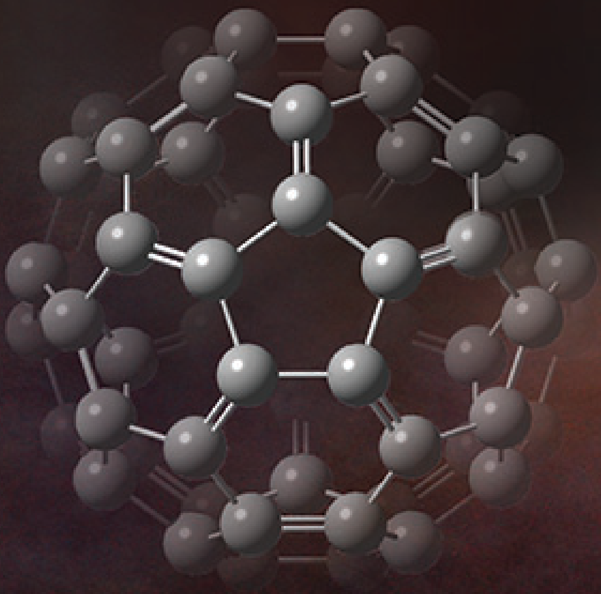
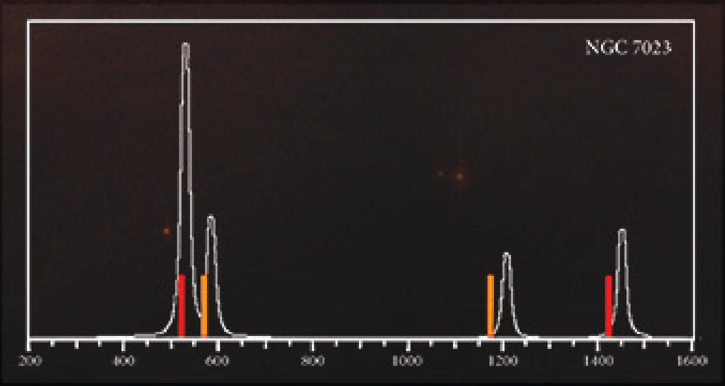


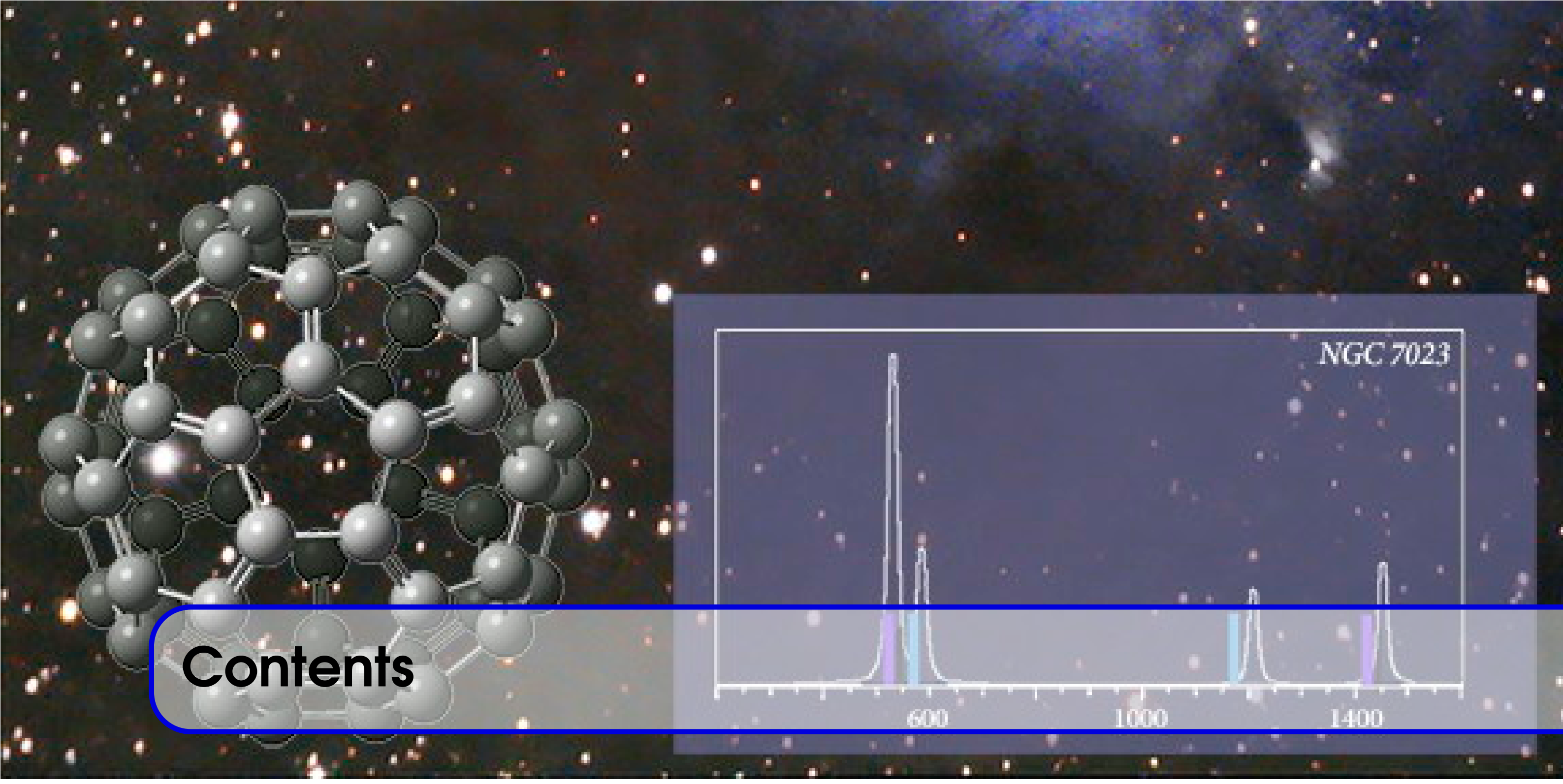


Gaussian 16 Users Reference

09 November 2017. (G16 Rev. B.01)

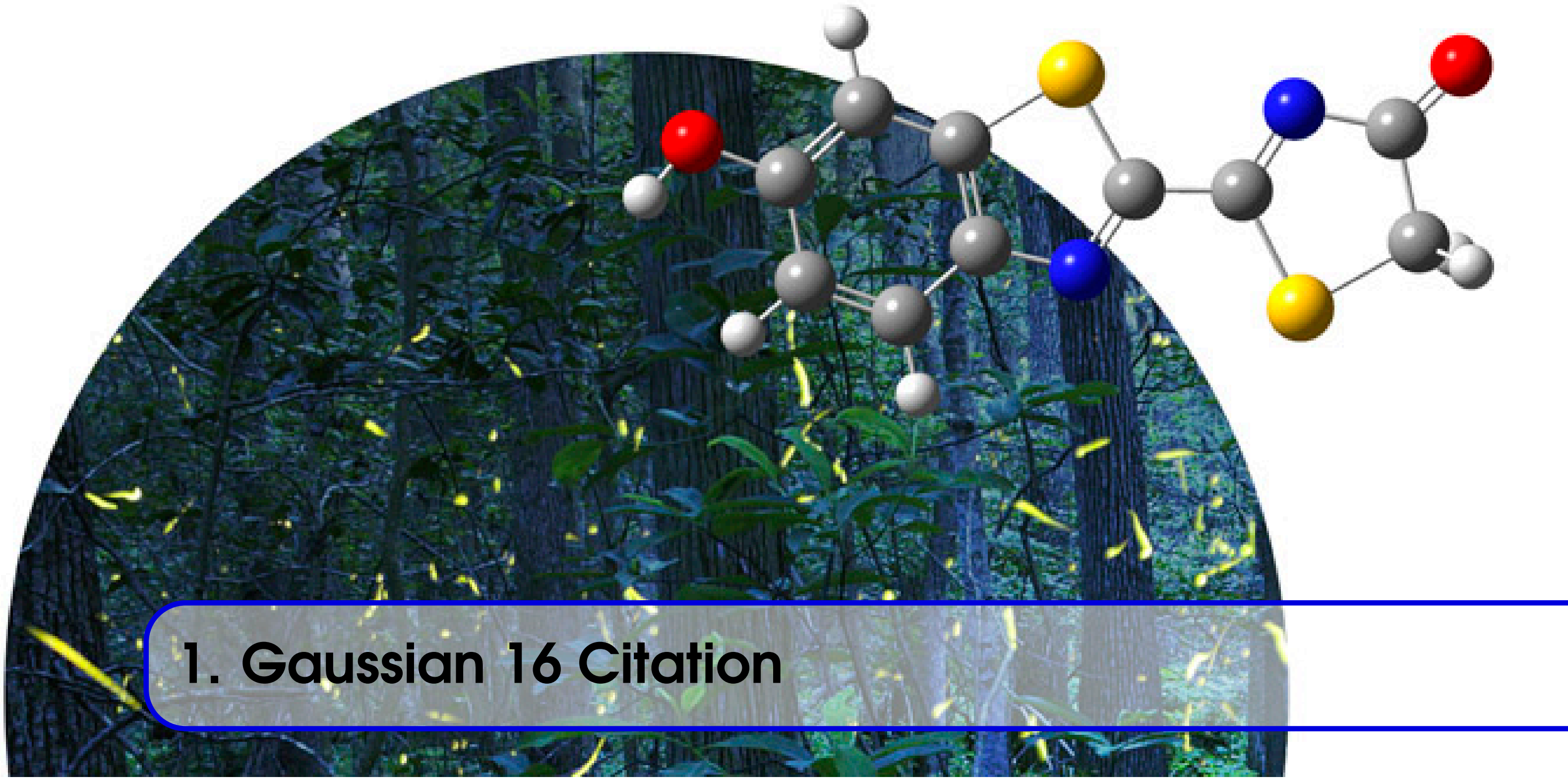
Compiled by Zork on July 20, 2018

Documented in LaTeX& compiled using XeLaTeX & Biber integrated in TeXLive 2018.



Contents

Part One



1. Gaussian 16 Citation

Gaussian 16 represents further development of the Gaussian 70, Gaussian 76, Gaussian 80, Gaussian 82, Gaussian 86, Gaussian 88, Gaussian 90, Gaussian 92, Gaussian 92/DFT, Gaussian 94, Gaussian 98, Gaussian 03 and Gaussian 09 systems previously published [1–13]. The current required citation for this work is given below; note that you should replace Revision B.01 with the identifier for the revision of the program that you actually use.

Normal Name Order :

Gaussian 16, Revision B.01, M. J. Frisch, G. W. Trucks, H. B. Schlegel, G. E. Scuseria, M. A. Robb, J. R. Cheeseman, G. Scalmani, V. Barone, G. A. Petersson, H. Nakatsuji, X. Li, M. Caricato, A. V. Marenich, J. Bloino, B. G. Janesko, R. Gomperts, B. Mennucci, H. P. Hratchian, J. V. Ortiz, A. F. Izmaylov, J. L. Sonnenberg, D. Williams-Young, F. Ding, F. Lipparini, F. Egidi, J. Goings, B. Peng, A. Petrone, T. Henderson, D. Ranasinghe, V. G. Zakrzewski, J. Gao, N. Rega, G. Zheng, W. Liang, M. Hada, M. Ehara, K. Toyota, R. Fukuda, J. Hasegawa, M. Ishida, T. Nakajima, Y. Honda, O. Kitao, H. Nakai, T. Vreven, K. Throssell, J. A. Montgomery, Jr., J. E. Peralta, F. Ogliaro, M. J. Bearpark, J. J. Heyd, E. N. Brothers, K. N. Kudin, V. N. Staroverov, T. A. Keith, R. Kobayashi, J. Normand, K. Raghavachari, A. P. Rendell, J. C. Burant, S. S. Iyengar, J. Tomasi, M. Cossi, J. M. Millam, M. Klene, C. Adamo, R. Cammi, J. W. Ochterski, R. L. Martin, K. Morokuma, O. Farkas, J. B. Foresman, and D. J. Fox, Gaussian, Inc., Wallingford CT, 2016.

Last Name First :

Gaussian 16, Revision B.01, Frisch, M. J.; Trucks, G. W.; Schlegel, H. B.; Scuseria, G. E.; Robb, M. A.; Cheeseman, J. R.; Scalmani, G.; Barone, V.; Petersson, G. A.; Nakatsuji, H.; Li, X.; Caricato, M.; Marenich, A. V.; Bloino, J.; Janesko, B. G.; Gomperts, R.; Mennucci, B.; Hratchian, H. P.; Ortiz, J. V.; Izmaylov, A. F.; Sonnenberg, J. L.; Williams-Young, D.; Ding, F.; Lipparini, F.; Egidi, F.; Goings, J.; Peng, B.; Petrone, A.; Henderson, T.; Ranasinghe, D.; Zakrzewski, V. G.; Gao, J.; Rega, N.; Zheng, G.; Liang, W.; Hada, M.; Ehara, M.; Toyota, K.; Fukuda, R.; Hasegawa, J.; Ishida, M.; Nakajima, T.; Honda, Y.; Kitao, O.; Nakai, H.; Vreven, T.; Throssell, K.; Montgomery, J. A., Jr.; Peralta, J. E.; Ogliaro, F.;

Bearpark, M. J.; Heyd, J. J.; Brothers, E. N.; Kudin, K. N.; Staroverov, V. N.; Keith, T. A.; Kobayashi, R.; Normand, J.; Raghavachari, K.; Rendell, A. P.; Burant, J. C.; Iyengar, S. S.; Tomasi, J.; Cossi, M.; Millam, J. M.; Klene, M.; Adamo, C.; Cammi, R.; Ochterski, J. W.; Martin, R. L.; Morokuma, K.; Farkas, O.; Foresman, J. B.; Fox, D. J. Gaussian, Inc., Wallingford CT, 2016.

