because the Jaguar control program `jexec` always runs as a separate process.

Jaguar batch jobs cannot use MPI for the individual subjobs. If you request multiple processors for a batch job with multiple input structures or files, the subjobs are distributed over the available processors with one job per processor.

---

# Chapter 14: The pK<sub>a</sub> Prediction Module

---

## 14.1 Introduction

Schrödinger's pK<sub>a</sub> prediction module represents the first attempt to utilize ab initio quantum chemical methods to reliably predict pK<sub>a</sub> values in aqueous media.[144] The module employs 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 to the methodology 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.

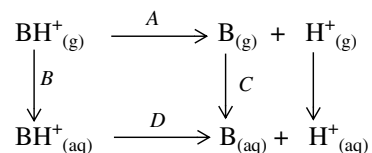
Several features of the method distinguish it from purely empirical fragment-based approaches, which are complementary to the present product. First, we expect that the use of ab initio quantum chemistry rather than fragment table lookups and interpolation will lead to a substantially wider range of 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 the MacroModel molecular modeling code. 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<sub>a</sub> 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.

## 14.2 Theory of pK<sub>a</sub> Calculation

### 14.2.1 Ab initio Quantum Chemical Calculation of pK<sub>a</sub> Values

The calculation of the pK<sub>a</sub> of a molecule in aqueous solution can be represented as a thermodynamic cycle:

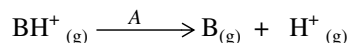


The strategy in our pK<sub>a</sub> module is to calculate parts *A*, *B*, and *C* of the above cycle, whereupon the actual pK<sub>a</sub>, which is related to *D* by

$$\text{pK}_a = \frac{1}{2.3RT} D$$

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:



The gas phase free energy difference between the protonated and deprotonated states can be computed via the usual relations

$$\begin{aligned}
 A &= \Delta H - T\Delta S \\
 &= E_{\text{B}(g)} - E_{\text{BH}^+(g)} + 5/2RT - T\Delta S
 \end{aligned}$$

Evaluation of this expression requires the following quantum chemical calculations:

1. *Geometry optimization* of the protonated and deprotonated species. Note that quantum chemical methods generally carry out a conjugate gradient optimization and hence cannot search for multiple minima. We shall assume in this discussion 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 those variants employing an admixture of 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; however, 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<sub>a</sub> calculations. This is because in determining the pK<sub>a</sub> 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<sub>a</sub>.

To overcome this problem, we have adopted a novel strategy, which is to fit the parameters for ions *directly to experimental pK<sub>a</sub> 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<sub>a</sub> 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<sub>a</sub> 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 14.1 on page 306](#). 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.

## 14.2.2 Empirical Corrections

The results of the above calculation can be assembled to yield a raw  $\text{pK}_a$  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  $\text{pK}_a$  values range from 0.65 to 9.17.

Our empirical corrections take the simple linear form:

$$\text{pK}_a = a \text{pK}_a(\text{raw}) + b$$

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  $\text{CO}_2^-$  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  $\text{pK}_a$  are well correlated with these charge shifts, linearly scaling the correction term to the raw  $\text{pK}_a$  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  $\text{pK}_a$  prediction. As the accuracy goal (0.5  $\text{pK}_a$  units) is beyond the capabilities of the raw DFT calculations, empirical corrections are necessary.

## 14.3 Predicting pKa Values in Complex Systems

The algorithm described in [Section 14.2 on page 300](#) 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.

### 14.3.1 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. In the current release, only method (1) below has been automated. It is still possible to carry out the strategies outlined in method (2), however at present the user must run multiple jobs and manually assemble the data into a final result.

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 respective class. This is probably a reasonable assumption for many problems. Note

that the conformation that is lowest in the protonated state may not be lowest in the deprotonated state. In many cases there are obvious electrostatic reasons why a conformational change upon protonation or deprotonation would occur. The program is set up to 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 MMFF 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. 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. A more accurate approach is to 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.

### 14.3.2 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 in our suite of test cases, which has three equivalent sites. In this situation, there is a statistical correction factor arising from increased entropy of the appropriate species. As we do not have an automated facility for recognizing equivalent sites in the current version of the program, the user must make this correction by hand to the result obtained from running the  $pK_a$  prediction module. The correction factor is  $\log(N^2)$ , where  $N$  is the number of equivalent sites, and the power of 2 comes from the fact that there are two particles involved:  $H^+$  and the species being protonated. For convenience, we supply here the correction factors for two and three equivalent sites (room temperature):

- 2 equivalent sites: bases +0.60, acids -0.60.
- 3 equivalent sites: bases +0.95, acids -0.95.

### 14.3.3 Multiple Protonation Sites

Many molecules have several sites, which can have different pK<sub>a</sub> values. Consider a case with two distinct possible protonation sites for which we want to calculate the pK<sub>a</sub> of site 1. Then the following situations are possible:

1. Then the following situations are possible:

1. The two pK<sub>a</sub> values are well separated and the pK<sub>a</sub> 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<sub>a</sub> calculation for site 1 is run with site 2 in the deprotonated state.
2. The two pK<sub>a</sub> values are well separated and the pK<sub>a</sub> 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<sub>a</sub> calculation for site 1 is run with site 2 in the protonated state.
3. The two pK<sub>a</sub> values are unknown, or the pK<sub>a</sub> 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.

Cases (1) and (2) are straightforward to handle using the current software. When the groups are close together, there are some scientific issues about the accuracy of results for multiply protonated states. This will become clearer as more test cases are run. In the next release, we intend to treat case (3) inside the program. With the present software, you must run two separate pK<sub>a</sub> jobs, each of which handles two of the four protonation states, and build the titration curve by hand.

## 14.4 Training Set Results

Table 14.1 presents a summary of the results for the functional groups that we have parameterized, including the number of cases studied, average deviation from experiment, and maximum deviation. A listing of the results for the individual test cases, comparing experimental and calculated pK<sub>a</sub> values can be found in Table 14.2.

The largest set 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 of 0.3 and maximum error of 0.9 that we observe.

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.

The largest error for alcohols comes from t-butanol, with the predicted pK<sub>a</sub> being too low (acidic) by about 3 pK<sub>a</sub> units. We believe the source of this error to come from the oversta-

bilization of the ionized form,  $(\text{CH}_3)_3\text{CO}^-$ . It is possible that the continuum solvation model does not fully account for the steric shielding from the three methyl groups on the negatively charged oxygen.

For functional groups where a relatively small number of compounds have been included in the parameterization, the results are obviously less reliable. We have nevertheless included some groups of this type (hydroxamic acids, sulfinic acids, sulfonic acids, thiophenol and imine) in the initial release. Feedback in these cases as to the validity of the parameterization would be particularly valuable to us in developing the next generation of parameters.

Table 14.1. Functional Groups for Which pKa Parameters Are Available

Functional Group	RMS Deviation	Maximum Absolute Deviation	.jres File Group Number
Alcohols	1.2	2.9	1
Phenols	0.3	0.6	2
Carboxylic acids	0.6	1.5	4
Thiols	0.2	0.4	10
Sulfonamides	0.6	1.5	15
Hydroxamic acids	0.6	1.5	6
Imides	0.9	1.4	13
Barbituric acids	0.4	0.6	12
Tetrazoles	0.7	1.4	17
Primary amines	0.5	0.9	31
Secondary amines	0.5	0.8	32
Tertiary amines	0.8	1.4	33
Anilines	0.3	0.6	25-30
Amidines	0.4	0.5	22
Heterocycles	0.3	0.5	19
Benzodiazepines	0.7	1.6	23
Guanidines	0.7	1.4	21
Pyrroles (C-2 protonation)	0.6	0.9	35
Indoles (C-3 protonation)	0.2	0.2	36
<b>Total RMS Deviation</b>	<b>0.7</b>		

Table 14.2. Molecules Used in the pKa Parameterization, Arranged by Functional Group

MOLECULE	pKa calc.	pKa exp.	Deviation
<b>ALCOHOLS</b>			
methanol	16.4	15.5	0.9
ethanol	16.0	15.9	0.1
propanol	16.0	16.2	-0.2
i-propanol	15.8	17.1	-1.3
2-butanol	16.8	17.6	-0.8
t-butanol	16.3	19.2	-2.9
allyl alcohol	15.3	15.5	-0.2
propargyl alcohol	13.4	14.3	-0.9
2-chloroethanol	15.0	14.4	0.6
2,2-dichloroethanol	13.9	12.2	1.7
2,2,2-trichloroethanol	12.6	12.4	0.2
2,2,2-trifluoroethanol	11.5	12.4	-0.9
1,2-ethanediol	15.7	14.0	1.7
1,2-propanediol	15.4	14.9	0.5
1,3-propanediol	16.4	15.1	1.3
1,4-butanediol	16.4	15.1	1.3
<b>PHENOLS</b>			
phenol	9.8	10.0	-0.2
4-aminophenol	10.6	10.2	0.4
4-chlorophenol	9.3	9.9	-0.6
4-fluorophenol	9.5	9.4	0.1
4-methoxyphenol	10.4	10.3	0.1
4-methylphenol	10.3	10.5	-0.2
4-nitrophenol	6.9	7.2	-0.3
p-xylol	10.4	10.3	0.1
4-hydroxybenzaldehyde	7.5	7.6	-0.1

Table 14.2. Molecules Used in the pKa Parameterization, Arranged by Functional Group

MOLECULE	pKa calc.	pKa exp.	Deviation
<b>CARBOXYLIC ACIDS</b>			
cis 1,2-cyclopropanedicarboxylic	4.4	3.6	0.8
trans 1,2-cyclopropanedicarboxylic	4.1	3.8	0.3
cis 2-chlorobut-2-enecarboxylic	3.6	2.8	0.8
trans 2-chlorobut-2-enecarboxylic	3.3	3.2	0.1
2-chlorobut-3-enecarboxylic	3.0	2.5	0.5
2-chloropropanecarboxylic	3.3	2.9	0.4
2,2-dimethylpropanoic	4.5	5.0	-0.5
2-furanecarboxylic	3.5	3.2	0.3
cis 2-methylcyclopropanecarboxylic	4.3	5.0	-0.7
trans 2-methylcyclopropanecarboxylic	4.5	5.0	-0.5
2-methylpropanecarboxylic	4.7	4.6	0.1
cis 3-chlorobut-2-enecarboxylic	3.9	4.1	-0.2
trans 3-chlorobut-2-enecarboxylic	3.6	3.9	-0.3
3-chloropropanecarboxylic	4.3	4.1	0.2
cis 3-chloropropenecarboxylic	4.0	3.5	0.5
trans 3-chloropropenecarboxylic	3.7	3.8	-0.1
3-chloropropynecarboxylic	2.7	1.9	0.8
3-nitro-2-propanecarboxylic	4.1	2.6	1.5
3-oxopropanecarboxylic	4.9	3.6	1.3
cis 4-chlorobut-3-enecarboxylic	4.3	4.1	0.2
trans 4-chlorobut-3-enecarboxylic	4.2	4.1	0.1
acetic acid	4.0	4.8	-0.8
acrylic acid	3.7	4.2	-0.5
benzoic acid	3.9	4.2	-0.3
butanoic acid	4.5	4.8	-0.3
trans cinnamic acid	4.3	4.4	-0.1
formic acid	3.2	3.8	-0.6
glycolic acid	3.3	3.8	-0.5

Table 14.2. Molecules Used in the pKa Parameterization, Arranged by Functional Group

MOLECULE	pKa calc.	pKa exp.	Deviation
glyoxylic acid	2.0	2.3	-0.3
malic acid	3.1	3.5	-0.4
malonic acid	3.6	2.9	0.7
oxalic acid	2.4	1.2	1.2
pentafluoropropanoic acid	0.8	-0.4	1.2
propanoic acid	4.4	4.9	-0.5
propargylic acid	2.3	1.9	0.4
succinic acid	4.3	4.2	0.1
dl tartaric acid	3.1	3.0	0.1
meso tartaric acid	2.6	3.2	-0.6
tartonic acid	2.7	2.4	0.3
trifluoroacetic acid	0.8	0.2	0.6
<b>THIOLS</b>			
methylthiol	9.9	10.3	-0.4
ethylthiol	10.7	10.6	0.1
2-mercaptoethanol	9.3	9.4	-0.1
1,2-ethanedithiol	9.1	9.1	0.0
<b>SULFONAMIDES</b>			
N-chlorotolylsulfonamide	4.6	4.5	0.1
dichlorphenamide	7.5	7.4	0.1
mafenide	9.3	8.5	0.8
methanesulfonamide	10.2	10.5	-0.3
quinethazone	9.3	9.3	0.0
saccharin	3.1	1.6	1.5
sulfamethizole	4.3	5.4	-1.1
sulfaperin	6.9	6.8	0.1
sulfacetamide	6.1	5.4	0.7

Table 14.2. Molecules Used in the pKa Parameterization, Arranged by Functional Group

MOLECULE	pKa calc.	pKa exp.	Deviation
sulfadiazine	6.4	6.5	-0.1
sulfadimethoxine	6.9	6.0	0.9
sulfamethazine	7.2	7.4	-0.2
sulfanylamide	10.9	10.4	0.5
sulfapyridine	7.7	8.4	-0.7
sulfaquinoxaline	6.0	5.5	0.5
sulthiame	10.3	10.0	0.3
<b>HYDROXAMIC ACIDS</b>			
formohydroxamic	8.1	8.7	-0.6
acetohydroxamic	8.4	8.7	-0.3
benzohydroxamic	8.4	8.8	-0.4
salicylhydroxamic	8.4	7.5	0.9
2-aminobenzohydroxamic	9.0	9.0	0.0
2-chlorobenzohydroxamic	8.3	7.8	0.5
2-fluorobenzohydroxamic	8.2	8.0	0.2
2-nitrobenzohydroxamic	8.5	7.0	1.5
3-nitrobenzohydroxamic	8.2	8.4	-0.2
4-aminobenzohydroxamic	8.9	9.4	-0.5
4-chlorobenzohydroxamic	8.4	8.7	-0.3
4-fluorobenzohydroxamic	8.4	8.8	-0.4
4-nitrobenzohydroxamic	8.2	8.3	-0.1
4-hydroxybenzohydroxamic	8.6	8.9	-0.3
<b>IMIDES</b>			
fluorouracil	8.6	8.0	0.6
methylthiouracil	7.7	8.2	-0.5
phenytoin	7.7	8.3	-0.6
3,3-methylphenylglutarimide	10.4	9.2	1.2



Table 14.2. Molecules Used in the pKa Parameterization, Arranged by Functional Group

MOLECULE	pKa calc.	pKa exp.	Deviation
3,3-dimethylsuccinimide	8.8	9.5	-0.7
dimethadione	7.5	6.1	1.4
phthalimide	8.5	9.9	-1.4
succinimide	9.1	9.6	-0.5
<b>BARBITURIC ACIDS</b>			
5,5-methylphenylbarbituric	7.5	7.4	0.1
1,5,5-trimethylbarbituric	7.8	8.3	-0.5
hexobarbital	7.6	8.2	-0.6
5,5-dimethylbarbituric	7.6	8.0	-0.4
1,5-dimethyl-5-phenylbarbituric	7.8	7.8	0.0
<b>TETRAZOLES</b>			
5-cyclopropyltetrazole	5.0	5.4	-0.4
5-methyltetrazole	5.0	5.6	-0.6
5-hydroxytetrazole	5.1	5.4	-0.3
5-phenoxytetrazole	4.7	4.4	0.3
5-phenyltetrazole	4.9	3.5	1.4
tetrazole	4.9	4.9	0.0
<b>PRIMARY AMINES</b>			
methylamine	10.5	10.2	0.3
ethylamine	10.9	10.6	0.3
propylamine	10.6	10.6	0.0
t-butylamine	10.8	10.7	0.1
2-aminoethanol	9.8	9.2	0.6
1,2-ethanediamine	10.1	10.7	-0.6
1,3-propanediamine	10.0	10.9	-0.9

Table 14.2. Molecules Used in the pKa Parameterization, Arranged by Functional Group

MOLECULE	pKa calc.	pKa exp.	Deviation
<b>SECONDARY AMINES</b>			
dimethylamine	10.9	10.7	0.2
diethylamine	11.0	11.0	0.0
azetidine	11.2	11.3	-0.1
pyrrolidine	11.1	11.3	-0.2
piperidine	11.0	11.1	-0.1
morpholine	9.3	8.5	0.8
2,5-diazahexane	9.3	10.4	-1.1
<b>TERTIARY AMINES</b>			
trimethylamine	10.0	9.8	0.2
triethylamine	10.5	11.0	-0.5
tripropylamine	9.3	10.7	-1.4
1-methylpiperidine	10.3	10.2	0.1
triallylamine	7.0	8.3	-1.3
1-allylpiperidine	9.8	9.7	0.1
dimethylcyclohexylamine	10.6	10.7	-0.1
dimethylbenzylamine	8.9	9.0	-0.1
diethylbenzylamine	9.2	9.5	-0.3
hexamethylenetetramine	6.3	5.3	1.0
DABCO	9.5	8.2	1.3
<b>ANILINES</b>			
aniline	4.6	4.6	0.0
4-chloroaniline	3.9	4.0	-0.1
4-methoxyaniline	5.4	5.2	0.2
4-nitroaniline	0.9	1.0	-0.1
p-toluidine	4.5	5.1	-0.6

Table 14.2. Molecules Used in the pKa Parameterization, Arranged by Functional Group

MOLECULE	pKa calc.	pKa exp.	Deviation
<b>AMIDINES</b>			
imidazo[2,3-b]thioxazole	8.2	8.0	0.2
tetrahydrozoline	10.0	10.5	-0.5
hydroxyimidazo[2,3-a]isoindole	9.1	8.6	0.5
tolazoline	10.6	10.3	0.3
<b>HETEROCYCLES</b>			
2-aminopyridine	7.2	6.7	0.5
2-aminothiazole	5.7	5.4	0.3
2-methylimidazole	8.0	8.0	0.0
3-aminopyridine	6.1	6.0	0.1
4-aminopyridine	9.7	9.7	0.0
4-methylpyridine	6.2	6.0	0.2
benzimidazole	5.3	5.8	-0.5
imidazole	6.9	7.0	-0.1
isoquinoline	5.5	5.4	0.1
melamine	5.0	5.0	0.0
pyrazine	1.2	0.7	0.5
pyrazole	2.5	2.5	0.0
pyridine	5.3	5.3	0.0
pyrimidine	1.1	1.3	-0.2
quinoline	5.0	4.8	0.2
thiazole	2.6	2.8	-0.2
<b>BENZODIAZEPINES</b>			
1,3-dihydro-1-methyl-5-phenyl-1,4-benzodiazepin-2-one	3.7	3.3	0.4
1,3-dihydro-3-hydroxy-5-phenyl-1,4-benzodiazepin-2-one	2.0	1.7	0.3

Table 14.2. Molecules Used in the pKa Parameterization, Arranged by Functional Group

MOLECULE	pKa calc.	pKa exp.	Deviation
1,3-dihydro-3-hydroxy-1-methyl-5-phenyl-1,4-benzodiazepin-2-one	1.4	1.6	-0.2
1,3-dihydro-5-phenyl-1,4-benzodiazepin-2-one	4.0	3.5	0.5
2,3-dihydro-1-methyl-5-phenyl-1,4-benzodiazepine	6.2	6.2	0.0
3-hydro-2-methylamine-4-oxy-5-phenyl-1,4-benzodiazepine	3.2	4.8	-1.6
<b>GUANIDINES</b>			
clonidine	7.8	8.1	-0.3
debrisoquin	12.0	11.9	0.1
guanidine	12.4	13.8	-1.4
methylguanidine	13.2	13.4	-0.2
<b>PYRROLES (C-2 protonation)</b>			
pyrrole	-2.9	-3.8	0.9
1-methylpyrrole	-2.2	-2.9	0.7
2-methylpyrrole	-0.7	-0.2	-0.5
3-methylpyrrole	-0.8	-1.0	0.2
<b>INDOLES (C-3 protonation)</b>			
indole	-3.7	-3.6	-0.1
1-methylindole	-2.1	-2.3	0.2
2-methylindole	-0.5	-0.3	-0.2
3-methylindole	-4.7	-4.6	-0.1

## 14.5 Running pK<sub>a</sub> Calculations

### 14.5.1 Activating the pKa Module

To run the pKa module, you need a special license in addition to the regular Jaguar license. To install the pKa module, first install Jaguar using the instructions in the *Schrödinger Product Installation Guide*. After you have successfully installed Jaguar, send in the machid information to obtain a license to activate the pK<sub>a</sub> module. Explicitly indicate in your license request that you want to run pK<sub>a</sub> calculations.

### 14.5.2 Jaguar Input Files for pK<sub>a</sub> Calculations

pK<sub>a</sub> calculations require input files in Jaguar format containing a molecular geometry and a labeled acidic site. The acidic site is either an acidic hydrogen in acids, or a heteroatom to be protonated in bases. If the starting geometry is not a neutral molecule but an ion, you have to specify its formal charge in the **atomic** section (see [Section 9.8 on page 218](#)). Also, if the geometry files are not in Jaguar format, you can translate them using the Read and Save windows (see [Section 3.4](#) and [Section 3.7](#)) or Babel (see [Section 11.2.5 on page 272](#) or type `jaguar babel` in a terminal window for usage instructions).

The acidic site can be marked using one of the following methods:

- By adding the suffix `_pk` to the atomic symbol
- By setting the `&gen` section keyword **ipkat** to either the atom's name, or to the atom's order number in the `&zmat` section. If you make this setting in the Edit Input panel in Maestro, you must save the input file to disk before starting the pK<sub>a</sub> job.

Here are three 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
H_pk 2.8258867600     -0.1221771000     -0.2269021000
H2  0.3281776900     -0.4358328800      1.0011835800
&
```

```
&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
&

```

### 14.5.3 Running pK<sub>a</sub> Calculations

To submit a pK<sub>a</sub> job using the GUI, follow the instructions for running batch jobs in [Section 3.6 on page 38](#) of this manual. Choose `pka.bat` as the batch file, and select your pK<sub>a</sub> Jaguar input files as the input files for the batch job.

To submit a pK<sub>a</sub> job for a single molecule using the command line, use the following command:

```
jaguar pka [-PROCS nproc] {jobname | 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 either the acid or the base. 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
-deprot basefile -prot acidfile

```

The input file name for the acid is `acidfile.in`; the input file name for the base is `basefile.in`. In this description, “acid” means either the acid or the protonated base, and “base” means the base or the deprotonated acid.

The number of processors used for parallel execution is `nproc`, and must be either 1 or 2. If you select two processors, the acid and base sections of the job are initiated as separate Jaguar jobs. Selecting more processors does not run the separate Jaguar jobs in parallel.

If you want to run more than one pK<sub>a</sub> job with a single command, you must use the jaguar batch command, and specify `pka.bat` as the batch file:

```
jaguar batch [options] pka.bat jobname1 [jobname2 ...]
```

The input files for the pK<sub>a</sub> jobs must be in the format described above. Use of the wildcard in job names is allowed. You cannot specify separate protonated and deprotonated species with the batch command. The command options are described in Table 11.3, Table 11.4, and Table 11.8.

### 14.5.4 Monitoring pK<sub>a</sub> Calculations

The pK<sub>a</sub> 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 14.3.

Table 14.3. 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

Final pK<sub>a</sub> and pK<sub>b</sub> values are calculated from data in these output files and written in *jobname.out* in the local directory (where *jobname.in* is a Jaguar input file submitted for a pK<sub>a</sub> calculation). For example, here is the final output file for formic acid:

```

Stoichiometry      Charge      pK
  CH2O2             0          pKa :  3.2
  CHO2              -1          pKb : 10.8
```

To list in a table all the pK<sub>a</sub> (and/or pK<sub>b</sub>) values, you can use the `jaguar results` command,