BibTex Starting Point :

```
@misc{g16,
author={M. J. Frisch and G. W. Trucks and H. B. Schlegel and G. E. Scuseria
   and M. A. Robb and J. R. Cheeseman and G. Scalmani and V. Barone and G.
   A. Petersson and H. Nakatsuji and X. Li and M. Caricato and A. V.
   Marenich and J. Bloino and B. G. Janesko and R. Gomperts and B. Mennucci
    and H. P. Hratchian and J. V. Ortiz and A. F. Izmaylov and J. L.
   Sonnenberg and D. Williams-Young and F. Ding and F. Lipparini and F.
   Egidi and J. Goings and B. Peng and A. Petrone and T. Henderson and D.
   Ranasinghe and V. G. Zakrzewski and J. Gao and N. Rega and G. Zheng and
   W. Liang and M. Hada and M. Ehara and K. Toyota and R. Fukuda and J.
   Hasegawa and M. Ishida and T. Nakajima and Y. Honda and O. Kitao and H.
   Nakai and T. Vreven and K. Throssell and Montgomery, {Jr.}, J. A. and J.
    E. Peralta and F. Ogliaro and M. J. Bearpark and J. J. Heyd and E. N.
   Brothers and K. N. Kudin and V. N. Staroverov and T. A. Keith and R.
   Kobayashi and J. Normand and K. Raghavachari and A. P. Rendell and J. C.
    Burant and S. S. Iyengar and J. Tomasi and M. Cossi and J. M. Millam
   and M. Klene and C. Adamo and R. Cammi and J. W. Ochterski and R. L.
   Martin and K. Morokuma and O. Farkas and J. B. Foresman and D. J. Fox},
title={Gaussian~16 {R}evision {B}.01},
year={2016},
note={Gaussian Inc. Wallingford CT}
}
```

EndNote/Papers Import File :

Clicking on the following link will have variable results, depending on your browser and system configuration. Right click to download the file (save with extension .enw): `http://gaussian.com/dl/g16_b01.enw`

1.1 Additional Citation Recommendations

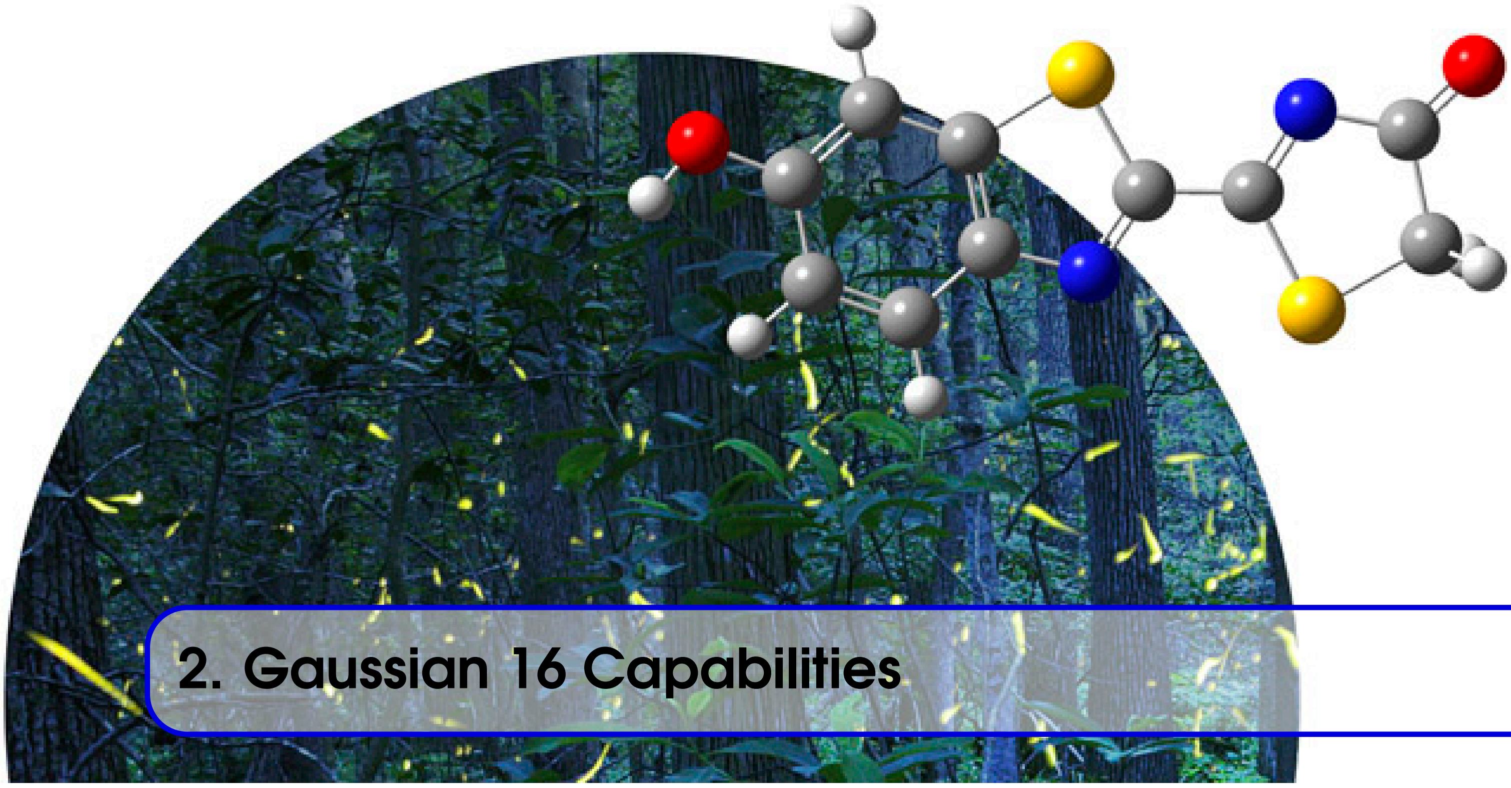
In general, we recommend citing the original references describing the theoretical methods used when reporting results obtained from Gaussian calculations, as well as giving the citation for the program itself. These references are given in the discussions of the relevant keywords. The only exceptions occur with long established methods such as Hartree-Fock theory which have advanced to the state of common practice and are essentially self-citing at this point.

In some cases, Gaussian output will display the references relevant to the current calculation type.

Gaussian also includes the NBO program as link 607. If this program is used, it should be cited separately as:

NBO Version 3.1, E. D. Glendening, A. E. Reed, J. E. Carpenter, and F. Weinhold.

The original literature references for NBO can also be cited [14–21].



2. Gaussian 16 Capabilities

2.1 Model Chemistries

The combination of method and basis set specifies a model chemistry to Gaussian, specifying the level of theory. Every Gaussian job must specify both a method and basis set. This is usually accomplished via two separate keywords within the route section of the input file, although a few method keywords imply a choice of basis set. Some jobs using a density functional method may also include a density fitting set (see Basis Sets for more information).

The following table lists methods which are available in Gaussian, along with the job types for which each one may be used. An asterisk indicates analytic calculations, while numerical-only calculations are indicated by *num* (see the discussion of the specific keyword in question for details).

Method Availabilities in Gaussian 16

	SP, Scan	Opt, Force	Freq	IRC	BOMD	ADMP	Polar	Stable	ONIOM	SCRF	PBC	EET
Molecular Mechanics methods	*	*	*		*				*	*		
AM1, PM3, PM3MM, PM6, PDDG, PM7	*	*	*	*	*	*	*	*	*	*		
HF	*	*	*	*	*	*	*	*	*	*	*	
DFT methods	*	*	*	*	*	*	*	*	*	*	*	
CASSCF	*	*	*	*	*		*		*	*		
MP2	*	*	*	*	*		*		*	*		
MP3, MP4(SDQ)	*	*	*num*	*	*		*num*		*	*		
MP4(SDTQ), MP5	*	*num*							*			
BD, CCD, CCSD, QCISD	*	*	*num*	*	*		*num*		*	*		
BD(T), CCSD(T), QCISD(T) or (TQ)	*	*num*							*			
EPT	*								*			
CBS, G*n*, W1 methods	*									*		
CIS	*	*	*	*	*		*		*	*		*
TD	*	*	*	*	*		*		*	*		*
EOM	*	*	*num*	*	*		*num*		*	*		*
ZINDO	*								*	*		
DFTB (*analytic*)	*	*	*	*	*		*		*			
CID, CISD	*	*	*num*	*	*		*num*		*	*		
GVB	*	*		*	*				*			

*=Analytic algorithm available *num*=Available via numeric differentiation

If no method keyword is specified, HF is assumed. Most method keywords may be prefaced by R for closed-shell restricted wavefunctions, U for unrestricted open-shell wavefunctions, or RO for restricted open-shell wavefunctions: for example, ROHF, UMP2, or RQCISD. RO is available only for Hartree-Fock and Density Functional methods, and AM1, PM3, PM3MM, PM6, and PDDG energies and gradients, and MP2, MP3, MP4, and CCSD energies.

In general, *only a single method keyword should be specified*, and including more than one of them will produce *bizarre* results. However, there are exceptions:

◊ CASSCF may be specified along with MP2 to request a CASSCF calculation including dynamic electron correlation.

◊ ONIOM and IRCMax jobs require multiple method specifications. However, they are given as options to the corresponding keyword.

◊ The form *model2 // model1* described in Job Type may be used to generate an automatic optimization followed by a single point calculation at the optimized geometry.

2.2 Job Types

The following table lists the job types available in Gaussian 16:

◊ SP: Single point energy.

◊ Opt: Geometry optimization.

◊ Freq: Frequency and thermochemical analysis.

◊ IRC: Reaction path following.

◊ IRCMax: Find the maximum energy along a specific reaction path.

◊ Scan: Potential energy surface scan.

◊ Polar: Polarizabilities and hyperpolarizabilities.

◊ ADMP and BOMD: Direct dynamics trajectory calculation.

◊ EET: Excitation energy transfer calculation.

◊ Force: Compute forces on the nuclei.

◊ Stable: Test wavefunction stability.

◊ Volume: Compute molecular volume.

◊ Density=Checkpoint Guess=Only: Recompute population analysis only.

◊ Guess=Only: Print initial guess only; generate fragment-based initial guess.

In general, only one job type keyword should be specified. The exceptions to this rule are:

◊ Polar and Opt may be combined with Freq. In the latter case, the geometry optimization is automatically followed by a frequency calculation at the optimized structure.

◊ Opt may be combined with the compound method keywords in order to specify options for the optimization portion of the calculation: e.g., Opt=(TS,ReadFC) CBS-QB3.

When no job type keyword is specified within the route section, the default calculation type is usually a single point energy calculation (SP). However, a route section of the form: *method2/basis2 // method1/basis1* may be used to request an optimization calculation (at *method1/basis1*) followed by a single point energy calculation (at *method2/basis2*) at the optimized geometry. For example, the following route section requests a B3LYP/6-31G(d) geometry optimization followed by a single point energy calculation using the CCSD/6-31G(d) model chemistry:

```
# CCSD/6-31G(d)//B3LYP/6-31G(d)
```

In this case, the Opt keyword is optional and is the default. Note that Opt Freq calculations may *not* use this syntax.

2.2.1 Molecular Properties

The following table provides a mapping between commonly-desired predicted quantities and the Gaussian 16 keywords that will produce them:

- ◊ Anharmonic IR/Raman/VCD/ROA spectra: Freq=Anharmonic
- ◊ Antiferromagnetic coupling: Guess=Fragment, Stable
- ◊ Atomic charges: Pop
- ◊ ΔG of solvation: SCRF=SMD
- ◊ Dipole moment: Pop
- ◊ Electron affinities: CBS-QB3, CCSD, EPT
- ◊ Electron density: cubegen
- ◊ Electronic circular dichroism: CIS, TD, EOM, SAC-CI
- ◊ Electrostatic potential: cubegen, Prop
- ◊ Electrostatic potential-derived charges: Pop=Chelp, ChelpG or MK
- ◊ Electronic transition band shape: Freq=FranckCondon, Freq=HerzbergTeller
- ◊ Polarizabilities/hyperpolarizabilities: Freq, Polar, CPHF=RdFreq, Polar=DCSHG
- ◊ High accuracy energies: CBS-QB3, G2, G3, G4, W1U, W1BD
- ◊ Hyperfine coupling constants (anisotropic): Prop
- ◊ Hyperfine spectra tensors (including *g* tensors): Freq=(VCD, VibRot, Anharmonic)
- ◊ Ionization potentials: CBS-QB3, CCSD, EPT
- ◊ IR and Raman spectra: Freq
- ◊ Pre-resonance Raman spectra: Freq CPHF=RdFreq
- ◊ Resonance Raman spectra: Freq=ReadFCHT
- ◊ Molecular orbitals: Pop=Regular
- ◊ Multipole moments: Pop
- ◊ NMR shielding and chemical shifts: NMR
- ◊ NMR spin-spin coupling constants: NMR=Mixed
- ◊ Optical rotations: Polar=OptRot
- ◊ Raman optical activity: Freq=ROA
- ◊ Thermochemical analysis: Freq
- ◊ UV/Visible spectra: CIS, ZIndo, TD, EOM, SAC-CI
- ◊ Vibration-rotation coupling: Freq=VibRot
- ◊ Vibrational circular dichroism: Freq=VCD
- ◊ Vibronic spectra: Freq=ReadFCHT

2.3 Program Limits

This section lists the various size limitations that exist within Gaussian 16.

- ◊ The integral program has the following limitations:
  - The maximum number of atoms is 250,000.
  - The maximum total number of primitive shells is 750,000.
  - The maximum number of primitive d-shells and higher is 250,000.
  - The maximum number of contracted shells is 250,000.
  - The maximum degree-of-contraction allowed is 100.
- ◊ Opt=(EF,EnOnly) optimizations–useful only for methods without analytic gradients–are limited to 50 variables.
- ◊ The GVB program is limited to 100 paired orbitals (which is not a restriction in practice).
- ◊ The internal version of NBO 3 is dimensioned for 250,000 atoms and 10,000 basis functions.