```
jaguar results -title -jobname -pka -pkb *.out
```

### 14.5.5 Initial Geometry

It is very important to choose the lowest energy conformer for the pK<sub>a</sub> calculations. As ab initio geometry optimizers only find the local minima, for flexible systems (long n-alkyl chains, many rotatable bonds) we strongly suggest first running a conformational search to determine the global minimum energy structure, which can then be used as an initial structure for the pK<sub>a</sub> run.

For certain systems it is expected that the protonated and deprotonated structure assume different conformations in their lowest energy state. For such cases, create Jaguar input files for both of them, and use the `-deprot` option of the `jaguar pka` command to specify the deprotonated structure. Output files and results are in the same format as in the single initial geometry runs.



---

## Chapter 15: Getting Help

---

For help installing and setting up licenses for Schrödinger software, see the *Schrödinger Product Installation Guide*.

The Maestro help facility consists of Auto-Help, Balloon Help (tooltips), and online help. To get help, follow the steps below.

- Check the Auto-Help window located below the title bar of the main window. If help is available for the task you are performing, it is automatically displayed there.
- If your question concerns an interface element, e.g., a button or option menu, there may be Balloon Help for the item. Move the mouse pointer over the element. If there is Balloon help for the element, it appears within a few seconds.
- If you do not find the help you need using the steps above, click the Help button in the panel for whose settings you are seeking help. The Help panel is opened and a relevant help topic is displayed.
- For help with a concept or action not associated with a panel, open the Help panel from the Help menu on the main menu bar or by using the key combination ALT+H.

If you do not find the information you need in the Maestro help system, check the following sources:

- The *Maestro User Manual* for questions about Maestro.
- The *NBO Manual* for information about NBO calculations.
- The *Maestro Release Notes*.
- The *Jaguar Release Notes*.
- The Frequently Asked Questions page, located at <http://www.schrodinger.com/Support/faq.html>

The manuals and the release notes are available in PDF format from the Schrödinger web site at <http://www.schrodinger.com/Support/pdf.html>. Information on additions and corrections to the manuals is also available from this web page.

If you have questions that are not answered from any of the above sources, contact Schrödinger using the information below.

Schrödinger

E-mail: [help@schrodinger.com](mailto:help@schrodinger.com)

USPS: 1500 SW First Ave. Suite 1180, Portland, OR 97201

Phone: (503) 299-1150

Fax: (503) 299-4532

WWW: <http://www.schrodinger.com>

FTP: <ftp://ftp.schrodinger.com>

Generally, e-mail correspondence is best because you can send machine output, if necessary. When sending e-mail messages, please include the following information, most of which can be obtained by entering `$SCHRODINGER/machid` at a command prompt:

- All relevant user input and machine output
- Jaguar purchaser (company, research institution, or individual)
- Primary Jaguar user
- Computer platform type
- Operating system with version number
- Jaguar version number
- Maestro version number
- mmshare version number

---

# 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, and copies of some of these references are available from Schrödinger upon request. 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 Wavefunctions 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. 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.
19. Vacek, G.; Perry, J. K.; Langlois, J. -M. *Chem. Phys. Lett.* **1999**, *310*, 189.
20. 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.
21. BIOGRAF manual.
22. MacroModel manual.
23. 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.
24. 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.)

25. Dunitz, 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.
26. Becke, A. D. *J. Chem. Phys.* **1993**, *98*, 1372.
27. Becke, A. D. *J. Chem. Phys.* **1993**, *98*, 5648.
28. Stephens, P. J.; Devlin, F. J.; Chabalowski, C. F.; Frisch, M. J. *J. Phys. Chem.* **1994**, *98*, 11623.
29. Slater, J. C. *Quantum Theory of Molecules and Solids, Vol. 4: The Self-Consistent Field for Molecules and Solids*. McGraw-Hill: New York, 1974.
30. 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.)
31. 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.
32. Becke, A. D. *Phys. Rev. A* **1988**, *38*, 3098.
33. 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.
34. Perdew, J. P.; Zunger, A. *Phys. Rev. B* **1981**, *23*, 5048.
35. Perdew, J. P. *Phys. Rev. B* **1986**, *33*, 8822; and Perdew, J. P. *Phys. Rev. B* (Erratum) **1986**, *34*, 7406.
36. Becke, A. D. *J. Chem. Phys.* **1997**, *107*, 8554.
37. Becke, A. D. *J. Chem. Phys.* **1998**, *109*, 2092.
38. Schmider, H. L.; Becke, A. *J. Chem. Phys.* **1998**, *109*, 8188; Schmider, H. L.; Becke, A. *J. Chem. Phys.* **1998**, *108*, 9624.
39. Hamprecht, F. A.; Cohen, A. J.; Tozer, D. J.; Handy, N. C. *J. Chem. Phys.* **1998**, *109*, 6264.
40. Boese, A. D.; Handy, N. C. *J. Chem. Phys.* **2001**, *114*, 5497.
41. Perdew, J. P.; Burke, K.; Ernzerhof, M. *Phys. Rev. Lett.* **1996**, *77*, 3865; *Phys. Rev. Lett.* (Erratum) **1997**, *78*, 1386.
42. Møller, C.; Plesset, M. S. *Phys. Rev.* **1934**, *46*, 618.

43. Sæbø, S.; Pulay, P. *Theor. Chim. Acta* **1986**, *69*, 357.
44. Sæbø, S.; Pulay, P. *Ann. Rev. Phys. Chem.* **1993**, *44*, 213.
45. Sæbø, S.; Tong, W.; Pulay, P. *J. Chem. Phys.* **1993**, *98*, 2170.
46. Foster, J. M.; Boys, S. F. *Rev. Mod. Phys.* **1960**, *32*, 300.
47. Pipek, J.; Mezey, P. G. *J. Chem. Phys.* **1989**, *90*, 4916.
48. Harding, L. B.; Goddard, W. A., III. *J. Am. Chem. Soc.* **1975**, *97*, 6293.
49. Carter, E. A.; Goddard, W. A., III. *J. Chem. Phys.* **1987**, *86*, 862.
50. Fischer, T. H.; Almlöf, J. *J. Phys. Chem.* **1992**, *96*, 9768.
51. Schlegel, H. B. *Theor. Chim. Acta* **1984**, *66*, 333.
52. *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.
53. Water's probe radius is set to 1.40 to reproduce solvation energies properly. All other probe radii are calculated from  $r^3 = \frac{3m\Delta}{4\pi\rho}$  ( $10^{24}$  Å<sup>3</sup>/cm<sup>3</sup>), where  $r$  is the solvent probe radius in Angstroms,  $m$  is the molecular mass obtained by dividing the molecular weight given in ref. [52] in grams per mole by  $6.02 \times 10^{23}$ ,  $\Delta$  is the packing density, and  $\rho$  is the density in g/cm<sup>3</sup> at 20 deg. C obtained from ref. [52]. Finding the actual  $\Delta$  would require a detailed knowledge of the structure of the liquid. Currently, all  $\Delta$  values for these liquids are assumed to be 0.5. (For FCC lattices,  $\Delta$  is 0.7405, and for BCC lattices,  $\Delta$  is 0.6802.)
54. Rappé, A. K.; Casewit, C. J.; Colwell, K. S.; Goddard, W. A.; Skiff, W. M. *J. Am. Chem. Soc.* **1992**, *114*, 10024.
55. 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.
56. Breneman, C. M.; Wiberg, K. B. *J. Comput. Chem.* **1990**, *11*, 361.
57. Mayo, S. L.; Olafson, B. D.; Goddard, W. A., III. *J. Phys. Chem.* **1990**, *94*, 8897.
58. Mulliken, R. S. *J. Chem. Phys.* **1955**, *23*, 1833.
59. Glendening, E. D.; Badenhop, J. K.; Reed, A. E.; Carpenter, J. E.; Bohmann, J. A.; Morales, C. M.; Weinhold, F. *NBO 5.0*; Theoretical Chemistry Institute: University of Wisconsin, Madison, WI, 2001.
60. Baker, J.; Jarzecki, A. A.; Pulay, P. *J. Phys. Chem A* **1998**, *102*, 1412.
61. Scott, A. P.; Radom, L. *J. Phys. Chem.* **1996**, *100*, 16502.
62. Hehre, W. J.; Stewart, R. F.; Pople, J. A. *J. Chem. Phys.* **1969**, *51*, 2657.

63. Hehre, W. J.; Ditchfield, R.; Stewart, R. F.; Pople, J. A. *J. Chem. Phys.* **1970**, *52*, 2769.
64. Pietro, W. J.; Levi, B. A.; Hehre, W. J.; Stewart, R. F. *Inorg. Chem.* **1980**, *19*, 2225.
65. Pietro, W. J.; Blurock, E. S.; Hout, R. F., Jr.; Hehre, W. J.; DeFrees, D. J.; Stewart, R. F. *Inorg. Chem.* **1980**, *20*, 3650.
66. Collins, J. B.; Schleyer, P. von R.; Binkley, J. S.; Pople, J. A. *J. Chem. Phys.* **1976**, *64*, 5142.
67. Binkley, J. S.; Pople, J. A.; Hehre, W. J. *J. Am. Chem. Soc.* **1980**, *102*, 939.
68. Gordon, M. S.; Binkley, J. S.; Pople, J. A.; Pietro, W. J.; Hehre, W. J. *J. Am. Chem. Soc.* **1982**, *104*, 2797.
69. 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.
70. Pulay, P.; Fogarasi, G.; Pang, F.; Boggs, J. E. *J. Am. Chem. Soc.* **1979**, *101*, 2550.
71. Dill, J. D.; Pople, J. A. *J. Chem. Phys.* **1975**, *62*, 2921.
72. Ditchfield, R.; Hehre, W. J.; Pople, J. A. *J. Chem. Phys.* **1971**, *54*, 724.
73. Hehre, W. J.; Pople, J. A. *J. Chem. Phys.* **1972**, *56*, 4233.
74. Binkley, J. S.; Pople, J. A. *J. Chem. Phys.* **1977**, *66*, 879.
75. Hariharan, P. C.; Pople, J. A. *Theor. Chim. Acta* **1973**, *28*, 213.
76. Hehre, W. J.; Ditchfield, R.; Pople, J. A. *J. Chem. Phys.* **1972**, *56*, 2257.
77. 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.
78. Rassolov, V. A.; Pople, J. A.; Ratner, M. A.; Windus, T. L. *J. Chem. Phys.* **1998**, *109*, 1223.
79. Clark, T.; Chandrasekhar, J.; Spitznagel, G. W.; Schleyer, P. von R. *J. Comput. Chem.* **1983**, *4*, 294.
80. Frisch, M. J.; Pople, J. A.; Binkley, J. S. *J. Chem. Phys.* **1984**, *80*, 3265.
81. Krishnan, R.; Binkley, J. S.; Seeger, R.; Pople, J. A. *J. Chem. Phys.* **1980**, *72*, 650.
82. McLean, A. D.; Chandler, G. S. *J. Chem. Phys.* **1980**, *72*, 5639.
83. 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.

- 
84. Rappé, A. K.; Goddard, W. A. Unpublished work.
  85. Dunning, T. H., Jr., *J. Chem. Phys.* **1989**, *90*, 1007.
  86. Kendall, R. A.; Dunning, T. H., Jr.; Harrison, R. J. *J. Chem. Phys.* **1992**, *96*, 6796.
  87. Woon, D. E.; Dunning, T. H., Jr. *J. Chem. Phys.* **1993**, *98*, 1358.
  88. Woon, D. E.; Dunning, T. H., Jr. *J. Chem. Phys.* **1994**, *100*, 2975.
  89. Easton, R. E.; Giesen, D. J.; Welch, A.; Cramer, C. J.; Truhlar, D. G. *Theor. Chim. Acta* **1996**, *93*, 281.
  90. Schafer, A.; Huber, C.; Ahlrichs, R. *J. Chem. Phys.* **1994**, *100*, 5829.
  91. Hay, P. J.; Wadt, W. R. *J. Chem. Phys.* **1985**, *82*, 270.
  92. Hay, P. J.; Wadt, W. R. *J. Chem. Phys.* **1985**, *82*, 284.
  93. Hay, P. J.; Wadt, W. R. *J. Chem. Phys.* **1985**, *82*, 299.
  94. The LACV3P basis set is a triple-zeta contraction of the LACVP basis set developed and tested at Schrödinger, Inc.
  95. Cundari, T. R.; Stevens, W. J. *J. Chem. Phys.* **1993**, *98*, 5555.
  96. Hurley, M.; Pacios, L. F.; Christiansen, P. A.; Ross, R. B.; Ermler, W. C. *J. Chem. Phys.* **1986**, *84*, 6840.
  97. Lajohn, L.; Christiansen, P. A.; Ross, R. B.; Atashroo, T.; Ermler, W. C. *J. Chem. Phys.* **1987**, *87*, 2812.
  98. Ross, R. B.; Powers, J. M.; Atashroo, T.; Ermler, W. C.; Lajohn, L.; Christiansen, P. A. *J. Chem. Phys.* **1990**, *93*, 6654.
  99. Ross, R. B.; Gayen, S.; Ermler, W. C. *J. Chem. Phys.* **1994**, *100*, 8145.
  100. Ermler, W. C.; Ross, R. B.; Christiansen, P. A. *Int. J. Quantum Chem.* **1991**, *40*, 829.
  101. Nash, C. S.; Bursten, B. E.; Ermler, W. C. *J. Chem. Phys.* **1997**, *106*, 5133.
  102. Wildman, S. A.; DiLabio, G. A.; Christiansen, P. A. *J. Chem. Phys.* **1997**, *107*, 9975.
  103. Diffuse and polarization functions for Ga-Rn taken from Dylla, K. G. *Theor. Chem. Acta* **1998**, *99*, 366; diffuse functions for rare gases extrapolated from those for the other elements in the row.
  104. 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.



105. Obara, S.; Saika, A. *J. Chem. Phys.* **1986**, *84*, 3963.
106. 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.
107. Murphy, R. B.; Messmer, R. P. *J. Chem. Phys.* **1993**, *98*, 10102.
108. For information on Molden, see the Molden web site <http://www.caos.kun.nl/~schaft/molden/molden.html>.
109. Stewart, J. J. P. *MOPAC 6*; QCPE #455.
110. Hohenberg, P.; Kohn, W. *Phys. Rev. B* **1964**, *136*, 864.
111. Kohn, W.; Sham, L. J. *Phys. Rev. A* **1965**, *140*, 1133.
112. Parr, R. G.; Yang, W. *Density-Functional Theory of Atoms and Molecules*; Oxford: New York, 1989.
113. *Density Functional Methods in Chemistry*; Labanowski, J. K., Andzelm, J. W., Eds.; Springer-Verlag: Berlin, 1991.
114. Colle, R.; Salvetti, O. *J. Chem. Phys.* **1990**, *93*, 534.
115. Kraka, E. *Chem. Phys.* **1992**, *161*, 149.
116. Audi, G.; Wapstra, A. H. *Nuclear Phys.* **1995**, *A595 4*, 409.
117. Császár, P.; Pulay, P. *J. Mol. Struct.* **1984**, *114*, 31.
118. Schlegel, H. B. *J. Comput. Chem.* **1982**, *3*, 214.
119. Powell, M. J. D. *Math. Prog.* **1971**, *1*, 26.
120. Bofill, J. M. *J. Comp. Chem.* **1994**, *15*, 1.
121. Murtagh, B. A.; Sargent, R. W. H. *Comp. J.* **1970**, *13*, 185.
122. Fletcher, R. In *Practical Methods of Optimization*; Wiley: New York, 1987.
123. Banerjee, A.; Adams, N.; Simons, J.; Shepard, R. *J. Phys. Chem.* **1985**, *89*, 52.
124. Culot, P.; Dive, G.; Nguyen, V. H.; Ghuysen, J. M. *Theor. Chim. Acta* **1992**, *82*, 189.
125. Simons, J.; Jorgensen, P.; Taylor, H.; Ozment, J. *J. Phys. Chem.* **1983**, *87*, 2745.
126. 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.
127. Rabuck, A. D.; Scuseria, G. E. *J. Chem. Phys.* **1999**, *110*, 695.

- 
128. Bayly, C. I.; Cieplak, P.; Cornell, W. D.; Kollman, P. A. *J. Phys. Chem.* **1993**, *97*, 10269.
  129. Stroud, A. H. *Approximate Calculation of Multiple Integrals*; Prentice-Hall: New York, 1971.
  130. Lebedev, V. I. *Zh. vychisl. Mat. mat. Fiz.* **1975**, *15*, 48-54.
  131. Lebedev, V. I. *Zh. vychisl. Mat. mat. Fiz.* **1976**, *16*, 293-306.
  132. Lebedev, V. I. *Sibirsk. Mat. Zh.* **1977**, *18*, 132-142.
  133. 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.
  134. Cramer, C. J.; Truhlar, D. G. *J. Comp. Aided Mol. Design* **1992**, *6*, 629.
  135. 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.
  136. Chasman, D.; Beachy, M. D.; Wang, L.; Friesner, R. A. Parallel Pseudospectral Electronic Structure. I. Hartree-Fock Calculations. *J. Comput. Chem.* **1998**, *19*, 1017.
  137. 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.
  138. Jang, Y. H.; Sowers, L. C.; Cagin, T.; Goddard, W. A., III. *J. Phys. Chem. A* **2001**, *105*, 274.
  139. Langlois, J. -M. Ph.D. Dissertation, California Institute of Technology, Pasadena, CA, 1994.
  140. Perez-Jorda, J. M.; Becke, A. D.; San-Fabian, E. *J. Chem. Phys.* **1994**, *100*, 6520.
  141. Baker, J.; Andzelm, J.; Scheiner, A.; Delley, B. *J. Chem. Phys.* **1994**, *101*, 8894.
  142. Mura, M. E.; Knowles, P. J. *J. Chem. Phys.* **1996**, *104*, 9848.
  143. Gonzalez, C.; Schlegel, H. B. *J. Chem. Phys.* **1989**, *90*, 2154; *J. Chem. Phys.* **1990**, *94*, 5523.
  144. Klicic, J. J.; Friesner, R. A.; Liu, S.-Y.; Guida, W. C. *J. Phys. Chem. A* **2002**, *106*, 1327.

---

# Index

---

## A

About button ..... 47, 263  
accuracy level..... 78, 84  
    input keyword for..... 195  
accurate energies ..... 81  
acidic site, designating for  $pK_a$   
    calculations..... 315  
AIMPAC .wfn file, input keyword for ..... 207  
all-analytic calculation ..... 78  
    input keyword for..... 195  
    output from..... 123  
analytic corrections ..... 78  
    keyword for..... 195  
    theory ..... 149  
analytic frequencies ..... 65–66  
analytic gradient of energy..... 83, 84  
    convergence criteria ..... 84  
    in output file ..... 111  
    input keywords for ..... 180, 184  
angles—see bond angles  
atom labels  
    bond lengths and angles in output..... 125  
    canonical orbital space..... 135, 209  
    format..... 28  
    GVB pairs ..... 57  
    in orbital output..... 209  
    in output ..... 102  
    in zmat section ..... 164  
    in Z-matrix input ..... 29  
    LMP2 pairs..... 56  
    Mulliken population output..... 120  
    orbitals in output ..... 134  
atom numbers  
    for GVB pairs..... 57  
    for LMP2 pairs ..... 56  
atom selection ..... 10  
Atom Selection dialog box..... 11  
atomic charges  
    from ESP fit ..... 60  
    Mulliken..... 64  
atomic charges, formal..... 220  
atomic charges—see electrostatic potential  
    fitting, Mulliken population analysis,  
    molecular charge

atomic masses  
    for frequency calculations..... 66  
    input keyword for..... 169  
    setting in atomic section..... 218–220  
atomic orbital space, output in ..... 132–133  
atomic properties, setting in atomic  
    section ..... 218–226  
atomic section ..... 218–226  
atomic units..... 164, 165  
    in output ..... 125  
    keyword for geometry input..... 168, 229  
.atomig file  
    default ..... 237  
    description and format ..... 242–243  
    specifying in input file ..... 161  
Auto-Help ..... 20, 319

## B

babel  
    using to convert file formats..... 201–204  
    using to read input files ..... 34  
Balloon Help ..... 20, 319  
basgss basis set label..... 227, 228  
.basis file  
    description and format ..... 237–241  
    specifying in input file ..... 161  
basis functions  
    contracted ..... 126, 127  
    derivatives of, list in output..... 205  
    file containing..... 237–241  
    for individual atoms ..... 221  
    in counterpoise calculations..... 32  
    keyword for printing ..... 204  
    listing in output ..... 121  
    Mulliken populations for..... 120  
    number of ..... 102, 243  
    output ..... 126–129  
    type, as listed in output ..... 120, 134, 209  
    uncontracted..... 126, 127  
basis input type ..... 221  
basis set ..... 70–74  
    conversion to Jaguar format ..... 241  
    diffuse functions..... 70–71, 193, 238  
    file containing..... 237–241  
    for individual atoms ..... 221

- for initial guess..... 76, 227, 242
  - in generated GAUSSIAN 9x input file.. 144
  - keyword for list in output..... 204
  - keywords ..... 193
  - listed in output ..... 102
  - minimal, with GVB..... 57
  - polarization functions..... 70, 74, 193, 238
  - specifying for GAUSSIAN 92
    - input ..... 145, 206
    - with ECP, in output ..... 123
  - basis set superposition error..... 32, 54, 55, 156
  - Basis Set window ..... 70–74
  - batch input file
    - example ..... 279
    - format..... 276–278
  - batch jobs
    - jaguar batch command for ..... 275–280
    - remote ..... 280
    - running from Maestro ..... 40–44
  - BFGS method for Hessian updating, input
    - keyword for ..... 182
  - BIOGRAF
    - .bgf and .hes files, reading ..... 286
    - .hes files, format..... 226
    - reading files from Maestro ..... 34–35
  - bohr units for geometry
    - input ..... 164, 165, 168, 229
  - bond angles
    - freezing all ..... 86
    - freezing for geometry
      - optimization ..... 32, 86–87, 166
    - in Z-matrix ..... 30
    - output keyword ..... 205
  - bond dissociation ..... 58, 141
    - assignment of GVB pairs for ..... 141
    - preferred level of treatment of ..... 58
  - bond lengths
    - freezing all ..... 86
    - freezing for geometry
      - optimization ..... 32, 86–87, 166
    - in Z-matrix ..... 29
    - output keyword ..... 205
  - bonding types, describing in lewis files .... 255
  - Boys localization..... 79
    - input keywords for ..... 173, 200
    - orbital printing ..... 123, 133, 200, 209
    - output from..... 123
  - Build panel..... 9
- C**
- calculation host ..... 269, 285–286
    - definition ..... 265
    - entries in schrodinger .hosts... 265
    - selecting ..... 39, 270
  - canonical orbital space,
    - output in ..... 132–135, 208–209
  - Cartesian coordinates
    - format for geometry ..... 27, 28–29, 164
    - freezing for geometry
      - optimization ..... 29, 86–87
  - ch program ..... 232
    - output from..... 112, 113, 116–121
  - charge fitting—see ESP fitting, Mulliken
    - population analysis
  - charge, atomic
    - from ESP fit ..... 60
    - keyword for formal ..... 220
  - charge, molecular
    - keywords ..... 169
    - setting in Maestro..... 33
  - charges, atomic
    - Mulliken..... 64
  - chdens electron density output
    - file..... 63, 119–120, 189
  - CIS calculations ..... 75, 179
  - cis program..... 232
  - Cleanup button..... 6
  - Command Input Area..... 6
  - command options, jaguar run..... 270
  - comment line
    - batch script..... 276
    - input file ..... 40, 45, 162
  - configuration interaction (CI)
    - CI singles calculations ..... 75, 179
    - energy lowering for GVB pair natural
      - orbitals..... 108
    - GBV-RCI theory ..... 153
  - connect input file section ..... 166
  - connectivity
    - keyword for bonding ..... 169
    - output keyword ..... 205
    - output option ..... 125–126
  - consecutive Jaguar jobs
    - running from Maestro ..... 40–44
    - running with jaguar batch ..... 275–280

- constraints for geometry  
 optimization ..... 29, 32, 86–87, 166  
 keywords for ..... 181  
 constraints, dynamic ..... 167  
 convergence criteria  
 geometry optimization ... 84–85, 111, 183  
 SCF energy ..... 77, 142  
 SCF energy, keyword for ..... 193–194  
 solvation energy, keyword for ..... 188  
 convergence problems, troubleshooting ..... 287  
 convergence schemes ..... 77  
 DIIS ..... 77, 123  
 keywords for ..... 194  
 OCBSE ..... 77, 123  
 coord input file section ..... 166  
 coordinates  
 Cartesian, in geometry input ..... 28  
 constraining and freezing ..... 86  
 for refinement of Hessian ..... 32  
 Coulomb corrections ..... 254  
 Coulomb field, charge fitting to ..... 117  
 Coulomb operator (J) ..... 195  
 contributions to energy ..... 131  
 keyword for output ..... 208  
 obtaining i/o information for ..... 125  
 pseudospectral assembly of ..... 148–149  
 counterpoise calculations ..... 32, 165  
 defining fragments for ..... 225  
 specifying atoms for ..... 221  
 coupled perturbed Hartree-Fock (CPHF) terms  
 for LMP2 dipole moments ..... 54, 62  
 for LMP2 ESP fitted charges ..... 54, 61  
 coupled perturbed Hartree-Fock—see CPHF  
 calculations  
 covalent radii ..... 221  
 CPHF calculations  
 of hyperpolarizability ..... 63, 189  