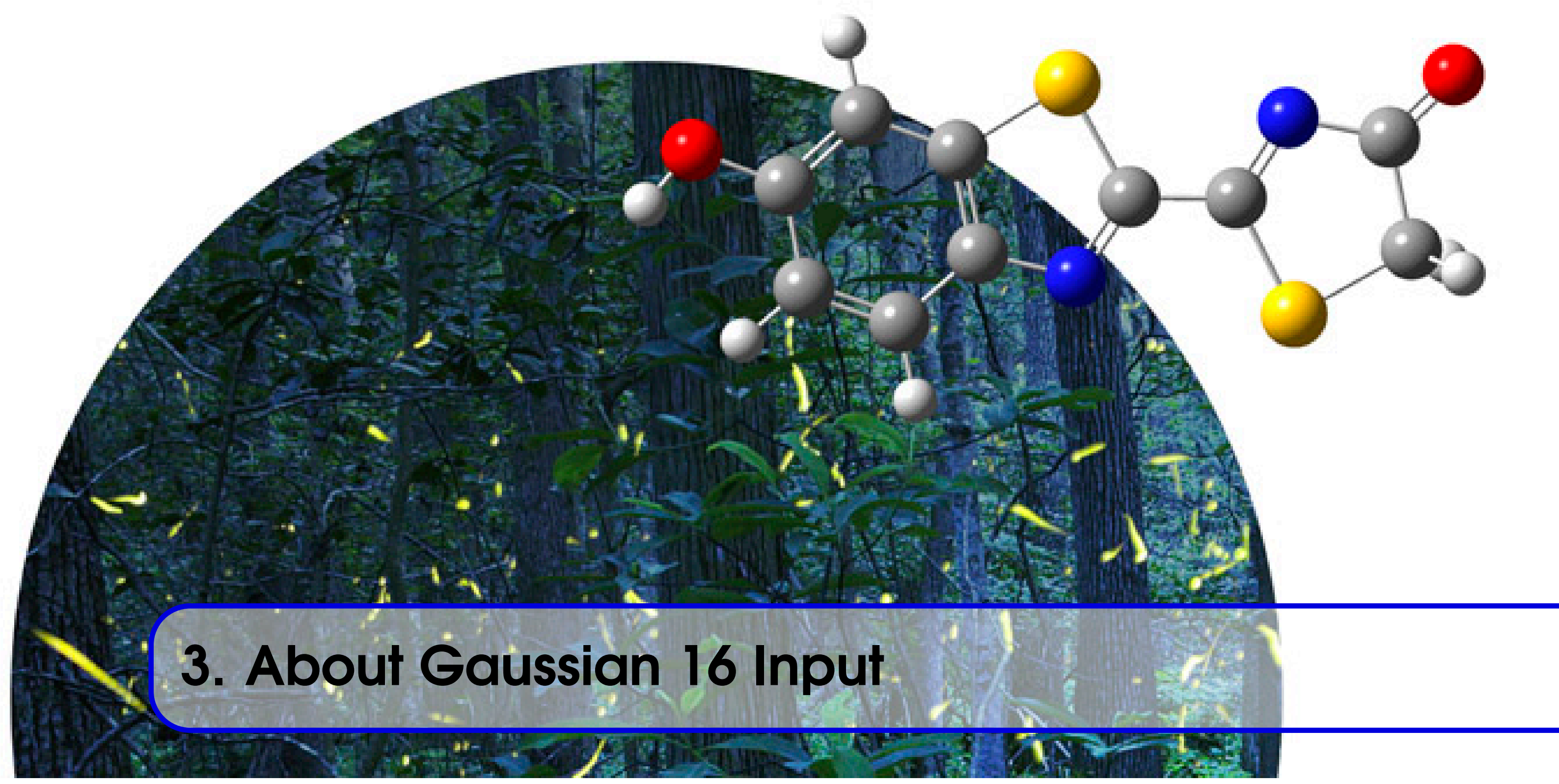
2.4 Links

The following are the component programs of Gaussian 16–known as *links*–along with their primary functions:

- ◊ L0: Initializes program and controls overlaying
- ◊ L1: Processes route section, builds list of links to execute, and initializes scratch files
- ◊ L101: Reads title and molecule specification
- ◊ L102: Fletcher-Powell optimizations
- ◊ L103: Berny optimizations to minima and TS, STQN transition state searches
- ◊ L105: Murtaugh-Sargent optimizations
- ◊ L106: Numerical differentiation of forces/dipoles to obtain polarizability/ hyperpolarizability
- ◊ L107: Linear-synchronous-transit (LST) transition state search
- ◊ L108: Unrelaxed potential energy surface scan
- ◊ L109: Newton-Raphson optimization
- ◊ L110: Double numerical differentiation of energies to produce frequencies
- ◊ L111: Double numerical differentiation of energies to compute polarizabilities and hyperpolarizabilities
- ◊ L112: Performs the Self-Consistent Virial Scaling method (SCVS), T. A. Keith's extension of [22–24]
- ◊ L113: EF optimization using analytic gradients
- ◊ L114: EF numerical optimization (using only energies)
- ◊ L115: Follows reaction path using GS3 algorithm
- ◊ L116: Numerical self-consistent reaction field (SCRF)
- ◊ L117: Performs IPCM solvation calculations.
- ◊ L118: BOMD calculations
- ◊ L120: Controls ONIOM calculations
- ◊ L121: ADMP calculations
- ◊ L122: Counterpoise calculations
- ◊ L123: Follows reaction path using the HPC algorithm (and others)
- ◊ L124: Performs ONIOM with PCM and external-iteration PCM
- ◊ L202: Reorients coordinates, calculates symmetry, and checks variables

- ◊ L301: Generates basis set information
- ◊ L302: Calculates overlap, kinetic, and potential integrals
- ◊ L303: Calculates multipole integrals
- ◊ L308: Computes dipole velocity and $\mathrm{Rx}\nabla$ integrals
- ◊ L310: Computes spdf 2-electron integrals in a primitive fashion
- ◊ L311: Computes sp 2-electron integrals
- ◊ L314: Computes spdf 2-electron integrals
- ◊ L316: Prints 2-electron integrals
- ◊ L319: Computes 1-electron integrals for approximate spin orbital coupling
- ◊ L401: Forms the initial MO guess
- ◊ L402: Performs semi-empirical and molecular mechanics calculations
- ◊ L405: Initializes an MCSCF calculation
- ◊ L502: Iteratively solves the SCF equations (conven. UHF & ROHF, all direct methods, SCRF)
- ◊ L503: Iteratively solves the SCF equations using direct minimization
- ◊ L506: Performs an ROHF or GVB-PP calculation
- ◊ L508: Quadratically convergent SCF program
- ◊ L510: MC-SCF
- ◊ L601: Population and related analyses (including multipole moments)
- ◊ L602: 1-electron properties (potential, field, and field gradient)
- ◊ L604: Evaluates MOs or density over a grid of points
- ◊ L607: Performs NBO analyses
- ◊ L608: Non-iterative DFT energies
- ◊ L609: Atoms in Molecules properties
- ◊ L610: Numerical integration (for testing integral codes)
- ◊ L701: 1-electron integral first or second derivatives
- ◊ L702: 2-electron integral first or second derivatives (sp)
- ◊ L703: 2-electron integral first or second derivatives (spdf)
- ◊ L716: Processes information for optimizations and frequencies
- ◊ L801: Initializes transformation of 2-electron integrals
- ◊ L802: Performs integral transformation (N^3 in-core)
- ◊ L804: Integral transformation
- ◊ L811: Transforms integral derivatives & computes their contributions to MP2 2^{nd} derivatives
- ◊ L901: Anti-symmetrizes 2-electron integrals
- ◊ L902: Determines the stability of the Hartree-Fock wavefunction
- ◊ L903: Old in-core MP2
- ◊ L904: Complete basis set (CBS) extrapolation method of Petersson, et. al.
- ◊ L905: Complex MP2
- ◊ L906: Semi-direct MP2
- ◊ L908: Electron Propagator Program
- ◊ L909: ADC(3) and related electron propagator models
- ◊ L913: Calculates post-SCF energies and gradient terms

- ◊ L914: CI-Singles, RPA and ZIndo excited states; SCF stability
- ◊ L915: Computes fifth order quantities (for MP5, QCISD(TQ) and BD(TQ))
- ◊ L916: Old MP4 and CCSD
- ◊ L918: Reoptimizes the wavefunction
- ◊ L923: SAC-CI program
- ◊ L925: Implements the Excited State Electron Transfer (EET) model
- ◊ L1002: Iteratively solves the CPHF equations; computes various properties (including NMR)
- ◊ L1003: Iteratively solves the CP-MCSCF equations
- ◊ L1014: Computes analytic CI-Singles second derivatives
- ◊ L1101: Computes 1-electron integral derivatives
- ◊ L1102: Computes dipole derivative integrals
- ◊ L1110: 2-electron integral derivative contribution to $F^{(x)}$
- ◊ L1111: 2 particle density matrix and post-SCF derivatives
- ◊ L1112: MP2 second derivatives
- ◊ L9999: Finalizes calculation and output



3. About Gaussian 16 Input

3.1 Syntax

Gaussian 16 input consists of a series of lines in an ASCII text file. The basic structure of a Gaussian input file includes several different sections:

◊ *Link 0 Commands*: Locate and name scratch files (not blank line terminated).

◊ *Route section* (*# lines*): Specify desired calculation type, model chemistry, and other options (blank line terminated). See Model Chemistries and Job Types for information about Gaussian 16 capabilities.

◊ *Title section*: Brief description of the calculation (blank line-terminated). This section is required in the input, but is not interpreted in any way by the Gaussian 16 program. It appears in the output for purposes of identification and description. Typically, this section might contain the compound name, its symmetry, the electronic state, and any other relevant information. The title section cannot exceed five lines and must be followed by a terminating blank line. The following characters should be avoided in the title section: @ # ! – _ \ *control characters* (*especially* Ctrl-G)

◊ *Molecule specification*: Specify molecular system to be studied (blank line-terminated). Full information is available in Molecule Specifications.

◊ *Optional additional sections*: Additional input needed for specific job types (usually blank line-terminated).

Many Gaussian 16 jobs will include only the second, third, and fourth sections. Here is an example of such a file, which requests a single point energy calculation on water:

```
# HF/6-31G(d)                         Route section

water energy                          Title section

0   1                                 Molecule specification
O  -0.464   0.177   0.0
H  -0.464   1.137   0.0
```

```
H    0.441   -0.143    0.0
```

In this job, the route and title sections each consist of a single line. The molecule specification section begins with a line giving the charge and spin multiplicity for the molecule: 0 charge (neutral molecule) and spin multiplicity 1 (singlet) in this case. The charge and spin multiplicity line is followed by lines describing the location of each atom in the molecule; this example uses Cartesian coordinates to do so. Molecule specifications are discussed in more detail later in this chapter.

The following input file illustrates the use of Link 0 commands and an additional input section:

```
%Chk=heavy                          Link 0 section
# HF/6-31G(d) Opt=ModRedun          Route section

Opt job                             Title section

0   1                               Molecule Specification section
atomic coordinates···

3 8                                 Add a bond and an angle to the internal
2 1 3                               coordinates used during the geom. opt.
```

This job requests a geometry optimization. The input section following the molecule specification is used by the Opt=ModRedundant keyword, and it serves to add an additional bond and angle in the internal coordinates used in the geometry optimization. The job also specifies a name for the checkpoint file.

For convenience, the table in the Section Ordering section details all possible sections that might appear within a Gaussian 16 input file along with the keywords associated with each one.

3.1.1 Syntax Rules

In general, Gaussian input is subject to the following syntax rules:

- ◊ Input is free-format and case-insensitive.
- ◊ Spaces, tabs, commas, or forward slashes can be used in any combination to separate items within a line. Multiple spaces are treated as a single delimiter.
- ◊ Options to keywords may be specified in any of the following forms:
  - *keyword = option*
  - *keyword(option)*
  - *keyword=(option1, option2, · · ·)*
  - *keyword(option1, option2, · · ·)*
- ◊ Multiple options are enclosed in parentheses and separated by any valid delimiter (commas are conventional and are shown above). The equals sign before the opening parenthesis may be omitted, or spaces may optionally be included before and/or after it. Note that some options also take values; in this case, the option name is followed by an equals sign: for example, CBSExtrap(NMin=6).
- ◊ All keywords and options may be shortened to their shortest unique abbreviation within the entire Gaussian 16 system. Thus, the Conventional option to the SCF keyword may be abbreviated to Conven, but not to Conv (due to the presence of the Convergence option). This holds true whether or not both Conventional and Convergence happen to be valid options for any given keyword.

◊ The contents of an external file may be included within a Gaussian 16 input file using the following syntax: *@filename*. This causes the entire file to be placed at the current location in the input stream. Appending /N to such commands will prevent the included file's contents from being echoed at the start of the output file.

◊ Comments begin with an exclamation point (!), which may appear anywhere on a line. Separate comment lines may appear anywhere within the input file.

3.2 Molecule Specifications

3.2.1 Overview

This input section specifies the nuclear positions and the number of electrons of α- and β-spin. There are several ways in which the nuclear configuration can be specified: as a Z-matrix, as Cartesian coordinates, or as a mixture of the two (note that Cartesian coordinates are just a special case of the Z-matrix).

The first line of the molecule specification section specifies the net electric charge (a signed integer) and the spin multiplicity (usually a positive integer). Thus, for a neutral molecule in a singlet state, the entry 0 1 is appropriate. For a radical anion, -1 2 would be used. Multiple charge/spin pairs may/must be included for some calculation types.

The charge and spin line is the only molecule specification input required if Geom=Checkpoint is used. The entire molecule specification (and title section) may be omitted by including Geom=AllCheck in the route section.

The remainder of the molecule specification gives the element type and nuclear position for each atom in the molecular system. The most general format for the line within it is the following:

Element-label[-Atom-type[-Charge]][(param=value[, ···])] Atom-position-parameters

Each line contains the element type, and possibly an optional molecular mechanics atom type and partial charge. Nuclear parameters for this atom are specified in the parenthesized list. The remainder of the line contains information about the atom's location, either as Cartesian coordinates or as a Z-matrix definition. We'll begin by considering the initial and final items, and then go on to discuss the remaining items.

Here is the basic format for specifying atoms within the molecule specification (omitting all of the optional items):

Element-label [freeze-code] x y z

Although these examples use spaces to separate items within a line, any valid separator may be used. The position of the atom is specified in Cartesian coordinates. *freeze-code* is an optional parameter related to freezing atoms during optimizations.

Element-label is a character string consisting of either the chemical symbol for the atom or its atomic number. If the elemental symbol is used, it may be optionally followed by other alphanumeric characters to create an identifying label for that atom. A common practice is to follow the element name with a secondary identifying integer: C1, C2, C3, and so on; this technique is useful in following conventional chemical numbering. The maximum length of the element label is 4 characters.

The remaining items on each line are Cartesian coordinates specifying the position of that nucleus.