 of polarizability ..... 63, 189  
 cpolar program ..... 232  
 cpu time ..... 204  
 Culot-Fletcher method for trust radius  
 adjustment, keyword for ..... 183  
 current working directory  
 default ..... 6  
 setting in Maestro ..... 6  
 .cutoff file ..... 252–253  
 default ..... 237  
 description and format ..... 252  
 specifying in input file ..... 161  
 cutoff methods ..... 78, 84, 195  
 cutoffs ..... 195, 196, 252–253  
 shown in output ..... 105, 137
- ## D
- .daf file  
 default ..... 237  
 description and format ..... 243–248  
 neighbor ranges ..... 243–244  
 specifying in input file ..... 161  
 dealiasing functions ..... 71, 243  
 choices for calculation ..... 244  
 contracted ..... 244, 245–246  
 keyword for list in output ..... 204  
 keywords ..... 210–211  
 long-range ..... 243, 245, 246  
 neighbor ranges for ..... 125, 243–245  
 ordering of sets ..... 245  
 output of number used ..... 204  
 short-range ..... 243–245, 246  
 uncontracted ..... 244, 245  
 default settings, reverting to ..... 46  
 default .atomig file ..... 242  
 delocalization of LMP2 pairs ..... 172, 217  
 density difference matrix  
 keyword for output of ..... 208  
 Per Iteration output option ..... 132  
 RMS of elements in output ..... 105, 137  
 density functional theory  
 (DFT) ..... 49–54, 159–160  
 customized functional for ..... 53–54  
 hybrid methods ..... 51–53  
 keywords for ..... 173–178, 210  
 method options ..... 51  
 optimization output ..... 109–112  
 output from ..... 106  
 standard functionals for ..... 51–53  
 density matrix  
 convergence criterion ..... 77, 84  
 convergence criterion, keyword  
 for ..... 193–194  
 in DIIS error vector ..... 105, 137  
 keyword for output ..... 208  
 density, plotting with plot section ..... 234–236  
 der1a program ..... 232  
 output from ..... 109–112

- 
- der1b program ..... 232
    - output from ..... 109–112
  - derivatives of basis functions ..... 129
    - keyword for list in output ..... 205
  - DFT window ..... 49–54
  - dftname values
    - name strings for construction of ..... 175
    - standard functional names ..... 174
  - DFT—see density functional theory
  - dielectric constant
    - keywords for ..... 188
    - setting in the Solvation window ..... 59
  - dielectric continuum method—see pbf program
  - DIIS convergence scheme ..... 77, 105, 137
    - keyword for coefficient output ..... 208
    - keywords for ..... 194
    - output ..... 105
  - dimming of menu items ..... 46
  - dipole moment—see multipole moments
  - Direct Inversion in the Iterative Subspace method—see DIIS convergence scheme
  - directory
    - changing in Maestro ..... 6
    - current working ..... 6
    - file output ..... 18
  - displacement
    - convergence criteria based on ..... 84
    - keywords for convergence criteria ..... 184
  - DISPLAY environment variable ..... 5, 283
  - display host ..... 281, 283–284
  - driver for geometry scan ..... 93–94
  - dsolv program ..... 233
  - dummy atoms
    - in the Hessian ..... 226
    - in Z-matrix input ..... 31
- E**
- echo input file section ..... 231
  - ECPs—see effective core potentials
  - Edit Geometry window ..... 57
  - Edit window ..... 26–28
    - fixing bond lengths or angles ..... 32
    - fixing Cartesian coordinates ..... 29
    - freezing bond lengths or angles ..... 86–87
    - freezing Cartesian coordinates ..... 86–87
  - editing geometries from Maestro ..... 26–28
  - editing input files from Maestro ..... 46–47
  - effective core potentials (ECPs)
    - basis sets for ..... 72–74
    - in atomic guess file ..... 243
    - in basis set file ..... 239–241
  - efields input file section ..... 229
  - eigenvector following in transition state
    - optimizations
      - keywords for ..... 182–183
      - option setting ..... 91
      - use in Hessian refinement ..... 92
  - elden program ..... 232
  - electric field for polarizability calculations
    - input file section for ..... 229
    - keyword for ..... 191
  - electron count, information in output ..... 104
  - electron density
    - evaluating on a grid ..... 63
    - keywords for ..... 189, 191, 211
    - output from calculation ..... 119
  - electrostatic potential
    - in .resp file ..... 207
    - output on a grid ..... 191
  - electrostatic potential fitting ..... 60–62
    - constraining to reproduce multipole moments ..... 61, 142
    - for LMP2 wavefunctions ..... 61
    - for solvation calculations ..... 58, 112, 113
    - grid for ..... 62
    - keywords for control of ..... 188
    - keywords for grid ..... 191, 212
    - output from ..... 117–118
    - recalculating multipole moments from ..... 62
    - RMS error in output ..... 117
    - setting options for ..... 60
  - energy components, keyword for output of ..... 208
  - energy convergence criterion ..... 77
    - keyword for ..... 193
  - energy difference
    - as geometry convergence criterion ..... 84, 111
    - keyword for geometry convergence criterion ..... 184
  - energy output
    - final GVB, components ..... 107
    - final SCF, components ..... 105
    - SCF components for each iteration... 131, 208
    - solvation ..... 113–116
    - total SCF, for each iteration ..... 105, 137

- two-electron contributions when  
 OCBSE selected..... 123
- enthalpy calculations—see thermochemical  
 properties
- entropy calculations—see thermochemical  
 properties
- environment variables  
 DISPLAY..... 5, 283  
 JAGUAR\_SCRATCH..... 266  
 JAGUAR\_SCRIPTS..... 41  
 MP\_HOSTFILE..... 297  
 MP\_RMPOOL..... 297  
 MPI\_P4SSPORT..... 293–295  
 MPI\_USEP4SSPORT..... 293–295  
 P4\_GLOBMEMSIZE..... 296  
 PATH..... 282, 291  
 SCHRODINGER..... 5, 282  
 SCHRODINGER\_MPI\_FLAGS..... 295, 298  
 SCHRODINGER\_MPISTART..... 294  
 SCHRODINGER\_NODEFILE.... 293, 295  
 SCHRODINGER\_POE\_FLAGS..... 298  
 SCHRODINGER\_TMPDIR..... 266
- ESP—see electrostatic potential
- exchange corrections, in pseudospectral  
 calculations..... 254
- exchange operator (K)  
 contributions to energy..... 131  
 keyword for per iteration output..... 208  
 keywords for calculation of..... 195  
 obtaining i/o information for..... 125  
 pseudospectral assembly of..... 148–149
- excited state calculations using CIS  
 keywords for..... 179  
 setting options for..... 75
- executable directory..... 102, 269  
 selecting with `jaguar run`..... 271
- execution path..... 231–234
- F**
- field, electric, input file section for..... 229
- file i/o directory..... 18
- file names..... 39, 45
- file output keywords..... 206
- file output options..... 129–131  
 GAMESS input file..... 207  
 GAUSSIAN 92 basis set (.gbs) file.... 145  
 GAUSSIAN 92 input (.g92) file..... 144  
 keywords for..... 206  
 XYZ file..... 130
- first shell correction factor for solvation,  
 keyword for..... 187
- Fock matrix  
 in DIIS error vector..... 105, 137  
 keyword for updating..... 195  
 keywords for output of..... 205, 208  
 new estimate from DIIS  
 scheme..... 105, 132, 137  
 Per Iteration output options..... 132  
 pseudospectral assembly of..... 147–149  
 updating..... 104, 137
- forces  
 analytic, availability..... 83  
 calculating only..... 83  
 keywords for..... 180, 184  
 numerical, availability..... 83  
 numerical, keywords for..... 184
- formal charge..... 220
- fragments  
 defining..... 225  
 frequencies for..... 193, 226
- freq program..... 232
- frequencies..... 65–66  
 fragment..... 193, 226  
 keywords for..... 184, 192  
 scaling..... 66–67  
 visualizing in Maestro..... 67  
 visualizing with Molden..... 66
- Frequencies window..... 64–69  
 output from..... 121–123
- frequency-related properties  
 keywords for..... 191  
 output..... 121–123  
 settings for..... 69
- functionals—see density functional theory  
 (DFT)
- G**
- GAMESS input files  
 keyword for..... 207  
 option for..... 130
- gas phase optimizations, option to skip..... 60
- GAUSSIAN 92  
 Hessian format..... 226  
 orbital output in format for..... 135, 209
- GAUSSIAN 92 basis set file (.gbs)  
 keyword for generation of..... 145, 207  
 option for generation of..... 130

- GAUSSIAN 92 input file (.g92)  
keyword for generation of ..... 144, 207  
option for generation of ..... 130, 143–145  
reading ..... 34–35, 145, 286
- Gaussian function list in output, keywords  
for ..... 204
- gen input file section ..... 168
- generalized gradient approximation ..... 159
- Generalized Valence Bond method—see GVB  
calculations
- geometry input ..... 26–35  
Cartesian format ..... 27, 28–29, 164  
editing ..... 26–28  
file types for scanning ..... 34–35  
format ..... 164–168  
input file sections for ..... 164–168  
keywords ..... 168  
output file, echoed in ..... 102  
sample calculation ..... 24  
symmetrizing ..... 37–38, 286  
troubleshooting ..... 286  
units input keyword ..... 165  
units, keyword for ..... 164, 168, 229  
Z-matrix format ..... 27, 29–32, 164–166
- geometry optimization ..... 83–94  
calculating forces only ..... 83, 180  
constraining bond lengths or  
angles ..... 32, 86–87, 166, 181  
constraining Cartesian  
coordinates ..... 29, 86–87  
convergence criteria . 84, 84–85, 111, 183  
detailed output, option for ..... 126  
frozen bond lengths or  
angles for ..... 32, 86–87, 166, 181  
frozen Cartesian coordinates for 29, 86–87  
GDIIS method ..... 181  
generating input with new geometry.. 141  
in solution ..... 58, 60, 180, 188  
initial Hessian for ..... 32–33, 85, 93, 165,  
181–182  
input file section for Hessian ..... 226–227  
keywords for ..... 179–183  
limiting step size for 85–86, 183–184, 188  
maximum number of  
iterations ..... 83, 141, 181  
refinement of initial Hessian  
for ..... 32–33, 92–93, 165, 166, 182  
tips ..... 141–142  
troubleshooting ..... 287  
trust radius for ..... 85–86, 183–184, 188  
updating of Hessian during ..... 182  
*see also* transition state optimization
- geometry scans ..... 93–94
- geometry section, Jaguar panel ..... 26–38
- geometry, translation and rotation of during  
calculation ..... 102
- geometry-only file, reading ..... 286
- geometry—see geometry input, geometry  
optimization
- geopt program ..... 233  
output ..... 110–112
- ghost atoms, use in charge fitting to bond  
midpoints ..... 117
- Gibbs free energy calculations—see thermo-  
chemical properties
- GPTSFILE ..... 191
- gradient—see analytic gradient of energy
- Grasp program ..... 189
- grid and RwR information, keyword for  
output of ..... 205
- .grid file  
default ..... 237  
description and format ..... 248–250  
specifying in input file ..... 161
- grid program ..... 232  
output from ..... 103, 110, 112
- grid shell locations, keyword for output of 205
- grids ..... 84  
custom ..... 162, 212  
for electrostatic potential fitting.. 62, 191,  
212  
grid file ..... 161  
information in log file ..... 137  
information in output ..... 103–105  
keywords for ..... 210–212, 253  
selecting DFT ..... 50  
shells for, in grid file ..... 249–250  
specified in cutoff file ..... 253  
*see also* grids, pseudospectral
- grids, pseudospectral  
accuracy level ..... 78, 195  