Here is a simple molecule specification section for ethane which uses element labels for the carbon atoms and element types for the hydrogen atoms:

```
0,1
```

```
C1   0.00    0.00    0.00
C2   0.00    0.00    1.52
H    1.02    0.00   -0.39
H   -0.51   -0.88   -0.39
H   -0.51    0.88   -0.39
H   -1.02    0.00    1.92
H    0.51   -0.88    1.92
H    0.51    0.88    1.92
```

Additional information can be specified for each atom with a parenthecized keyword list following the element specification/label. For example:

```
C(Iso=13,Fragment=2) 0.00 0.00 0.00
```

The available items are documented in the other tabs.

Z-matrix Input

Z-matrix molecule specifications are also accepted. In this case, the *atom-position-parameters* are the labels or line numbers for previously-specified atoms which will be used to define the current atom's position; we will designate them here as *atom1*, *atom2* and *atom3*. The position of the current atom is specified by giving the length of the bond joining it to *atom1*, the angle formed by this bond and the bond joining *atom1* and *atom2*, and the dihedral (torsion) angle formed by the bond joining *atom2* and *atom3* with the plane containing the current atom, *atom1* and *atom2*.

See Constructing Z Matrices for details.

3.2.2 Nuclei Properties

Isotopes and other nuclear parameters can be specified within the atom type field using parenthesized keywords and values, as in the following example:

```
C(Iso=13,Spin=3) 0.0 0.0 0.0
```

The line specifies a ^{13}C atom with a nuclear spin of 3/2 (3 * 1/2), located at the origin. The following items may be included in the list of parameters:

◊ Iso=*n*: Isotope selection. If integers are used to specify the atomic masses, the program will automatically use the corresponding actual exact isotopic mass (e.g., 18 specifies ^{18}O, and Gaussian uses the value 17.99916).
◊ Spin=*n*: Nuclear spin, in units of 1/2.
◊ ZEff=*n*: Effective charge. This parameter is used in spin orbit coupling (see CASSCF=SpinOrbit), and the ESR *g* tensor and the electronic spin-molecular rotation hyperfine tensor (NMR, Output=Pickett).
◊ QMom=*n*: Nuclear quadrupole moment.
◊ NMagM=*n*: Nuclear magnetic moment in nuclear magnetons.
◊ ZNuc=*n*: Modifies nuclear charge.
◊ RadNuclear=*radius*: Specifies the radius to use when finite (non-point) nuclei are used. *radius* is a floating point value in atomic units.

3.2.3 Fragments

Fragments within a molecular system may be defined using the Fragment parameter, which appears in parentheses following the atom label along with any isotope and/or nuclear parameter values. The value to Fragment is an integer; all atoms with the same fragment number are defined as a fragment. Fragments are useful for fragment guess calculation, counterpoise calculations, and so on.

For example, the following biphenyl structure is divided into two fragments by benzene ring:

```
0,1 0,1 0,1          Total spin & charge, followed by fragment-specific ones.
C(Fragment=1)      -3.05015529   -0.24077322    0.00000698
C(Fragment=1)      -1.64875545   -0.24070572    0.00067327
C(Fragment=1)      -0.94811361    0.97297577    0.00020266
C(Fragment=1)      -1.64887160    2.18658975   -0.00093259
C(Fragment=1)      -3.05027145    2.18652225   -0.00159819
C(Fragment=1)      -3.75091329    0.97284076   -0.00112735
H(Fragment=1)      -3.58511088   -1.16744597    0.00036555
H(Fragment=1)      -1.11371117   -1.16732692    0.00154256
H(Fragment=1)      -1.11391601    3.11326250   -0.00129286
H(Fragment=1)      -3.58531573    3.11314346   -0.00246648
H(Fragment=1)      -4.82091317    0.97278922   -0.00163655
C(Fragment=2)       0.59188622    0.97304995    0.00093742
C(Fragment=2)       1.29252806    2.18673144    0.00046795
C(Fragment=2)       1.29264421   -0.24056403    0.00207466
C(Fragment=2)       2.69392790    2.18679894    0.00113535
C(Fragment=2)       2.69404405   -0.24049653    0.00274263
C(Fragment=2)       3.39468590    0.97318496    0.00227326
H(Fragment=2)       0.75768862   -1.16723678    0.00243403
H(Fragment=2)       0.75748378    3.11335264   -0.00040118
H(Fragment=2)       3.22888349    3.11347169    0.00077519
H(Fragment=2)       3.22908834   -1.16711773    0.00360969
H(Fragment=2)       4.46468577    0.97323650    0.00278063
```

This example also illustrates the use of fragment-specific charge and spin multiplicity specifications. The format of the corresponding input line in this case is:

total charge, total spin, fragment1 charge, fragment1 spin, fragment2 charge, fragment2 spin

Negative spin multiplicity values have a special meaning for Guess=Fragment calculations, indicating that the unpaired orbitals for the corresponding fragment are to become β spin orbitals in the combined set specified. Negative spin multiplicities will generate an error in any other job type.

For Guess=Fragment and Counterpoise calculations, fragment numbers must begin at 1 and run consecutively. For other calculation types, this restriction is not enforced, but violating it may result in some extraneous, empty data sections in the output (e.g., all zero fragment population analyses).

GaussView provides a graphical tool for defining fragments.

3.2.4 MM Types

Molecule specifications for molecular mechanics calculations may also include atom typing and partial charge information. Here are some examples:

```
C-CT               Specifies an SP3 aliphatic carbon atom.
C-CT-0.32          Specifies an SP3 aliphatic carbon atom with a partial charge of 0.32.
```

`O-O-0.5` *Specifies a carbonyl group oxygen atom with a partial charge of -0.5.*

Atom types and optional partial charges can be specified for each atom. Nuclear parameters can also be defined, as in these examples:

```
C-CT(Iso=13)
C-CT--0.1(Spin=3)
```

3.2.5 PDB Parameters

Several additional items may be defined along with the nuclear parameters and/or fragment definitions. These items are designed for use with PDB files to retain residue and other structural information they contain and as such will not be defined by the user. However, you may see them in Gaussian 16 input files created by GaussView using structures originating in PDB files.

- ◊ RESNum specifies the residue in which the atom is located. The value takes the form of $n[X[Y]]$, where n is an integer (which need not be positive), X is the optional one-character insertion code, and Y is the optional chain letter. If the chain is specified but there is no insertion code, then X can be an underscore: ResNum=-17_C for the residue with number -17 in chain C.
- ◊ RESName specifies the three character residue name.
- ◊ PDBName specifies the name assigned to the atom if it is not just the element name.

3.2.6 Ghost Atoms

An atom with mechanics type Bq (e.g., O-Bq) is set up as a ghost [25] of the corresponding atom, with its normal basis functions and numerical integration grid points but no nuclear charge or electrons. This requests a counterpoise calculation. Such calculations differ slightly from ones requested with Massage in previous versions of Gaussian in that they include the grid points from the ghost atoms in DFT XC quadrature. The new way is a more consistent superposition correction and also easier to use. Note that counterpoise calculations can also be requested with the Counterpoise keyword.

3.2.7 Periodic Systems

Periodic systems are specified with a normal molecule specification for the unit cell. The only additional required input are one, two or three translation vectors appended to the molecule specification (with no intervening blank line), indicating the replication direction(s). For example, the following input specifies a one-dimensional PBC single point energy calculation for neoprene:

```
# PBEPBE/6-31g(d,p)/Auto SCF=Tight

neoprene, -CH2-CH=C(Cl)-CH2- optimized geometry

0 1
C,-1.9267226529,0.4060180273,0.0316702826
H,-2.3523143977,0.9206168644,0.9131400756
H,-1.8372739404,1.1548899113,-0.770750797
C,-0.5737182157,-0.1434584477,0.3762843235
H,-0.5015912465,-0.7653394047,1.2791284293
```

```
C,0.5790889876,0.0220081655,-0.3005160849
C,1.9237098673,-0.5258773194,0.0966261209
H,1.772234452,-1.2511397907,0.915962512
H,2.3627869487,-1.0792380182,-0.752511583
Cl,0.6209825739,0.9860944599,-1.7876398696
TV,4.8477468928,0.1714181332,0.5112729831
```

The final line specifies the translation vector. Note that it specifies TV as the atom symbol.

The following molecule specification could be used for a two-dimensional PBC calculation on a graphite sheet:

```
0 1
C                  0.000000    0.000000    0.000000
C                  0.000000    1.429118    0.000000
TV                 2.475315    0.000000    0.000000
TV                -1.219952    2.133447    0.000000
```

Here is the molecule specification that could be used for a three-dimensional PBC calculation on gallium arsenide:

```
0 1
 Ga                0.000000    0.000000    0.000000
 Ga                0.000000    2.825000    2.825000
 Ga                2.825000    0.000000    2.825000
 Ga                2.825000    2.825000    0.000000
 As                1.412500    1.412500    1.412500
 As                1.412500    4.237500    4.237500
 As                4.237500    1.412500    4.237500
 As                4.237500    4.237500    1.412500
 TV                5.650000    0.000000    0.000000
 TV                0.000000    5.650000    0.000000
 TV                0.000000    0.000000    5.650000
```

3.3 Basis Sets

Most methods require a basis set be specified; if no basis set keyword is included in the route section, then the STO-3G basis will be used. The exceptions consist of a few methods for which the basis set is defined as an integral part of the method; they are listed below:

- ◊ All semi-empirical methods, including ZIndo for excited states.
- ◊ All molecular mechanics methods.
- ◊ Compound model chemistries: all Gn, CBS and W1 methods.

Basis sets other than those listed here may also be input to the program using the ExtraBasis and Gen keywords. The ChkBasis keyword indicates that the basis set is to read from the checkpoint file (defined via the %Chk command). See the individual descriptions of these keywords for details.

The following basis sets are stored internally in the Gaussian 16 program (see references cited for full descriptions), listed below by their corresponding Gaussian 16 keyword (with two exceptions):

- ◊ STO-3G [26, 27]
- ◊ 3-21G [28–33]
- ◊ 6-21G [28, 29]
- ◊ 4-31G [34–37]
- ◊ 6-31G [34–43]
- ◊ 6-31G† [1]: Gaussian 16 also includes the 6-31G† and 6-31G‡ basis sets of George Petersson and coworkers, defined as part of various Complete Basis Set methods [44, 45]. These are accessed via the 6-31G(d') and 6-31G(d',p') keywords (respectively). Single or double diffuse functions may also be added, as can f functions: e.g., 6-31+G(d'f).
- ◊ 6-311G: Specifies the 6-311G basis for first-row atoms and the McLean-Chandler (12s,9p) $\rightarrow$ (621111, 52111) basis sets for second-row atoms [46, 47] (note that the basis sets for P, S, and Cl are those called negative ion basis sets by McLean and Chandler; these were deemed to give better results for neutral molecules as well), the basis set of Blaudeau and coworkers for Ca and K [41], the Wachters-Hay [48, 49] all electron basis set for the first transition row, using the scaling factors of Raghavachari and Trucks [50], and the 6-311G basis set of McGrath, Curtiss and coworkers for the other elements in the third row [40, 51, 52]. Note that Raghavachari and Trucks recommend both scaling and including diffuse functions when using the Wachters-Hay basis set for first transition row elements; the 6-311+G form must be specified to include the diffuse functions. MC-311G is a synonym for 6-311G.
- ◊ D95V: Dunning/Huzinaga valence double-zeta [53].
- ◊ D95: Dunning/Huzinaga full double zeta [53].
- ◊ SHC: D95V on first row, Goddard/Smedley ECP on second row [53, 54]. Also known as SEC.
- ◊ CEP-4G: Stevens/Basch/Krauss ECP minimal basis [55–57].
- ◊ CEP-31G: Stevens/Basch/Krauss ECP split valance [55–57].
- ◊ CEP-121G: Stevens/Basch/Krauss ECP triple-split basis [55–57].

  Note that there is only one CEP basis set defined beyond the second row, and all three keywords are equivalent for these atoms.
- ◊ LanL2MB: STO-3G [26, 27] on first row, Los Alamos ECP plus MBS on Na-La, Hf-Bi [58–60].
- ◊ LanL2DZ: D95V on first row [53], Los Alamos ECP plus DZ on Na-La, Hf-Bi [58–60].
- ◊ SDD: D95 up to Ar [53] and Stuttgart/Dresden ECPs on the remainder of the periodic table [61–85]. The SDD, SHF, SDF, MHF, MDF, MWB forms may be used to specify these basis sets/potentials within Gen basis input. Note that the number of core electrons must be specified following the form (e.g., MDF28 for the MDF potential replacing 28 core electrons). OldSDD requests the previous default.
- ◊ SDDAll: Selects Stuttgart potentials for Z > 2.
- ◊ cc-pVDZ, cc-pVTZ, cc-pVQZ, cc-pV5Z, cc-pV6Z: Dunning's correlation consistent basis sets [86–90] (double, triple, quadruple, quintuple-zeta and sextuple-zeta, respectively). These basis sets have had redundant functions removed and have been rotated [91] in order to increase computational efficiency.

  These basis sets include polarization functions by definition. The following table lists the valence polarization functions present for the various atoms included in these basis sets:

[1] 6-31G† is not a keyword of basis set in Gaussian. Don't confuse it with 6-31+G. – Noted by the editor.

Atoms	cc-pVDZ	cc-pVTZ	cc-pVQZ	cc-pV5Z	cc-pV6Z
H	2s,1p	3s,2p,1d	4s,3p,2d,1f	5s,4p,3d,2f,1g	6s,5p,4d,3f,2g,1h
He	2s,1p	3s,2p,1d	4s,3p,2d,1f	5s,4p,3d,2f,1g	*not available*
Li-Be	3s,2p,1d	4s,3p,2d,1f	5s,4p,3d,2f,1g	6s,5p,4d,3f,2g,1h	*not available*
B-Ne	3s,2p,1d	4s,3p,2d,1f	5s,4p,3d,2f,1g	6s,5p,4d,3f,2g,1h	7s,6p,5d,4f,3g,2h,1i
Na-Ar	4s,3p,1d	5s,4p,2d,1f	6s,5p,3d,2f,1g	7s,6p,4d,3f,2g,1h	*not available*
Ca	5s,4p,2d	6s,5p,3d,1f	7s,6p,4d,2f,1g	8s,7p,5d,3f,2g,1h	*not available*
Sc-Zn	6s,5p,3d,1f	7s,6p,4d,2f,1g	8s,7p,5d,3f,2g,1h	9s,8p,6d,4f,3g,2h,1i	*not available*
Ga-Kr	5s,4p,2d	6s,5p,3d,1f	7s,6p,4d,2f,1g	8s,7p,5d,3f,2g,1h	*not available*

These basis sets may be augmented with diffuse functions by adding the AUG- prefix to the basis set keyword (rather than using the + and ++ notation–see below).