basis set availability ..... 71  
keywords for ..... 196, 211, 212, 248, 253  
ultrafine ..... 78
- guess input file section ..... 227



- GVB calculations ..... 56–57, 149–155  
 from HF converged wave  
   function ..... 76–77, 199  
 from HF initial guess ..... 76  
 from input HF wavefunction ..... 199  
 generating GAUSSIAN 92 input for ..... 144  
 GVB data output ..... 208  
 keywords for ..... 170–171  
 keywords for SCF settings ..... 193–196  
 output from ..... 107–108, 131  
 output options ..... 132  
 pair selection tips ..... 141  
 printing orbitals ..... 134–135, 209  
 restarting ..... 216  
 troubleshooting ..... 286–287  
 gvbig input file section ..... 216  
 GVB pairs  
   choosing from Maestro ..... 56–57  
   definition ..... 150  
   for GVB-LMP2 calculations ..... 58  
   input file section for ..... 216–217  
   output information ..... 107–108  
   selection tips ..... 141  
   setting from Lewis dot structure 170–171  
   specifying for GAUSSIAN 92 input  
     generation ..... 144  
     troubleshooting ..... 286–287  
 GVB window ..... 56–57  
 gvbig program ..... 107, 232  
 GVB-LMP2 calculations ..... 57–58  
 GVB-RCI calculations ..... 56–57, 153–155
- H**  
 ham input file section ..... 230  
 Hamiltonians  
   information in output . 104, 107, 108, 131  
   user input of ..... 230  
 harmonic frequencies ..... 122  
 Hartree-Fock (HF) calculations .. 56, 102, 103,  
 105, 137  
   keywords for SCF settings ..... 193–196  
   output from standard ..... 102  
   printing orbitals ..... 133, 135, 209  
   used for GVB initial guess ..... 76, 108  
 heat capacity calculations—see  
   thermochemical properties  
 help ..... 20, 319  
 Help button ..... 319
- Help panel ..... 20, 319  
 Help window ..... 47–48, 281  
 hess input file section ..... 226  
 Hessian ..... 85, 92  
   coordinates for refinement of ..... 32  
   effect of quality on geometry  
     convergence ..... 83, 92  
   for IRC calculation ..... 95  
   input file section ..... 226–227  
   keywords for ..... 181–183  
   level shifting ..... 182, 183  
   reading BIOGRAF in Maestro ..... 35  
   refinement of initial ..... 32–33, 92–93,  
     165–166, 182  
   selecting initial ..... 85, 181  
   updating, keyword for ..... 182  
 Hessian refinement, specifying coordinates  
   for ..... 93  
 heteroatom pairs for local LMP2  
   calculations ..... 55–56, 172, 205  
 hfig program ..... 232  
   output from ..... 103  
 hybrid methods, DFT ..... 51  
 hybridization types, describing in Lewis  
   files ..... 256  
 hyperpolarizability  
   keywords for ..... 189–190  
   output ..... 118  
   selection of options for ..... 62–63
- I**  
 .in file—see input file  
 infrared intensities ..... 69, 192  
   output ..... 123  
 initial guess  
   choosing type ..... 76–77  
   file information for ..... 237, 242–243  
   for restarted calculations ..... 76  
   for transition metal systems . 76, 139–141  
   GVB, from HF wave function ..... 199  
   improving ..... 139–141  
   information in output ..... 103  
   input file section for ..... 227–228  
   keywords for ..... 198–199  
   orbital output in format for ..... 135, 209  
   output of GVB ..... 132, 208  
   printing orbitals after ..... 133, 209  
   stopping after ..... 144, 198

- 
- input file ..... 161–236
    - comments in ..... 40, 162
    - comments, on symmetrizing ..... 38
    - defining fragments ..... 225–226
    - description of sections ..... 162–164
    - directory ..... 44
    - echoing in output ..... 231
    - echoing in output file ..... 124
    - editing ..... 237
    - editing in Maestro ..... 46–47
    - gen section ..... 145
    - general description and format.. 161–164
    - in jaguar run command ..... 269–271
    - keywords ..... 168–213
    - lmp2 keywords ..... 172
    - name ..... 270
    - reading ..... 34–35
    - reading in Maestro ..... 34–35, 286
    - restart ..... 142–143
    - saving with Save window ..... 44–45
    - section delineators ..... 163
    - spacing characters ..... 162
    - summary of sections ..... 163
  - input file sections
    - atomic ..... 218–226
    - connect ..... 166–168
    - coord ..... 166–168
    - echo ..... 231
    - efields ..... 229
    - gen ..... 168–213
    - guess ..... 227–228
    - gvb ..... 216–217
    - ham ..... 230
    - hess ..... 226–227
    - lmp2 ..... 172, 217–218
    - nbo ..... 236
    - orbman ..... 230–231
    - path ..... 231–234
    - plot ..... 234–236
    - pointch ..... 229
    - zmat, zmat2, zmat3 ..... 164–165
    - zvar ..... 164
    - zvar, zvar2, zvar3 ..... 166
  - input of molecular structure—see geometry input
  - installation directory ..... 281–282
  - integrals, one-electron ..... 102
  - integrals, two-electron
    - contributions to energy ..... 107, 131
    - number and type computed ..... 78
  - internal coordinates
    - in optimization, keyword for ..... 180
    - specifying with connect section. 166–168
    - specifying with coord section .... 166–168
  - intrinsic reaction coordinate (IRC)
    - calculations ..... 95, 185
  - IR intensities—see infrared intensities
  - ira program ..... 232
  - irb program ..... 232
  - isotopes ..... 218–220
    - keyword for, gen section ..... 169
  - iterations, maximum number
    - geometry optimization ..... 83, 141, 181
    - SCF ..... 77, 139, 195
- J**
- J2 theory calculations ..... 81
  - jaguar command ..... 282
  - jaguar command ..... 266–275
    - jaguar babel ..... 272–275
    - jaguar batch ..... 275–280
    - jaguar help ..... 267
    - jaguar jobs ..... 267
    - jaguar kill ..... 271
    - jaguar machid ..... 267
    - jaguar platform ..... 267
    - jaguar results ..... 97–102
    - jaguar run ..... 269–271
    - jaguar sysreq ..... 267
  - Jaguar copyright information ..... 102
  - Jaguar data directory ..... 237
  - Jaguar files ..... 237
  - Jaguar programs ..... 232
  - JAGUAR\_SCRATCH environment variable 266
  - JAGUAR\_SCRIPTS environment variable . 41
  - jexec program ..... 232
  - job directory, local ..... 39, 45
  - job name ..... 39, 45, 102, 266, 270
    - resetting ..... 46
- K**
- keywords
    - atomic mass ..... 169
    - dealiasing function ..... 210
    - file output ..... 206
-

- frequency-related properties ..... 191  
 geometry optimization ..... 179  
 grid ..... 210  
 GVB ..... 170  
 initial guess ..... 198  
 lewis dot structure ..... 170  
 localization ..... 200  
 memory usage ..... 212  
 orbital output ..... 208  
 properties ..... 188  
 SCF iteration ..... 207  
 SCF method ..... 193  
 solvation ..... 187  
 standard output ..... 204  
 transition state ..... 179  
 killing jobs ..... 271
- L**
- launch host ..... 23, 283–284  
 LDA—see Local Density Approximation  
 least-squares operator Q, description of... 147–148  
 level shifting  
   of Hessian, keyword for ..... 183  
   of SCF orbital energies ..... 78  
 Lewis dot structure  
   keywords for ..... 170–171  
   .lewis file ..... 237  
   setting GVB pairs from ..... 170–171  
   setting van der Waals radii from 113, 187  
 .lewis file  
   description and format ..... 253–262  
   specifying in input file ..... 161  
 Linear Synchronous Transit (LST) methods—  
   see QST-guided transition state searches  
 LMP2 calculations, setting up ..... 55  
 lmp2 input file section ..... 217  
 LMP2 method ..... 54–56, 156–158  
   counterpoise corrections ..... 32  
   grid ..... 211  
   grid used for ..... 211  
   input file section ..... 217–218  
   keywords ..... 172–173  
   pseudospectral implementation.. 156–158  
   settings ..... 54–56  
 LMP2 pairs  
   delocalization of ..... 172, 173, 217–218  
   input file section for ... 172, 173, 217–218  
   input keywords for ..... 172, 173  
   keywords for ..... 172  
 lmp2 program ..... 232  
 lmp2der program ..... 232  
 lmp2dip program ..... 232  
 lmp2gda program ..... 232  
 lmp2gdb program ..... 232  
   output from ..... 110  
 Local Density Approximation (LDA).. 51, 159  
   dftname values for ..... 174  
 local host ..... 285  
 local job directory ..... 39, 45  
 local LMP2 method ..... 55–56  
   grid ..... 211  
   keywords ..... 205  
 local MP2 method—see LMP2 method, local  
   LMP2 method  
 Local MP2 window ..... 54–56  
 local program ..... 232  
 localization of orbitals  
   for LMP2 calculations ..... 55, 173  
   keywords ..... 173, 200  
   options for ..... 79  
 log file ..... 25, 45, 46, 136–137
- M**
- machid program ..... 233  
 machines.LINUX file ..... 292, 293  
 machines.LINUX file ..... 291  
 MacroModel .dat files, problems reading 286  
 Maestro  
   customizing settings with schrod-  
   inger.hosts file ..... 263–265, 287  
   help ..... 319  
   launching ..... 5  
   main window ..... 7  
   Monitor panel ..... 25, 40  
   problems starting ..... 283–284  
   scratch project ..... 7  
 masses  
   for frequency calculations ..... 66  
   input keyword for ..... 169  
   setting in atomic section ..... 218–220  
 memory  
   keywords for ..... 212–213  
   troubleshooting related to ..... 287  
 memory usage keywords ..... 212

- memory, disk, and i/o information  
keyword for ..... 204  
output option ..... 124–125
- Methods window ..... 74–79  
accuracy level ..... 78  
analytic corrections ..... 78  
convergence issues ..... 77–78  
initial guess selection in ..... 76–77  
keywords for ..... 193–197  
localization of orbitals ..... 79  
output from ..... 123, 131  
symmetry, use of ..... 79  
wavefunction type selection in ..... 74
- minimum energy path (MEP) calculations .. 95
- Molden orbitals file (.mol.f), keyword for 207
- molecular charge  
keyword for ..... 169  
setting for ..... 33
- molecular properties ..... 60  
calculating ..... 60–64  
output ..... 116–121
- molecular state keywords ..... 169
- Molecular State window ..... 33–34
- molecular structure—see geometry input or geometry optimization
- Møller-Plesset second-order perturbation theory—see MP2
- Monitor panel ..... 25, 40
- mouse functions ..... 14
- MP\_HOSTFILE environment variable ..... 297
- MP\_RMPOOL environment variable ..... 297
- MP2 (Møller-Plesset second-order perturbation theory) ..... 54–56, 156–158  
keywords for ..... 172  
output from ..... 106–107
- MPI flags, setting ..... 298
- MPI\_P4SSPORT environment variable ..... 293–295
- MPI\_USEP4SSPORT environment variable ..... 293–295
- mpich utility ..... 293, 295
- MQM basis set file (.bas), keyword for .. 207
- Mulliken population analysis ..... 64  
for basis functions ..... 64  
keyword for ..... 190  
output from ..... 120–121  
output of multipole moments from .... 121  
recalculating multipole moments from ..... 62, 64
- Mulliken spin populations ..... 64
- multiple Jaguar jobs, running  
from Maestro ..... 40–44  
with jaguar batch ..... 275–280
- multiplicity  
keyword for ..... 169  
setting for ..... 33–34
- multipole moments  
calculating ..... 62, 141  
constraining electrostatic potential fitting to reproduce ..... 61, 117, 142  
from electrostatic potential fit ..... 62  
keyword for ..... 189  
output ..... 116  
output option ..... 125  