◊ Basis sets of Ahlrichs and coworkers: the SV, SVP, TZV, TZVP keywords refer to the initial formations of the split valence and triple zeta basis sets from this group [92, 93]. The newer redefinitions of these basis sets in [94, 95] are requested with the keywords Def2SV, Def2SVP, Def2SVPP, Def2TZV, Def2TZVP, Def2TZVPP, Def2QZV, Def2QZVP, Def2QZVPP, and QZVP. Note that Def2SVPP corresponds to the "def2-SV(P)" basis set in [94]; all other names follow those in the paper with the hyphen removed.

◊ MIDI! of Truhlar and coworkers [96]. The MidiX keyword is used to request this basis set.

◊ EPR-II and EPR-III: The basis sets of Barone [97] which are optimized for the computation of hyperfine coupling constants by DFT methods (particularly B3LYP). EPR-II is a double zeta basis set with a single set of polarization functions and an enhanced s part: (6,1)/[4,1] for H and (10,5,1)/[6,2,1] for B to F. EPR-III is a triple-zeta basis set including diffuse functions, double d-polarizations and a single set of f-polarization functions. Also in this case the s-part is improved to better describe the nuclear region: (6,2)/[4,2] for H and (11,7,2,1)/[7,4,2,1] for B to F.

◊ UGBS: The universal Gaussian basis set of de Castro, Jorge and coworkers [98–106]. Additional polarization functions may be added by including a suffix to this keyword:

UGBS*n*P|V|O

where *n* is an integer indicating whether to add 1, 2 or 3 polarization functions for each function in the normal UGBS basis set. The second item is a code letter indicating which function should be augmented polarization functions: P adds them to all functions, V adds them to all valence functions, and O requests the scheme used in Gaussian 03 (see below). For example, the UGBS1P keyword requests this basis set with one additional polarization function to all orbitals, and UGBS2V adds two additional polarization function to all valence orbitals.

The O suffix adds the same functions as the UGBS*n*P keywords in Gaussian 03. UGBS1O adds a p function for each s, a d function for each p, and so on; UGBS2O adds a p and d function for each s, a d and f function for each p, and UGBS3O adds a p, d and f for each s, etc.

Diffuse functions may be added as usual with + or ++; the first of these may be specified as 2+ to add two diffuse functions for heavy atoms.

◊ MTSmall of Martin and de Oliveira, defined as part of their W1 method (see the W1U keyword) [107].

◊ The DGDZVP, DGDZVP2 and DGTZVP basis sets used in *DGauss* [108, 109].

◊ CBSB7: Selects the 6-311G(2d,d,p) basis set used by CBS-QB3 high accuracy energy method [110].

The notation specifies two additional d polarization functions on second rows atoms, one d function on first row atoms and a p function on hydrogens (note that this three-field polarization function syntax is *not* supported by Gaussian 16).

Adding Polarization and Diffuse Functions

Single first polarization functions can also be requested using the usual * or ** notation. Note that (d,p) and ** are synonymous – 6-31G** is equivalent to 6-31G(d,p), for example – and that the 3-21G* basis set has polarization functions on second row atoms only. The + and ++ diffuse functions [111] are available with some basis sets, as are multiple polarization functions [112]. The keyword syntax is best illustrated by example: 6-31+G(3df,2p) designates the 6-31G basis set supplemented by diffuse functions, 3 sets of d functions and one set of f functions on heavy atoms, and supplemented by 2 sets of p functions on hydrogens.

When the AUG- prefix is used to add diffuse functions to the cc-pV*Z basis sets, one diffuse function of each function type in use for a given atom is added [87, 88]. For example, the AUG-cc-pVTZ basis places one s, one d, and one p diffuse functions on hydrogen atoms, and one d, one p, one d, and one f diffuse functions on B through Ne and Al through Ar.

There are several options for augmenting the cc-pV*Z basis sets with diffuse functions:

◊ spAug-cc-pV*Z augments with s and p functions only, including s functions on H and He.

◊ dAug-cc-pV*Z augments with with 2 shells of each angular momentum instead of one.

◊ Truhlar's "calendar" basis set variations [113] are available. The naming of this series of basis sets comes from the fact that the cc-pV*Z basis sets with added polarization functions are known as the Aug-cc-pV*Z. Truhlar noted that "Aug" is also an abbreviation for the month of August in English, so he proposed new augmentation schemes for the cc-pV*Z basis sets, also named after months of the year. They are constructed by removing diffuse functions from the Aug basis sets. For example, the Jul-cc-pV*Z basis sets remove the diffuse function from H and He from Aug-cc-pV*Z. Jun-cc-pV*Z also removes the highest angular momentum diffuse function from all other atoms, May-cc-pV*Z removes the two highest angular momentum functions, and Apr-cc-pV*Z removes the three highest angular momentum functions.

  Nevertheless, by default, at least s and p diffuse functions are always included in these basis sets. This serves to avoid some inherent inconsistencies, but it differs from Truhlar and coworkers' original definitions. Use the forms TJul, TJun, and so on to specify the original versions where the limit is applied unconditionally: e.g., TMay-cc-ppVDZ includes only a diffuse s function on Cl but both diffuse s and p functions on Fe and Br, while May-cc-ppVDZ has diffuse s and p functions on all of these atoms.

Adding a single polarization function to 6-311G (i.e. 6-311G(d)) will result in one d function for first and second row atoms and one f function for first transition row atoms, since d functions are already present for the valence electrons in the latter. Similarly, adding a diffuse function to the 6-311G basis set will produce one s, one p, and one d diffuse functions for third-row atoms.

When a frozen core calculation is done using the D95 basis, both the occupied core orbitals and the corresponding virtual orbitals are frozen. Thus while a D95** calculation on water has 26 basis functions, and a 6-31G** calculation on the same system has 25 functions, there will be 24 orbitals used in a frozen core post-SCF calculation involving either basis set.

The following table lists polarization and diffuse function availability and the range of applicability for each built-in basis set in Gaussian 16:

Basis Set	Applies to	Polarization Functions	Diffuse Functions
3-21G	H-Xe		+
6-21G	H-Cl	* or **	
4-31G	H-Ne	* or **	
6-31G	H-Kr	through (3df,3pd)	+,++
6-311G	H-Kr	through (3df,3pd)	+,++
D95	H-Cl except Na and Mg	through (3df,3pd)	+,++
D95V	H-Ne	(d) or (d,p)	+,++
SHC	H-Cl	*	
CEP-4G	H-Rn	* (Li-Ar only)	
CEP-31G	H-Rn	* (Li-Ar only)	
CEP-121G	H-Rn	* (Li-Ar only)	
LanL2MB	H-La, Hf-Bi		
LanL2DZ	H, Li-La, Hf-Bi		
SDD, SDDAll	all but Fr and Ra		
cc-pVDZ	H-Ar, Ca-Kr	*included in definition*	added via AUG- prefix (H-Ar, Sc-Kr)
cc-pVTZ	H-Ar, Ca-Kr	*included in definition*	added via AUG- prefix (H-Ar, Sc-Kr)
cc-pVQZ	H-Ar, Ca-Kr	*included in definition*	added via AUG- prefix(H-Ar, Sc-Kr)
cc-pV5Z	H-Ar, Ca-Kr	*included in definition*	added via AUG- prefix (H-Na, Al-Ar, Sc-Kr)
cc-pV6Z	H, B-Ne	*included in definition*	added via AUG- prefix (H, B-O)
SV	H-Kr		
SVP	H-Kr	*included in definition*	
TZV and TZVP	H-Kr	*included in definition*	
QZVP and Def2	H-La, Hf-Rn	*included in definition*	
MidiX	H, C-F, S-Cl, I, Br	*included in definition*	
EPR-II, EPR-III	H, B, C, N, O, F	*included in definition*	
UGBS	H-Lr	UGBS(1,2,3)P	+,++,2+,2++
MTSmall	H-Ar		
DGDZVP	H-Xe		
DGDZVP2	H-F, Al-Ar, Sc-Zn		
DGTZVP	H, C-F, Al-Ar		
CBSB7	H-Kr	included in definition	+,++

STO-3G and 3-21G accept a * suffix, but this does not actually add any polarization functions.

3.3.1 Issues Arising from Pure vs. Cartesian Basis Functions

The following additional keywords are useful in conjunction with these basis set keywords:

◊ 5D and 6D: Use 5 or 6 d functions (pure vs. Cartesian d functions), respectively.

◊ 7F and 10F: Use 7 or 10 f functions (pure vs. Cartesian f functions), respectively. These keywords also apply to all higher functions (g and beyond).

Gaussian users should be aware of the following points concerning pure vs. Cartesian basis functions:

◊ All of the built-in basis sets use pure f functions. Most also use pure d functions; the exceptions are 3-21G, 6-21G, 4-31G, 6-31G, 6-31G†, 6-31G‡, CEP-31G, D95 and D95V. The preceding keywords may be used to override the default pure/Cartesian setting. Note that basis functions are generally converted to the other type automatically when necessary, for example, when a wavefunction is read from the checkpoint file for use in a calculation using a basis consisting of the other type [114].

◊ Within a job, all d functions must be 5D or 6D, and all f and higher functions must be pure or Cartesian.

◊ When using the ExtraBasis, Gen and GenECP keywords, the basis set explicitly specified in the route section always determines the default form of the basis functions (for Gen, these are 5D and 7F). For example, if you use a general basis set taking some functions from the 3-21G and 6-31G basis sets, pure functions will be used unless you explicitly specify 6D in the route section in addition to Gen. Similarly, if you add basis functions for a transition metal from the 6-311G(d) basis set via ExtraBasis to a job that specifies the 6-31G(d) basis set in the route section, Cartesian d functions will be used. Likewise, if you want to add basis functions for Xe from the 3-21G basis set to the 6-311 basis set via the ExtraBasis keyword, the Xe basis functions will be pure functions.

3.3.2 Density Fitting

Gaussian 16 provides the density fitting approximation for pure DFT calculations [115, 116]. This approach expands the density in a set of atom-centered functions when computing the Coulomb interaction instead of computing all of the two-electron integrals. It provides significant performance gains for pure DFT calculations on medium sized systems too small to take advantage of the linear scaling algorithms without a significant degradation in the accuracy of predicted structures, relative energies and molecular properties. Gaussian 16 can generate an appropriate fitting basis automatically from the AO basis, or you may select one of the built-in fitting sets.

The desired fitting basis set is specified as a third component of the model chemistry, as in this example:

```
# BLYP/TZVP/TZVPFit
```

Note that slashes must be used as the separator characters between the method, basis set, and fitting set when a density fitting basis set is specified.

The following fitting sets keywords are available in Gaussian 16:

◊ DGA1 and DGA2 [108, 109]. DGA1 is available for H through Xe, and DGA2 is available for H, He and B through Ne.

◊ SVPFit [117, 118] and Def2SV [94], corresponding to the SVP basis set.

◊ TZVPFit [117, 118] and Def2TZV [94], corresponding to the TZVP basis set.

◊ The W06 fitting set of Ahlrichs and coworkers [94, 95], designed for use with the Def2SVP, Def2TZV, and QZVP basis sets.

◊ Fit: Select the fitting set corresponding to the specified basis set. If there is no such fitting set, the result will produce an error.

◊ NoFit: Turn off fitting set use for this calculation. This keyword is used to override the DensityFit keyword present within the *Default.Route* file.

◊ Auto: Generate a fitting set automatically (see below).

Density fitting sets can be generated automatically from the AO primitives within the basis set. This is requested using the Auto fitting set keyword. The program automatically truncates the set at a reasonable angular momentum: the default is *Max(MaxTyp+1,2*MaxVal)*, where *MaxTyp* is the highest angular momentum in the AO basis, and *MaxVal* is the highest valence angular momentum. You can request that all generated functions be used with Auto=All, or request those up to a certain level with Auto=*N*, where *N* is the maximum angular momentum retained in the fitting functions. Finally, the PAuto form generates all products of AO functions on one center instead of just squares of the AO primitives, but this is typically more functions than are needed.

By default, no fitting set is used. Density fitting basis sets may be augmented with the ExtraDensityBasis keyword, defined in full with the Gen keyword, and optionally retrieved from the checkpoint file (use ChkBasis to do so). The options to the DensityFit keyword can be used to control some aspects of the fitting set used within calculations.

Density fitting can be made the default for jobs using pure DFT functionals by adding the DensityFit keyword to the route section (-#-) line in the *Default.Route* file. Fitting is faster than doing the Coulomb term exactly for systems up to several hundred atoms (depending on basis set), but is slower than exact Coulomb using linear scaling techniques (which are turned on automatically with exact Coulomb) for very large systems.

3.4 Constructing Z-Matrices

This section presents a brief overview of traditional Z-matrix descriptions of molecular systems. There are restrictions on the size of a Z-matrix: the maximum number of variables and the maximum number of atoms within a calculation. These are set consistently for a maximum of 250,000 real atoms (including ghost but not dummy atoms), and a maximum of 250,000 Z-matrix centers (atoms, ghost atoms, and dummy atoms).

Each line of a Z-matrix gives the internal coordinates for one of the atoms within the molecule. The most-used Z-matrix format uses the following syntax:

Element-label, atom 1, bond-length, atom 2, bond-angle, atom 3, dihedral-angle [format-code]

Although these examples use commas to separate items within a line, any valid separator may be used. *Element-label* is a character string consisting of either the chemical symbol for the atom or its atomic number. If the elemental symbol is used, it may be optionally followed by other alphanumeric characters to create an identifying label for that atom. A common practice is to follow the element name with a secondary identifying integer: C1, C2, etc.

Atom1, *atom2*, *atom3* are the labels for previously-specified atoms and are used to define the current atoms' position. Alternatively, the other atoms' line numbers within the molecule specification section may be used for the values of variables, where the charge and spin multiplicity line is line 0.

The position of the current atom is then specified by giving the length of the bond joining it to *atom1*, the angle formed by this bond and the bond joining *atom1* and *atom2*, and the dihedral angle formed by the plane containing *atom1*, *atom2* and *atom3* with the plane containing the current atom, *atom1* and *atom2*. Note that bond angles must be in the range $0^o < angle < 180^o$. Dihedral angles may take on any value.

The optional format-code parameter specifies the format of the Z-matrix input. For the syntax being described here, this code is always 0. This code is needed only when additional parameters follow the normal Z-matrix specification data, as in an ONIOM calculation.

As an initial example, consider hydrogen peroxide. A Z-matrix for this structure would be:

```
H
O 1 0.9
```

```
O 2 1.4 1 105.0
H 3 0.9 2 105.0 1 120.0
```

The first line of the Z-matrix simply specifies a hydrogen. The next line lists an oxygen atom and specifies the internuclear distance between it and the hydrogen as 0.9 Angstroms. The third line defines another oxygen with an O-O distance of 1.4 Angstroms (i.e., from atom 2, the other oxygen) and having an O-O-H angle (with atoms 2 and 1) of 105 degrees. The fourth and final line is the only one for which all three internal coordinates need be given. It defines the other hydrogen as bonded to the second oxygen with an H-O distance of 0.9 Angstroms, an H-O-O angle of 105 degrees and a H-O-O-H dihedral angle of 120 degrees.

Variables may be used to specify some or all of the values within the Z-matrix. Here is another version of the previous Z-matrix:

```
H
O 1 R1
O 2 R2 1 A
H 3 R1 2 A 1 D
  Variables:
R1 0.9
R2 1.4
A 105.0
D 120.0
```

Symmetry constraints on the molecule are reflected in the internal coordinates. The two H-O distances are specified by the same variable, as are the two H-O-O bond angles. When such a Z-matrix is used for a geometry optimization in internal coordinates (Opt=Z-matrix), the values of the variables will be optimized to locate the lowest energy structure. For a full optimization (FOpt), the variables are required to be linearly independent and include all degrees of freedom in the molecule. For a partial optimization (POpt), variables in a second section (often labeled Constants:) are held fixed in value while those in the first section are optimized:

```
  Variables:
R1 0.9
R2 1.4
A 105.0
  Constants:
D 120.0
```

See the examples in the discussion of the Opt keyword for more information about optimizations in internal coordinates.

Mixing Internal and Cartesian Coordinates

Cartesian coordinates are actually a special case of the Z-matrix, as in this example:

```
C   0.00   0.00   0.00
C   0.00   0.00   1.52
H   1.02   0.00  -0.39
H  -0.51  -0.88  -0.39
```

```
H  -0.51   0.88  -0.39
H  -1.02   0.00   1.92
H   0.51  -0.88   1.92
H   0.51   0.88   1.92
```

It is also possible to use both internal and Cartesian coordinates within the same Z-matrix, as in this example:

```
O 0 xo  0.  zo
C 0 0.  yc  0.
C 0 0. -yc  0.
N 0 xn  0.  0.
H 2 r1 3 a1 1  b1
H 2 r2 3 a2 1  b2
H 3 r1 2 a1 1 -b1
H 3 r2 2 a2 1 -b2
H 4 r3 2 a3 3  d3
  Variables:
xo -1.
zo  0.
yc  1.
xn  1.
r1 1.08
r2 1.08
r3 1.02
a1 125.
a2 125.
d3 160.
b1  90.
b2 -90.
```

This Z-matrix has several features worth noting:

◊ The variable names for the Cartesian coordinates are given symbolically in the same manner as for internal coordinate variables.

◊ The integer 0 after the atomic symbol indicates symbolic Cartesian coordinates to follow.

◊ Cartesian coordinates can be related by a sign change just as dihedral angles can.

Alternate Z-matrix Format

An alternative Z-matrix format allows nuclear positions to be specified using two bond angles rather than a bond angle and a dihedral angle. This is indicated by a 1 in an additional field following the second angle (this field defaults to 0, which indicates a dihedral angle as the third component):

```
C4 O1 0.9 C2 120.3 O2 180.0 0
C5 O1 1.0 C2 110.4 C4 105.4 1
C6 O1 R C2 A1 C3 A2 1
```

The first line uses a dihedral angle while the latter two use a second bond angle.

3.4.1 Dummy Atoms

This section will illustrate the use of dummy atoms within Z-matrices, which are represented by the pseudo atomic symbol X. The following example illustrates the use of a dummy atom to fix the three-fold axis in C_{3v} ammonia:

```
N
X 1 1.
H 1 nh 2 hnx
H 1 nh 2 hnx 3  120.0
H 1 nh 2 hnx 3 -120.0

nh 1.0
hnx 70.0
```

The position of the dummy on the axis is irrelevant, and the distance 1.0 used could have been replaced by any other positive number. **hnx** is the angle between an N-H bond and the threefold axis.

Here is a Z-matrix for oxirane:

```
X
C1  X halfcc
O   X      ox C1 90.
C2  X halfcc  O 90. C1 180.0
H1  C1     ch  X hcc  O  hcco
H2  C1     ch  X hcc  O -hcco
H3  C2     ch  X hcc  O  hcco
H4  C2     ch  X hcc  O -hcco

halfcc   0.75
ox       1.0
ch       1.08
hcc    130.0
hcco   130.0
```

This example illustrates two points. First, a dummy atom is placed at the center of the C-C bond to help constrain the **cco** triangle to be isosceles. **ox** is then the perpendicular distance from O to the C-C bond, and the angles **oxc** are held at 90 degrees. Second, some of the entries in the Z-matrix are represented by the negative of the dihedral angle variable **hcco**.

The following examples illustrate the use of dummy atoms for specifying linear bonds. Geometry optimizations in internal coordinates are unable to handle bond angles of 180 degrees which occur in linear molecular fragments, such as acetylene or the C_4 chain in butatriene. Difficulties may also be encountered in nearly linear situations such as ethynyl groups in asymmetrical molecules. These situations can be avoided by introducing dummy atoms along the angle bisector and using the half-angle as the variable or constant:

```
N
C 1 cn
X 2 1. 1 90.
H 2 ch 3 90. 1 180.

cn 1.20
ch 1.06
```

Similarly, in this Z-matrix intended for a geometry optimization, **half** represents half of the N-C-O angle which is expected to be close to linear. Note that a value of **half** less than 90 degrees corresponds to a cis arrangement:

```
N
C 1 cn
X 2 1. 1 half
O 2 co 3 half 1 180.0
H 4 oh 2  coh 3   0.0

cn 1.20
co 1.3
oh 1.0
half 80.0
coh 105.
```

3.4.2 Model Builder

The model builder is another facility within Gaussian for quickly specifying certain sorts of molecular systems [119]. It is requested with the ModelA or ModelB options, and it requires additional input in a separate section within the job file.

The basic input to the model builder is called a *short formula matrix*, a collection of lines, each of which defines an atom (by atomic symbol) and its connectivity, by up to six more entries. Each of these can be either an integer, which is the number of the line defining another explicitly specified atom to which the current atom is bonded, or an atomic symbol (e.g. H, F) to which the current atom is connected by a terminal bond, or a symbol for a terminal functional group which is bonded to the current atom. The functional groups currently available are OH, NH2, Me, Et, NPr, IPr, NBu, IBu, and TBu.

The short formula matrix also implicitly defines the rotational geometry about each bond in the following manner. Suppose atoms X and Y are explicitly specified. Then X will appear in row Y and Y will appear in row X. Let *I* be the atom to the right of X in row Y and *J* be the atom to the right of Y in row X. Then atoms *I* and *J* are put in the trans orientation about the X-Y bond. The short formula matrix may be followed by optional lines modifying the generated structure. There are zero or more of each of the following lines, which must be grouped together in the order given here:

AtomGeom,*I*,*Geom*:

Normally the local geometry about an atom is defined by the number and types of bond about the atom (e.g., carbon in methane is tetrahedral, in ethylene is trigonal, etc.). All bond angles at one center must be are equal. The AtomGeom line changes the value of the bonds at center *I*. *Geom* may be the angle as a floating point number, or one of the strings Tetr, Pyra, Trig, Bent, or Line.

BondRot,*I*,*J*,*K*,*L*,*Geom*:

This changes the orientations of the *I*-*J* and *K*-*L* bonds about the *J*-*K* bond. *Geom* is either the dihedral angle or one of the strings Cis (≥0), Trans (≥180), Gaup (≥+60), or Gaum (≥-60).

BondLen,*I*,*J*,*NewLen*:

This sets the length of the *I*-*J* bond to *NewLen* (a floating point value).

The model builder can only build structures with atoms in their normal valencies. If a radical is desired, its extra valence can be effectively "tied down" using dummy atoms, which are specified by a minus sign before the atomic symbol (e.g., -H). Only terminal atoms can be dummy atoms.

The two available models (A and B) differ in that model A takes into account the type (single, double, triple, etc.) of a bond in assigning bond lengths, while model B bond lengths depend only on the types of the atoms involved. Model B is available for all atoms from H to Cl except He and Ne. If Model A is requested and an atom is used for which no Model A bond length is defined, the appropriate Model B bond length is used instead.

3.5 Multistep Jobs

Multiple Gaussian jobs may be combined within a single input file. The input for each successive job is separated from that of the preceding job step by a line of the form:

```
--Link1--
```

Here is an example input file containing two job steps:

```
%Chk=freq
# HF/6-31G(d) Freq

Frequencies at STP
```

Molecule specification

```
--Link1--
%Chk=freq
%NoSave
# HF/6-31G(d) Geom=Check Guess=Read Freq=(ReadFC,ReadIsotopes)

Frequencies at 300 K
```

charge and spin

```
300.0  2.0
```

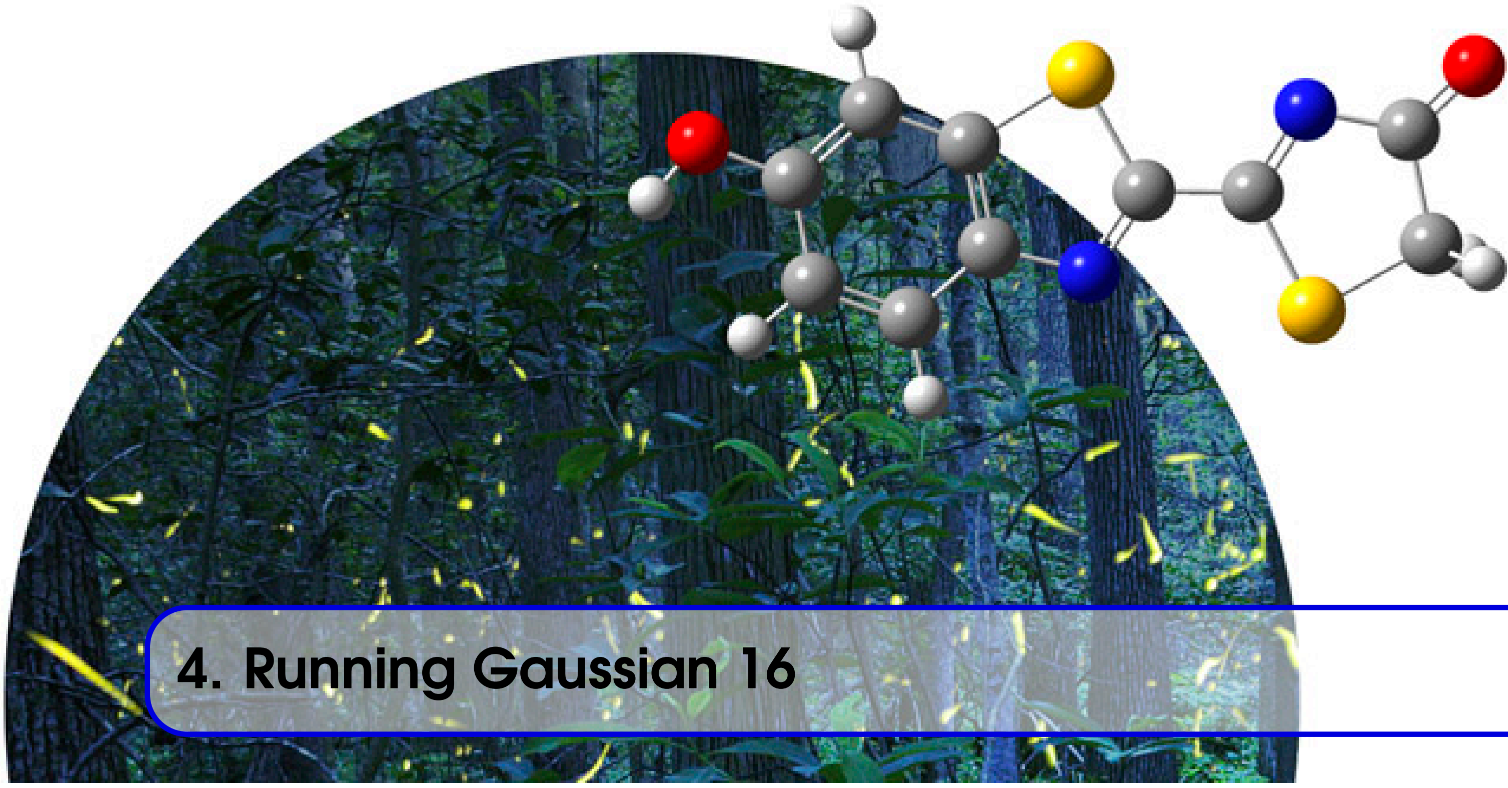
Isotope specifications

This input file computes vibrational frequencies and performs thermochemical analysis at two different temperatures and pressures: first at 298.15 K and 1 atmosphere, and then again at 300 K and 2 atmospheres. Note that a blank line must precede the --Link1-- line.