recalculating from electrostatic potential fitting ..... 62, 118  
recalculating from Mulliken population analysis ..... 62, 64, 121  
tensors listed in output ..... 116  
units keyword ..... 205
- Murtagh-Sargent method, keyword for ..... 182

## N

- Natural Bond Orbital (NBO)  
calculations ..... 64, 121, 236
- nbo input file section ..... 236
- neighbor ranges ..... 244
- nice option, jaguar run command .... 270
- non-local ensity approximation (NLDA) ... 159
- nude program ..... 232
- number of iterations for geometry  
convergence, maximum ..... 83, 141, 181
- number of processors, determining optimum ..... 298
- numeric updating of Hessian, keyword for 182
- numerical gradient of energy ..... 83  
keywords for ..... 180, 184  
output from ..... 112
- numerical Hessian, printing in freq output 205
- numerical methods ..... 78, 147–149  
.cutoff file determination of ..... 253
- numerical second derivative of energy ... 65–66  
keywords for ..... 182, 184, 192

- O**
- OCBSE convergence scheme ..... 77
  - onee program ..... 232
  - output from..... 102–103, 110, 112
  - one-electron Hamiltonian
    - keyword for output of ..... 205
    - output option ..... 126
  - one-electron integrals ..... 102
    - energy contributions ..... 113, 115, 131
  - open shell singlet, keyword for ..... 199
  - open-shell systems ..... 74
    - energy contributions in output ..... 105
    - keywords for ..... 194
  - Optimization window ..... 83–94
  - optimizing geometry ..... 83–94
    - calculating forces only ..... 83, 180
    - constraining bond lengths or angles ..... 32, 86–87, 166, 181
    - constraining Cartesian coordinates ..... 29, 86–87
    - convergence criteria ..... 84–85, 111, 183
    - convergence criterion for SCF ..... 84
    - detailed output option ..... 126
    - fixed bond lengths or angles
      - for ..... 32, 166, 181
    - fixed Cartesian coordinates for ..... 29
    - frozen bond lengths or angles for... 86–87
    - frozen Cartesian coordinates for ... 86–87
    - GDIIS method keyword ..... 181
    - generating input with new geometry.. 141
    - in solution ..... 58, 60, 180
    - initial Hessian for ..... 32–33, 85, 93, 165, 181–182, 226–227
    - keywords for ..... 179–183
    - limiting step size for 85–86, 183–184, 188
    - maximum iterations ..... 83, 141, 181
    - output from..... 110
    - output of forces ..... 109
    - output options ..... 125
    - refinement of initial Hessian
      - for ..... 32–33, 92–93, 165, 166, 182
    - tips ..... 141–142
    - transition states ..... 88
    - troubleshooting ..... 287
    - trust radius ..... 85–86, 183–184, 188
    - updating of Hessian ..... 182
  - options, effect on output file ..... 106
  - orbital energies
    - in output ..... 105
    - output option ..... 131
  - orbitals
    - combining ..... 230–231
    - information in output ..... 103, 104, 107
    - keywords for output of ..... 208–210
    - output options ..... 133–136
    - plotting ..... 79–81, 234–236
    - printing for guess section ..... 228
    - reordering ..... 230–231
  - orbman input file section ..... 230
  - order of Jaguar programs run,
    - specifying ..... 231–234
  - ordering of dealiasing functions ..... 245
  - organometallics, improving convergence
    - for ..... 76, 139–141
  - output file
    - comments in ..... 40
    - echoing input file in ..... 231
    - effect of calculation options on
      - content ..... 106–123
      - general description ..... 102–123
      - location ..... 25, 46
      - reference in log file ..... 137
      - standard output settings ..... 124–129
      - summarizing ..... 97–102
  - output file information
    - basis set ..... 123
    - convergence methods other than DIIS 123
    - DFT calculation options ..... 106
    - frequency, IR, and thermochemistry
      - calculations ..... 121
    - geometry and transition state
      - optimizations ..... 109
    - GVB calculations ..... 107
    - GVB-RCI calculations ..... 108
    - GVB-RCI optimizations ..... 112
    - LMP2 calculation options ..... 106
    - properties ..... 116
    - solvation calculations ..... 112
  - output options ..... 106–136
    - atomic units ..... 125
    - bond lengths and angles ..... 125
    - connectivity table ..... 125
    - detailed timing information ..... 125
    - echo input file and parameter list ..... 124
    - files ..... 129–131, 206

- Gaussian function list (basis set)..... 126  
 Gaussian function list (derivatives).... 129  
 geometry optimization details..... 126  
 memory, disk, and i/o information..... 124  
 one-electron Hamiltonian..... 126  
 orbitals..... 133–136, 208–210  
 overlap matrix ..... 126  
 per iteration ..... 131–132, 207–208  
 standard..... 124–129, 204–205  
 output, summarizing ..... 97–102  
 output—see output file, output options, standard  
 output, file output options, per iteration out-  
 put options, orbitals, babel  
 overlap matrix  
 in DIIS error vector..... 105, 137  
 keyword for eigenvector and eigenvalue  
 output ..... 205  
 keyword for output of ..... 205  
 output option ..... 126  
 smallest eigenvalue, listed in output .. 102
- P**  
 P4\_GLOBMEMSIZE environment variable 296  
 parallel execution ..... 39, 298  
 frequency jobs ..... 289  
 jobs that can't be run..... 289  
 optimum processor number..... 298  
 parallel Jaguar module  
 IBM installation ..... 296  
 installing..... 289  
 Linux installation ..... 291  
 SGI installation ..... 290  
 partial charge  
 from ESP fit ..... 60  
 partial charges  
 Mulliken..... 64  
 PATH environment variable ..... 282, 291  
 path input file section ..... 231  
 path specifying order of programs  
 input file section for ..... 231–234  
 pbf program..... 233  
 output from geometry optimizations.. 116  
 per iteration output options 131–132, 207–208  
 physical constants and conversion factors,  
 keyword for ..... 195  
 pick state ..... 10  
 Pipek-Mezey localization..... 55, 79  
 keywords for ..... 173, 200  
 orbital printing ..... 133
- pKa calculation  
 ab initio ..... 300  
 conformational flexibility in ..... 303  
 empirical corrections in..... 302  
 equivalent sites in ..... 303, 304  
 geometry optimization in ..... 300  
 initial geometry in ..... 318  
 input files for ..... 315  
 monitoring ..... 317  
 multiple protonation sites in ..... 303, 305  
 running ..... 316  
 single point energies in ..... 301  
 solvation free energy in..... 301  
 theory ..... 300
- pKa prediction module  
 installing..... 315  
 introduction ..... 299  
 parameterized functional groups..... 306  
 results ..... 305
- plot data, generating..... 79, 214, 234  
 plot input file section..... 234  
 POE (Parallel Operating Environment)  
 options automatically set..... 298  
 version required ..... 296
- point charges, input file section for..... 229  
 pointch input file section..... 229  
 Poisson-Boltzmann equations ..... 59  
 Poisson-Boltzmann solver ..... 58–60  
 output from..... 113
- polar program ..... 232
- polarizability  
 keywords for ..... 189–190  
 options for ..... 62–63
- post program ..... 233  
 output from..... 113
- post-SCF DFT calculations..... 50
- potential energy surface scan ..... 93–94
- potential, electrostatic  
 plotting ..... 234–236
- Powell update method, keyword ..... 182
- pre program ..... 232  
 output from..... 102, 110, 113
- Preferences panel ..... 18
- pressure for thermochemical calculations  
 keyword for ..... 192  
 option for..... 69  
 output ..... 122
- P-RFO level shifting, keyword for..... 182
- probe program ..... 232

- probe radius of solvent ..... 59, 324
- processors  
   determining optimum number ..... 298  
   setting number for a host ..... 266
- product geometry  
   for IRC calculations ..... 95  
   in transition state search ..... 89  
   specifying in input file ..... 165  
   specifying in Maestro ..... 90
- product installation ..... 319
- program order, specifying ..... 231–234
- programs in Jaguar—see relevant program name
- project entries ..... 7  
   including, excluding and fixing ..... 8
- Project Facility, introduction to ..... 7
- Project Table panel ..... 8
- projects ..... 7
- properties  
   keywords for ..... 188–191  
   options for ..... 60–69
- Properties window ..... 60–64  
   electron density ..... 63  
   ESP charge fitting ..... 60–62  
   Mulliken population analysis ..... 64  
   multipole moments ..... 62  
   output from ..... 116–121  
   polarizability and  
     hyperpolarizability ..... 62–63
- pseudospectral method ..... 147–149
- publications, citing Jaguar in ..... 3
- Q**
- QST-guided transition state searches ..... 89–90  
   additional structures for ..... 165–166  
   keyword for ..... 180  
   LMP2 delocalization for ..... 173, 218
- quadratic energy error  
   keyword for output of ..... 205
- quadratic synchronous transit—see QST-guided transition state searches
- quitting Maestro ..... 25
- R**
- radian units for geometry input ..... 168
- radius  
   covalent ..... 221  
   van der Waals ..... 221
- RCI (restricted configuration interaction)  
   calculations ..... 56–57, 153–155, 216  
   output from ..... 107, 108–109
- rci program ..... 232
- reactant geometry  
   for IRC calculations ..... 95  
   in transition state search ..... 89  
   specifying in input file ..... 165  
   specifying in Maestro ..... 90
- Read window ..... 34–35
- reading input files ..... 34–35  
   file types ..... 34–35, 286  
   geometry input ..... 34–35  
   troubleshooting ..... 286
- reset option, Jaguar panel ..... 46
- RESP file  
   keyword for ..... 207
- restarting calculations ..... 142–143  
   GVB ..... 216  
   initial guess ..... 76  
   restart file ..... 206, 207  
   with improved guess ..... 141
- Restarting jobs ..... 142
- restricted configuration interaction—see RCI calculations
- restricted open-shell wave functions  
   keyword for ..... 194  
   option for ..... 74
- results, summary of ..... 97–102  
   final ..... 100  
   for each atom ..... 102  
   intermediate ..... 101
- RFO level shifting, keyword for ..... 182
- .rhosts file, use with MPICH ..... 292
- RODFT—see restricted open-shell wave functions
- ROHF—see restricted open-shell wave functions
- row, Project Table ..... 7
- Run window ..... 25, 38–40, 263
- running jobs  
   from Maestro ..... 25, 38–46  
   from the command line ..... 269–271  
   individual ..... 38–40  
   multiple ..... 40–44  
   on a remote host ..... 270  
   selecting a Jaguar version ..... 271  
   troubleshooting ..... 281–286

- rwr program..... 232  
execution sequence ..... 109, 110
- S**
- sample calculation..... 23–26  
Save window ..... 44–45, 162  
scaling frequencies..... 66–67  
keywords for ..... 192  
scanning geometries..... 93–94  
SCF energy output ..... 104  
SCF iterations  
keywords for ..... 207  
maximum number of..... 77, 139  
SCF level shift..... 78  
SCF method keywords ..... 193  
scf program..... 232  
GVB calculations ..... 108  
output from 103–105, 107–108, 110, 112–115, 131–132  
solvation calculations..... 112  
Schrödinger contact information..... 21, 319  