3.6 Section Ordering

Section	Keywords	Final blank line?
Link 0 commands	% commands	no
Route Section (# lines)	all	yes
Extra Overlays	ExtraOverlays	yes
Title section	all except Geom=AllCheck	yes
Molecule specification	all except Geom=AllCheck	yes
Connectivity specifications	Geom=Connect or ModConnect	yes
Alterations to frozen atoms	Geom=ReadOpt	yes
Modifications to coordinates	Opt=ModRedundant	yes
2nd title and molecule specification	Opt=QST2 or QST3	yes for both
Connectivity specs. for 2nd set of coordinates	Geom=Connect or ModConnect and Opt=QST2 or QST3	yes
2nd Alterations to frozen atoms	Geom=ReadOpt	yes
Modifications to 2nd set of coordinates	Opt=QST2 or QST3	yes
3rd title and initial TS structure	Opt=QST3	yes for both
Connectivity specs. for 3rd set of coords.	Geom=Connect or ModConnect Opt=(ModRedun, QST3)	yes
3rd Alterations to frozen atoms	Geom=ReadOpt	yes
Modifications to 3rd set of coordinates	Opt=(ModRedun, QST3)	yes
PDB secondary structure information	automatic if residue info in molecule specification	yes
Atomic masses	ReadIsotopes option	yes
External	External=($\cdots$,ReadInput)	yes
Molecular Mechanics parameters	HardFirst, SoftFirst, SoftOnly, Modify options	yes
Frequency of interest	CPHF=RdFreq or Freq=ROA	yes
Background charge distribution	Charge	yes
BOMD/ADMP input (1 or more sections)	ADMP and BOMD required input and ReadVelocity, ReadMWVelocity options	yes
PCM input	SCRF=(ExternalIteration,Read) and (SCRF=ONIOMPCM=A,Read)	yes
Coordinates for IRC table	IRC(Report=Read)	yes
Harmonic constraints	Geom=ReadHarmonic	yes
Semi-empirical parameters (Gaussian format)	Input option and Both options	yes
Semi-empirical parameters (MOPAC format)	MOPACExternal and Both options	yes
Basis set specification	Gen, GenECP, ExtraBasis	yes
Basis set alterations	Massage	yes
Finite field coefficients	Field=Read	yes
ECP specification	Pseudo=Cards, GenECP	yes
Density fitting basis set specification	ExtraDensityBasis	yes
PCM solvation model input	SCRF=Read	yes
DFTB parameters	DFTB	yes
Source for initial guess	Guess=Input	yes

Symmetry types to combine	Guess=LowSymm	no
Orbital specifications (separate α & β)	Guess=Cards	yes
Orbital alterations (separate α & β)	Guess=Alter	yes
Orbital reordering (separate α & β)	Guess=Permute	yes
# Orbitals/GVB pair	GVB	no
Weights for CAS state averaging	CAS=StateAverage	yes
States of interest for spin orbit coupling	CASSCF=SpinOrbit	no
Orbital freezing information	ReadWindow options	yes
EPT orbitals to refine	EPT=ReadOrbitals	yes
Atoms list for spin-spin coupling constants	NMR=ReadAtoms	yes
Alternate atomic radii	Pop=ReadRadii or ReadAtRadii	yes
Antechamber output file	Pop=MK IOp(6/50=1)	yes
Data for electrostatic properties	Prop=Read or Opt	yes
NBO input	Pop=NBORead or Pop=NBO6Read	no
Harmonic Normal Mode selection	Freq=SelectNormalModes	yes
Hindered Rotor input	Freq=ReadHindered	yes
Anharmonic Normal Mode selection	Freq=SelectAnharmonicModes	yes
Input for Anharmonic	Freq=ReadAnharmonic	yes
Input for FCHT	Freq=ReadFCHT	yes
Atom list for Pickett file generation	Output=Pickett	no
ACID output filename	NMR=CGST IOp(10/93=1)	yes
PROAIMS output filename	Output=WFN	no
Matrix element filename	Output=MatrixElement or Output=Raw	yes



4. Running Gaussian 16

4.1 Preliminaries

4.1.1 System Requirements

◊ The Gaussian directories will require about 2-3 GB of disk space for the executables, depending on the computer system.

◊ The default memory allocation in Gaussian 16 is 800 MB. The large fixed dimensions in the program necessitate a swap space size of 1 - 2 GB. Of course, additional swap space will be required if more memory is requested in a job by using the %Mem Link 0 command, or via the -M- command in the *Default.Route* file. These requirements are for *each* simultaneously executing job.

◊ Refer to the platform list which comes with the CD. The most recent version of this document can always be found at `www.gaussian.com/g16/g16_plat.pdf`.

4.1.2 Configuring the Gaussian Execution Environment

Gaussian locates executables and creates scratch files in directories specified by several environment variables. The user is responsible for defining two of them:

◊ g16root: Indicates the directory where the g16 subdirectory resides (i.e., the directory above it).

◊ GAUSS_SCRDIR: Indicates the directory which should be used for scratch files.

4.1.3 Initialization Files

The Gaussian system includes initialization files to set up the user environment for running the program. These files are:

`$g16root/g16/bsd/g16.login`	*C shell*
`$g16root/g16/bsd/g16.profile`	*Bourne shell*

It is customary to include lines like the following within the .login or .profile file for Gaussian users:

.login files:

```
setenv g16root /location
source $g16root/g16/bsd/g16.login
setenv GAUSS_SCRDIR /path
```

.profile files:

```
export g16root=/location
. $g16root/g16/bsd/g16.profile
export GAUSS_SCRDIR=/location
```

Once things are set up correctly, the g16 command is used to execute Gaussian 16.

4.1.4 Environment Variables Reference

The environment variables created by g16.login and g16.profile include:

◊ GAUSS_EXEDIR: Specifies the directories in which the Gaussian images are stored. By default it includes the main directory $g16root/g16 and several alternate directories.

◊ GAUSS_ARCHDIR: Specifies the directory in which the main site-wide archive file is kept, and into which temporary archive files should be placed if the main archive is unavailable. It defaults to $g16root/g16/arch if unset.

◊ G16BASIS: The directory which contains files specifying the standard Gaussian internally stored basis sets, as well as some additional basis sets in the form of general basis set input. This environment variable is provided for convenience and is designed for use with the @ include mechanism.

◊ Gaussian defaults may be set via environment variables of the form GAUSS_*x*DEF. See the documentation for the *Default.Route* file for details.

◊ Network/cluster parallel calculations using Linda may also use the GAUSS_LFLAGS environment variable to pass options to the Linda process. See Parallel Jobs for details.

4.2 Running under UNIX

Once all input and resource specifications are prepared, you are ready to run the program. Gaussian 16 may be run interactively using one of two command styles:

g16 *job-name*

g16 *<input-file >output-file*

In the first form, the program reads input from *job-name*.gjf and writes its output to *job-name*.log. When *job-name* is not specified, the program reads from standard input and writes to standard output, and these can be redirected or piped in the usual UNIX fashion. Either form of command can be forced in the background in the same manner as any shell command using the ampersand.

4.2.1 Scripts and Gaussian

Scripts designed to run Gaussian 16 may also be created in several ways. First, g16 commands like those above may be included in a shell script. Secondly, actual Gaussian input may be included in the script using the « construct:

```
#!/bin/sh
```

```
g16 <<END >water.log
%Chk=water
# RHF/6-31G(d)

water energy

0  1
O
H  1  1.0
H  1  1.0  2  120.0

END
echo "Job done."
```

All lines preceding the string following the « symbols are taken as input to the g16 command.

Finally, loops may be created to run several Gaussian jobs in succession. For example, the following script runs all of the Gaussian input files specified as its command line arguments, and it maintains a log of its activities in the file Status:

```
#!/bin/sh

echo "Current Job Status:" > Status
for file in $argv; do
   nam=`echo $file | sed 's/\..*$//'`
   echo "Starting file $file at `date`" >> Status
   g16 < $file > $nam.log
   echo "$file Done with status $status" >> Status
done
echo "All Done." >> Status
```

The following more complex script creates Gaussian input files on-the-fly from the partial input in the files given as the script's command line arguments. The latter are lacking full route sections; their route sections consist of simply a # sign or a # line containing special keywords needed for that molecular system, but no method, basis set, or calculation type.

The script creates a two-step job for each partial input file – a Hartree-Fock optimization followed by an MP2 single point energy calculation – consisting of both the literal commands included in the script and the contents of each file specified at script execution time. It includes the latter by exploiting the Gaussian 16 include file mechanism:

```
#!/bin/sh

echo "Current Job Status:" > Status
for file in $argv; do
echo "Starting file $file at `date`" >> Status
nam=`echo $file | sed 's/\..*$//'`
g16 <<END> $nam.log
%Chk=$nam
# HF/6-31G(d) FOpt
@$file/N

--Link1--
```

```
%Chk=$nam
%NoSave
# MP2/6-31+G(d,p) SP Guess=Read Geom=AllCheck
END
echo "$file Done with status $status" >> Status
done # end of for...do
echo "All Done." >> Status
```

4.2.2 Batch Execution with NQS

Gaussian may be run using the NQS batch facility on those UNIX systems that support it. The subg16 command, defined in the initialization files, submits an input file to a batch queue. It has the following syntax:
subg16 *queue-name job-name* [`-scrdir` *dir1*] [`-exedir` *dir2*] [`-p` *n*]

The two required parameters are the queue and job names. Input is taken from *job-name*.gjf and output goes to *job-name*.log, just as for interactive runs. The NQS log file is sent to *job-name*.batch-log.

The optional parameters -scrdir and -exedir are used to override the default scratch and executable directories, respectively. Any other parameters are taken to be NQS options. In particular, -p *n* can be used to set the priority within the queue to *n*. This is priority for initiation (1 being lowest), and does not affect the run-time priority.

To submit an NQS job from an interactive session, a file like the following should be created (with filename *name*.job):

```
# QSUB -r name -o name.out -eo
# QSUB -lt 2000 -lT 2100
# QSUB -lm 34mw -lM 34mw
g16 <name.gjf
```

where *name* should be replaced with a name that is appropriate to your calculation. The first line names the running job, names the output file, and causes errors to be included in the output file. The time parameters are different to allow addition of job control for cleanup, (for example, archiving the checkpoint file in the event that the job exceeds its time limit). The memory parameters are used both for initial scheduling of your job for execution and by the program to determine dynamic memory use.

This job would then be submitted by issuing the command,
$ **qsub** *name***.job**
and the output would be placed in your current working directory.

4.3 Scratch Files

Gaussian uses several scratch files in the course of its computation. They include:

◊ The Checkpoint file: *name*.chk
◊ The Read-Write file: *name*.rwf
◊ The Two-Electron Integral file: *name*.int (empty by default)
◊ The Two-Electron Integral Derivative file: *name*.d2e (empty by default)
◊ The Scratch file: *name*.skr

By default, scratch files are deleted at the end of a successful run. However, you may wish to save the checkpoint file for later use in another Gaussian job, for use by a visualization program, to restart a failed job, and so on. This may be accomplished by *naming* the checkpoint file, providing an explicit name and/or location for it, via a %Chk command within the Gaussian input file. Here is an example:

```
%Chk=water
```

This command, which is placed at the very beginning of the input file – before the route section – gives the checkpoint file the name water.chk, overriding the usual generated name and causing the file to be saved at job conclusion. In this case, the file will reside in the current directory. However, a command like this one will specify an alternate directory location as well as filename:

```
%Chk=/chem/scratch2/water
```

While scratch files are deleted automatically for successful jobs, they are not deleted when a job is killed externally or otherwise terminates abnormally. Consequently, leftover files may accumulate in the scratch directory. An easy method for avoiding excessive clutter is to have all users share a common scratch directory and to have that scratch directory cleared at system boot time by adding an rm command to the appropriate system boot script (e.g., /etc/rc or one of the files under /etc/rc.d/rc3.d). If the NQS batch system is in use, clearing the scratch directory should also be done before NQS is started, ensuring that no jobs are using the directory when it is cleared.

If disk space in the scratch directory is limited, but space is available elsewhere on the system, you may want to split the various scratch files among several disk locations. The following commands allow you to specify the names and locations of the other scratch files:

%RWF=*path*	*Read-Write file*
%Int=*path*	*Integral file*
%D2E=*path*	*Integral Derivative file*

In general, the read-write file is by far the largest, and so it is the one for which an alternate location is most often specified.

4.3.1 Splitting Scratch Files Across Disks

An alternate syntax is provided for splitting the read-write file, the Integral file, and/or the Integral Derivative file among two or more disks (or file systems). Here is the syntax for the %RWF command:

%RWF=*loc1,size1,loc2,size2*, $\cdots$

where each *loc* is a directory location or a file pathname, and each *size* is the maximum size for the file segment at that location. Gaussian will automatically generate unique filenames for any *loc* which specifies a directory only. On UNIX systems, directory specifications (without filenames) must include a terminal slash.

By default, the sizes are in units of 8-byte words. This value may also be followed by KB, MB, GB, TB, KW, MW, GW or TW (without intervening spaces) to specify units of kilo-, mega-, giga-, or tera-bytes or words. Note that 1 MB = 1024^2 bytes = 1,048,576 bytes (not 1,000,000 bytes).

A value of -1 for any size parameter indicates that any and all available space may be used, and a value of 0 says to use the current size of an existing segment. -1 is useful only for the last file specified, for which it is the default.

For example, the following directive splits the read-write file across three disks:

```
%RWF=/dalton/s0/,4GB,/scratch/,3GB,/temp/s0/my_job,-1
```

The maximum sizes for the file segments are 4 GB, 3 GB, and unlimited, respectively. Gaussian will generate names for the first two segments, and the third will be given the name my_job. Note that the directory specifications include terminal slashes.

Due to limitations in current UNIX implementations, -1 should be used with caution, as it will attempt to extend a file segment beyond all remaining disk capacity on these systems; using it will also have the side effect of keeping any additional file segments included in the list from ever being used.

Gaussian 16 can address single scratch files of up to 16 GB under 32-bit operating systems. There is no need to split scratch files into 2 GB files. The 16 GB total scratch space limit is inherent in 32-bit integers, however, and splitting the scratch file will not overcome it.

4.3.2 Saving and Deleting Scratch Files

By default, unnamed scratch files are deleted at the end of the Gaussian run, and named files are saved. The %NoSave command may be used to change this default behavior. When this directive is included in an input file, named scratch files whose directives appear in the input file *before* %NoSave will be deleted at the end of a run (as well as all unnamed scratch files). However, if the % directive naming the file appears *after* the %NoSave directive, the file will be retained. For example, these commands specify a name for the checkpoint file, and an alternate name and directory location for the read-write file, and cause only the checkpoint file to be saved at the conclusion of the Gaussian job:

```
%RWF=/chem/scratch2/water    Files to be deleted go here.
%NoSave
%Chk=water                   Files to be saved go here.
```

Note that all files are saved when a job terminates abnormally.

4.4 Memory Use

The %Mem command controls the amount of dynamic memory to be used by Gaussian. By default, 800 MB (100MW) are used. This can be changed to a different value by specify a number followed by KB, MB, GB, TB, KW, MW, GW or TW (without intervening spaces) to specify units of kilo-, mega-, giga-, or tera-bytes or words. The default units are megawords.