SCHRODINGER directory..... 281–282  
SCHRODINGER environment variable .. 5, 282  
SCHRODINGER\_MPI\_FLAGS environment variable ..... 295, 298  
SCHRODINGER\_NODEFILE environment variable ..... 293, 295  
SCHRODINGER\_POE\_FLAGS environment variable ..... 298  
SCHRODINGER\_TMPDIR environment variable ..... 266  
schrodinger.hosts file 38, 39, 263–266, 269, 286, 287  
scratch entry, project table ..... 9  
scratch project ..... 7  
SCRF method for solvation calculations 58–59, 112  
search method, transition state ..... 89  
searching along paths and eigenvectors ..... 91  
second derivative of energy  
keywords for ..... 182, 184, 192  
secure servers, MPICH ..... 293  
selecting atoms..... 10  
self-consistent reaction field method for solvation calculations ..... 58–59  
semaphores, freeing ..... 296  
shared memory, building MPICH with..... 291  
shells, information in output ..... 103  
Simons’ method for trust radius adjustment,  
keyword for ..... 183  
singlet, open shell, keyword for ..... 199  
sole program ..... 233  
output from..... 115  
solv program ..... 233  
output from..... 113  
solvation ..... 58–60  
energy output ..... 115  
keywords for ..... 187–188  
output from calculations ..... 112–116  
probe radius..... 59, 188  
solvent choice..... 59  
van der Waals radii for ..... 113, 218–221, 253–262  
Solvation window ..... 58–60  
solvent parameters ..... 59, 60  
SPARTAN archive files ..... 130, 207  
spin multiplicity  
keyword for ..... 169  
setting for ..... 33–34  
spin populations, Mulliken..... 64  
SQM frequency scaling method..... 66–67  
ssh, use with MPICH ..... 292  
standard functionals, dftname values for ... 174  
standard output  
keywords for ..... 204–205  
options for ..... 124–129  
structure, input—see geometry input  
structure, optimizing—see geometry optimization  
structures  
building ..... 9  
rotating and translating ..... 14  
submission host, definition ..... 265  
submitting jobs—see running jobs  
superblocks ..... 196  
symmetrizing geometry input ..... 37–38, 286  
symmetry  
effect on structure..... 286  
in IRC calculations..... 95, 186  
keywords for ..... 169  
option for..... 79  
output information ..... 102  
specifying for GAUSSIAN 9x input ..... 144  
use of in calculation ..... 37–38



- synchronous transit quasi-Newton methods—  
see QST-guided transition state searches
- T**
- technical support ..... 20, 319
- temperatures for thermochemical calculations  
  keywords for ..... 192  
  output ..... 122  
  setting ..... 69
- temporary directory ..... 39–40, 285  
  after job is killed ..... 271  
  errors related to ..... 284–285  
  in output file ..... 102  
  saving at end of job ..... 270  
  specifying in hosts file ..... 265–266
- temporary files ..... 40, 271  
  saving at end of job ..... 270
- thermal smearing ..... 197
- thermochemical properties ..... 69  
  keywords for ..... 192  
  output ..... 122
- time stamps in log file, option for ..... 270
- timex program ..... 233
- timing information  
  keyword for ..... 204  
  option for ..... 125
- toolbar, Maestro ..... 11
- torsional angles, in Z-matrix ..... 30
- torsions, freezing all ..... 86
- transition metals  
  improving convergence .. 76–77, 139–141  
  initial guess for ..... 76, 226
- transition state optimization ..... 83–94  
  constraining bond lengths or  
    angles ..... 86, 181  
  convergence criteria ..... 84–85, 183  
  convergence criterion for SCF ..... 84  
  eigenvector following ..... 91, 182–183  
  frozen bond lengths or angles ..... 86, 181  
  in solution ..... 58, 60, 180  
  initial Hessian ..... 32–33, 85, 93, 165,  
    181–182  
  keywords ..... 179–183  
  level shifting of Hessian ..... 182  
  limiting step size for ..... 85–86, 183–184  
  maximum iterations ..... 83, 181  
  refinement of initial Hessian ..... 32–33,  
    92–93, 165, 166, 182  
  search method ..... 89  
  trust radius ..... 85–86, 183–184  
  updating of Hessian ..... 182
- trial wave function—see Initial guess
- troubleshooting ..... 281–287
- trust radius for optimizations . 85–86, 183–188
- two-electron integrals, number of ..... 78
- U**
- UDFT—see unrestricted wave functions
- UHF—see unrestricted wave functions
- undoing a Maestro operation ..... 17
- unrestricted wavefunctions  
  keyword for ..... 194  
  option for ..... 74
- user name, setting for different hosts ..... 265
- V**
- van der Waals radii ..... 62, 221  
  for solvation calculations ..... 187  
  input file sections for ..... 218–221  
  listed in output ..... 113  
  setting from Lewis file data ..... 258
- van der Waals surface ..... 62
- variables in geometry input ..... 28–29, 31–32,  
  87–88, 166
- versions of Jaguar, listing ..... 269
- vibrational frequencies ..... 65–66  
  keywords for ..... 184, 191–193  
  scaling ..... 66–67, 192  
  visualizing in Maestro ..... 67  
  visualizing with Molden ..... 66
- virial ratio ..... 204
- virtual orbitals ..... 105, 208  
  number printed ..... 134
- W**
- wave function type ..... 74
- workflow, project based ..... 7
- Workspace  
  description ..... 7  
  including, excluding and fixing entries .. 8  
  scratch entry ..... 9  
  toolbar ..... 11
- X**
- XYZ file (.xyz) output option ..... 130

<b>Z</b>	
zero point energies—see thermochemical properties	
zmat input file section .....	164
zmat2 input file section .....	164
zmat3 input file section .....	164
Z-matrix format.....	27, 29–32, 164–166
dummy atoms in.....	31
variables in .....	31, 32, 166
zvar input file section .....	166
zvar2 input file section .....	166
zvar3 input file section .....	166

---

# Keyword Index

---

## Numerics

2spin .....220

## B

babel .....201–204

babelg .....201–204

basis .....193, 225

## C

cfiterr .....190–191

charge .....225

cov .....220

covfac .....126, 169

cut20 .....103, 198

## D

daf .....225

dcoarse .....212

dconv .....183, 190, 193–194

dconvci .....179

denspc .....189, 190, 191

dfine .....212

dftname .....173–175

dgrad .....212

dmedium .....212

dufine .....212

## E

econv .....183, 190, 193–194

econvci .....179

efield .....191

epsin .....188

epsout .....188

esolv0 .....188

esp .....220

extentx .....235

extenty .....235

extentz .....235

## F

fdtemp .....197

formal .....220

frag .....225

freqfrag .....193, 226

## G

gcharge .....191, 212, 253

gcoarse .....212, 253

gconv1-gconv7 .....184

gdftcphf .....210

gdftder2 .....210

gdftfine .....210

gdftgrad .....210, 253

gdftmed .....210, 253

geldens .....189, 191, 212, 253

gfine .....212, 253

ggrad .....212, 253

glmp2 .....212

glmp2der .....212

gmedium .....212, 253

grid .....225

gufine .....212, 253

## I

iacc .....195

iaccg .....181, 183

iacscf .....195

icanorb .....197

icavity .....187

icfit .....188

ichange .....196

icis .....179

iconv .....194

icpfrag .....225

idelfrag .....226

idelocv .....173

idenavg .....178, 196

idfgdX .....210

idft .....173–178

idoabe .....170

ifdtherm .....197

ifollow .....182

ifreq .....192

igeopt .....180

igonly .....144, 198

igscan .....180

iguess .....198–199, 227

igvball .....170, 171

igvbsel .....170, 171

<b>ihamtyp</b>	199, 230	<b>ircmax</b>	186
<b>iheter</b>	172	<b>ircmode</b>	186
<b>ihfgvb</b>	170, 199	<b>ircmxyc</b>	186
<b>ihuptyp</b>	182	<b>ircstep</b>	186
<b>ilagr</b>	181	<b>irder</b>	192
<b>imw</b>	192	<b>irefhup</b>	182
<b>incdip</b>	188, 190	<b>ireson</b>	172, 173
<b>inhess</b>	181, 182, 226	<b>irfo</b>	182
<b>intopt</b>	167, 180	<b>isolv</b>	187
<b>iorb1a</b>	215, 234	<b>isolvg</b>	187
<b>iorb1b</b>	215, 235	<b>isotope</b>	220
<b>iorb2a</b>	215, 234	<b>isqm</b>	192
<b>iorb2b</b>	215, 235	<b>istavg</b>	196
<b>iordboy</b>	200	<b>isurf</b>	187, 262
<b>ip1, ip3-ip8, ip11-ip13, ip18-ip19, ip24-ip26, ip170, ip173, ip192-ip193</b>	204–205	<b>isymm</b>	170, 190
<b>ip100</b>	209	<b>iteravg</b>	196
<b>ip100-101, ip103-107</b>	209	<b>itradj</b>	183, 184
<b>ip105</b>	209, 210	<b>itrcut</b>	183, 184
<b>ip107</b>	201	<b>itrvec</b>	183
<b>ip15, ip17, ip110, ip121-ip123, ip149, ip188, ip201</b>	208	<b>itwice</b>	196
<b>ip151</b>	143, 206	<b>iuhf</b>	194
<b>ip152</b>	143, 207	<b>iunit</b>	164, 165, 168, 214, 229
<b>ip160</b>	144, 145, 206	<b>ivanset</b>	187, 262
<b>ip165</b>	206	<b>ixtrboy</b>	201
<b>ip170</b>	107, 205		
<b>ip28</b>	191	<b>J</b>	
<b>ip29</b>	219	<b>jdft</b>	173–178
<b>ip472</b>	186	<b>jksep</b>	195
<b>ip90, ip160-ip161, ip163-ip165, ip168, ip172, ip175</b>	207		
<b>ipkat</b>	315	<b>K</b>	
<b>iplotden</b>	215	<b>ksep</b>	187, 204
<b>iplotesp</b>	215		
<b>iplotspn</b>	215	<b>L</b>	
<b>ipltunit</b>	235	<b>lastwv</b>	196
<b>ipolar</b>	189, 190, 191	<b>lcoarse</b>	212
<b>ipopsym</b>	170	<b>ldens</b>	189, 191
<b>ipvirt</b>	208	<b>ldips</b>	189
<b>iqcoarse</b>	212–213, 214	<b>lewdot</b>	170, 171, 172
<b>iqfine</b>	212–213, 214	<b>lewstr</b>	170, 171, 187, 262
<b>iqgrad</b>	212–213, 214	<b>lfine</b>	212
<b>iqmedium</b>	212–213, 214	<b>lgrad</b>	212
<b>iqst</b>	173, 180	<b>lmedium</b>	212
<b>iqufine</b>	212–213, 214	<b>loclmp2c</b>	173
<b>irc</b>	186	<b>loclmp2v</b>	173
		<b>locpostc</b>	200
		<b>locpostv</b>	200
		<b>lufine</b>	212

**M**

mass .....220  
 massav .....169  
 maxciit .....179  
 maxdiis .....194  
 maxit .....195  
 maxitcp .....192  
 maxitg .....181  
 molchg .....169, 229  
 mp2 .....172  
 mulk .....220  
 mulken .....190  
 multip .....169, 220  
 mxpage .....213  
 mxpr .....213  
 mxrwr .....213  
 mxstrip .....213

**N**

nbcmx .....213  
 nbuck .....213  
 ndfgrdX1 .....210  
 ndfgrdX2 .....210  
 ndisk .....213  
 needgwd .....181  
 newcon .....195  
 nhesref .....182  
 nmder .....66, 180, 191  
 noatcor .....195  
 noauto .....196  
 nogas .....180  
 nogdiis .....181  
 noopta .....181  
 noopttr .....181  
 nooptt .....181  
 nops .....195, 253  
 nosuper .....196  
 noupdatt .....195  
 npts .....235  
 nrestart .....179  
 nroot .....179  
 ntemp .....192  
 numd .....71, 193

**O**

origin .....235

**P**

pertnd .....180, 184  
 plotfmt .....216  
 plotres .....215  
 press .....192

**Q**

qstinit .....184

**R**

radprb .....188  
 rmscp .....192

**S**

scalfr .....192  
 sconv .....188  
 stdiis .....194

**T**

tmpini .....192  
 tmpstp .....192  
 tradmn .....184  
 tradmx .....184  
 tremx .....184  
 treok .....184  
 trescal .....184  
 trgmxx .....184  
 trust .....184, 188

**V**

vdw .....220  
 vdw2 .....220  
 vshift .....178, 194, 195

**W**

wispc .....190, 191

**X**

xadj .....215  
 xcor1-xcor4 .....177, 178  
 xcorn1-xcorn9 .....177, 178  
 xex11 .....177, 178  
 xex19 .....177, 178  
 xexn11-xexn19 .....177, 178  
 xhf .....177-178  
 xmaxadj .....215  
 xminadj .....215

<b>Y</b>		<b>y</b>	
yadj .....	216	ymaxadj .....	215
ycorl1-ycorl4 .....	178	yminadj .....	215
ycornl1-ycornl9 .....	178	<b>Z</b>	
yexl1 .....	178	zadj .....	216
yexl9 .....	178	zmaxadj .....	216
yexnl1-yexnl9 .....	178	zminadj .....	216
yhf .....	178	zpmem .....	213



---

120 West 45th Street  
32nd Floor  
New York, NY 10036

1500 SW First Avenue  
Suite 1180  
Portland, OR 97201

3655 Nobel Drive  
Suite 550  
San Diego, CA 92122

Dynamostraße 13  
68165 Mannheim  
Germany

**SCHRÖDINGER**