For example, the following command also sets the amount of dynamic memory to 1 GB:

```
%Mem=1GB
```

Even larger allocations may be needed for very large direct SCF calculations, at least $3N^2$ words, where N is the number of basis functions.

Warning: Requesting more memory than the amount of physical memory actually available on a computer system will lead to very poor performance.

If Gaussian is being used on a machine with limited physical memory, so that the default amount is not available, the default algorithms as well as the default memory allocation will be set appropriately during installation.

4.5 Parallel Jobs

4.5.1 Shared-Memory Multiprocessor Parallel Execution

Gaussian defaults to execution on only a single processor. If your computer system has multiple processors/cores, and parallel processing is supported in your version of Gaussian, you may the specific CPUs on which to run with the %CPU link 0 command. For example, the following specifies that the program should run on the first 5 cores of a hexacore system (reserving one core for other use):

```
%CPU=0,1,2,3,4
```

The node list can also be specified as a range (e.g., 0-5). Ranges can also be followed by a suffix of the form /*n*, which says to use every nth processor in the range (e.g., /2 specifies every second processor/core).

The older %NProcShared link 0 command can be used to specify the total number of processors on which to execute (leaving the selection of processors to the operating system). Clearly, the number of processors requested should not exceed the number of processors available, or a substantial decrease in performance will result.

4.5.2 Cluster/Network Parallel Execution

Parallel jobs can run across discrete systems in a LAN or nodes within a cluster using the Linda parallel execution environment. HF, CIS=Direct, and DFT calculations are Linda parallel, including energies, optimizations, and frequencies. TDDFT energies and gradients and MP2 energies and gradients are also Linda parallel. Portions of MP2 frequency and CCSD calculations are Linda parallel, but others are only SMP-parallel.

For a Linda parallel job to execute successfully, the following conditions must be true:

◊ You have already executed the appropriate Gaussian 16 initialization file ($g16root/g16/bsd/g16.login or $g16root/g16/bsd/g16.profile). Test this by running a serial Gaussian 16 calculation on the master node.
◊ The directory $g16root/g16 is accessible on all nodes.
◊ The LD_LIBRARY_PATH environment variable is set to locate the Linda shared libraries.
◊ Local scratch space is available on each node if needed (via the GAUSS_SCRDIR environment variable).
◊ All nodes on which the program will run are trusted by the current host. You should be able to login remotely with the rsh or ssh command without having to give a password to each of them. Contact your system administrator about configuring security for the nodes in the network.

Each of these nodes needs to have some access to the Gaussian 16 directory tree. The recommended configuration is to have G16 on each system that will be used for the parallel job. Note that the Linda binaries need to have the same path on each machine. If this is not feasible, the G16 tree can be made accessible via an NFS-mounted disk which is mounted in an identical location on all nodes.

For MP2 calculations, each node must also have some local disk where Gaussian 16 can put temporary files. This is defined as usual via the GAUSS_SCRDIR environment variable, which should be set in the .cshrc or .profile for your account on each node.

The %LindaWorkers directive is used to specify computers where Linda worker processes should run. It has the following syntax:

%LindaWorkers=*node1*[:*n1*][,*node2*[:*n2*]] [· · ·]

This lists the TCP node name for each node to use. By default, one Linda worker is started on each node, but the optional value allows this to be varied. A worker is always started on the node where the job is started (the master node) whether or not it appears in the node list. For example, the following directive causes a network parallel job to be run across the specified 5 nodes. Nodes *hamlet* and *ophelia* will each run two worker processes.

```
%LindaWorkers=hamlet:2,ophelia:2,laertes,horatio,lear
```

By default, Linda uses rsh to communicate between nodes. You can use ssh instead by including the following Link 0 option within the job file:

```
%UseSSH
```

Finally, you can also create a configuration file on the master node named .tsnet.config which contains the following line:

```
Tsnet.Node.lindarsharg: ssh
```

This will cause ssh to be used instead. This file can be placed in your home directory or in the directory from which you launch calculations. The %UseRSH directive is also supported; its purpose is to specify the use of rsh when ssh has been made the default in *Default.Route* (see below).

Note that in all cases passwordless ssh logins must already be configured from the master to all worker nodes.

A few Linda options that are sometime useful are:

`-v`	*Display verbose messages*
`-vv`	*Display very verbose messages*

You can use the GAUSS_LFLAGS environment variable to set them. For example, one could turn on very verbose Linda output using:

```
$ export GAUSS_LFLAGS="-vv"
```

The verbose option may also be set by including the %DebugLinda link 0 command.

There are many other Linda options but most of them are not used by Gaussian. The -opt *option* form can be used in GAUSS_LFLAGS to invoke any valid .tsnet.config file directive. Note that Gaussian 16/Linda does not use the native Linda resources *minworker* and *maxworker*.

4.5.3 Combining Shared-Memory and Network/Cluster Parallelism

The following link 0 commands start a four-way parallel worker on hosts *norway* and *italy* and two such worker processes on *spain*:

```
%CPU=0-3
%LindaWorkers=norway,italy,spain:2
```

These link 0 commands are appropriate when *norway* and *italy* are 4 processor/core computers, and *spain* is an 8 processor/core computer.

It is always best to use SMP-parallelism within nodes and Linda only between nodes. For example on a cluster of 4 nodes, each with a dual quad-core EM64T, one should use the following (rather than using more than one Linda worker per node):

```
%CPUs=0-7
%LindaWorkers=node1,node2,node3,node4
```

4.5.4 Starting and Monitoring Parallel Gaussian 16 Jobs

Once the proper setup is completed, the g16 command is used as usual to initiate a parallel Gaussian 16 job:

$ **g16 input.gjf &**

In the case of distributed parallel calculations, Gaussian 16 will start the master and worker processes as needed.

Linda parallel links have filenames of the form *.exel. Using the top or other commands on worker nodes will let you see l*xxx*.exel when it starts.

The relevant measure of performance for a parallel calculation is the elapsed or wall clock time. The easiest way to check this is to use an external monitor like time, times, or timex, e.g.

$ **times g16 input.gjf &**

This command will report elapsed, CPU and system times for the Gaussian job. Note that the last two are relevant only on the master node, and similar amounts of CPU and system time are expended on each node. The parallel speedup is the ratio of the elapsed time running a serial job and the elapsed time for the parallel job.

4.6 Using GPUs

Gaussian 16 can use NVIDIA K40, K80 and P100 GPUs under Linux (the latter starting with Rev. B.01). Earlier GPUs do not have the computational capabilities or memory size to run the algorithms in Gaussian 16.

4.6.1 Allocating Memory for Jobs

Allocating sufficient amounts of memory to jobs is even more important when using GPUs than for CPUs, since larger batches of work must be done at the same time in order to use the GPUs efficiently. The K40 and K80 units can have up to 16 GB of memory. Typically, most of this should be made available to Gaussian. Giving Gaussian 8-9 GB works well when there is 12 GB total on each GPU; similarly, allocating Gaussian 11-12 GB is appropriate for a 16 GB GPU. In addition, at least an equal amount of memory must be available for each CPU thread which is controlling a GPU.

4.6.2 About Control CPUs

When using GPUs, each GPU must be controlled by a specific CPU. The controlling CPU should be as physically close as possible to the GPU it is controlling. GPUs cannot share controlling CPUs. *Note that CPUs used as GPU controllers cannot be used as compute nodes as well.*

The hardware arrangement on a system with GPUs can be checked using the **nvidia-smi** utility. For example, this output is for a machine with two 16-core Haswell CPU chips and four K80 boards, each of which has two GPUs:

	GPU0	GPU1	GPU2	GPU3	GPU4	GPU5	GPU6	GPU7	CPU	Affinity
GPU0	X	PIX	SOC	SOC	SOC	SOC	SOC	SOC	0-15	*cores on first chip*
GPU1	PIX	X	SOC	SOC	SOC	SOC	SOC	SOC	0-15	
GPU2	SOC	SOC	X	PIX	PHB	PHB	PHB	PHB	16-31	*cores on second chip*
GPU3	SOC	SOC	PIX	X	PHB	PHB	PHB	PHB	16-31	
GPU4	SOC	SOC	PHB	PHB	X	PIX	PXB	PXB	16-31	

```
GPU5   SOC   SOC   PHB   PHB   PIX     X   PXB   PXB  16-31
GPU6   SOC   SOC   PHB   PHB   PXB   PXB     X   PIX  16-31
GPU7   SOC   SOC   PHB   PHB   PXB   PXB   PIX     X  16-31
```

The important part of this output is the CPU affinity. This example shows that GPUs 0 and 1 (on the first K80 card) are connected to the CPUs on chip 0 while GPUs 2-7 (on the other two K80 cards) are connected to the CPUs on chip 1.

4.6.3 Specifying GPUs & Control CPUs for a Gaussian Job

The GPUs to use for a calculation and their controlling CPUs are specified with the %GPUCPU Link 0 command. This command takes one parameter:

%GPUCPU=*gpu-list*=*control-cpus*

where gpu-list is a comma-separated list of GPU numbers, possibly including numerical ranges (e.g., **0-4,6**), and control-cpus is a similarly-formatted list of controlling CPU numbers. The corresponding items in the two lists are the GPU and its controlling CPU.

For example, on a 32-processor system with 6 GPUs, a job which uses all the CPUs – 26 CPUs doing parts of the computation and 6 CPUs used for controlling GPUs – would use the following Link 0 commands:

`%CPU=0-31` *Control CPUs are included in this list.*

`%GPUCPU=0,1,2,3,4,5=0,1,16,17,18,19`

These command state that CPUs 0-31 will be used in the job (although not all of them will be used for computation). GPUs 0 through 5 will be used, with GPU0 controlled by CPU 0, GPU1 controlled by CPU 1, GPU2 controlled by CPU 16, GPU3 controlled by CPU 17, and so on. Note that the controlling CPUs are included in %CPU.

In the preceding example, the GPU and CPU lists could be expressed more tersely as:

```
%CPU=0-31
%GPUCPU=0-5=0-1,16-19
```

Normally one uses consecutive processors in the obvious way, but things can be associated differently in special cases. For example, suppose the same machine already had a job using 6 CPUs, running with %CPU=16-21. Then, in order to use the other 26 CPUs with 6 controlling GPUs, you would specify:

```
%CPU=0-15,22-31
%GPUCPU=0-5=0-1,22-25
```

This job would use a total of 26 processors, employing 20 of them for computation, along with the six GPUs controlled by CPUs 0, 1, 22, 23, 24 and 25 (respectively).

In **[REV B]**, the lists of CPUs and GPUs are both sorted and then matched up. This ensures that the the lowest numbered threads are executed on CPUs that have GPUs. Doing so ensures that if a part of a calculation has to reduce the number of processors used (i.e., because of memory limitations), it will preferentially use/retain the threads with GPUs (since it removes threads in reverse order).

4.6.4 GPUs and Overall Job Performance

GPUs are effective for larger molecules when doing DFT energies, gradients and frequencies (for both ground and excited states), but they are not effective for small jobs. They are also not used effectively by post-SCF calculations such as MP2 or CCSD.

Each GPU is several times faster than a CPU. However, on modern machines, there are typically many more CPUs than GPUs. The best performance comes fromm using all the CPUs as well as the GPUs.

In some circumstances, the potential speedup from GPUs can be limited because many CPUs are also used effectively by Gaussian 16. For example, if the GPU is 5x faster than a CPU, then the speedup of using the GPU versus the CPU alone would be 5x. However, the potential speedup resulting from using GPUs on a larger computer with 32 CPUs and 8 GPUs is 2x:
Without GPUs: 32*1 = 32
With GPUs: (24*1) + (8*5) = 64 *Remember that control CPUs are not used for computation.*
Speedup: 64/32 = 2

Allocation of memory

GPUs can have up to 16 GB of memory. One typically tries to make most of this available to Gaussian. Be aware that there must be at least an equal amount of memory given to the CPU thread running each GPU as is allocated for computation. Using 8-9 GB works well on a 12 GB GPU, or 11-12 GB on a 16 GB GPU (reserving some memory for the system). Since Gaussian gives equal shares of memory to each thread, this means that the total memory allocated should be the number of threads times the memory required to use a GPU efficiently. For example, when using 4 CPUs and 2 GPUs each with 16 GB of memory, you should use 4 $\times$ 12 GB of total memory. For example:

```
%Mem=48GB
%CPU=0-3
%GPUCPU=0-1=0,2
```

You will need to analyze the characteristics of your own environment carefully when making decisions about which processors and GPUs to use and how much memory to allocate.

4.6.5 GPUs in a Cluster

GPUs on nodes in a cluster can be used. Since the %CPU and %GPUCPU specifications are applied to each node in the cluster, the nodes must have identical configurations (number of GPUs and their affinity to CPUs); since most clusters are collections of identical nodes, this is not usually a problem.

4.7 g16 Command Line Options

The g16 command accepts the following options on all platforms:

Option	**Description**	**Example**
-r=*list*	Route section default keyword list	`-r="Int=SuperFine Freq=HPModes"`
-m=*value*	Memory amount for Gaussian jobs	`-m=200GB`
-c=*list*	Processor/core list for multiprocessor parallel jobs	`-c="0-31"`
-g=*list*	GPUs=Cores list for GPU parallel jobs	`-g="0-8=0-6,31,32"`
-s=*command*	Command to use for starting workers for network parallel jobs: one of ssh or rsh	`-s=ssh`
-w=*list*	List of hostnames for network parallel jobs.	`-w="apple,banana,cherry"`

-p	Number of processors/cores for multiprocessor parallel jobs. Deprecated; use -c.	
-l	Number of nodes for network parallel jobs. Deprecated; use -w.	
-x	Complete route for the job (the route is not read from the input file).	`-x="# b3lyp/6-31G(d) opt freq"`
-y	Checkpoint file.	`-y="$HOME/project3/gfp.chk"`
-fchk	Generate the specified formatted checkpoint file as well as a binary one, and update it each time the checkpoint file is updated.	`-fchk=/plots/job8.fchk`
-z	Read-write file.	`-z=/scratch2/fox7.rwf`
-ic	Existing checkpoint file from which to read input.	`-ic=job6`
-im	Matrix element file from which to read input.	`-im=fox7.dat`
-im4	Matrix element file using 4-byte integers from which to read input.	`-im4=fox12`
-ir	Raw matrix element file from which to read input.	`-ir=job6`
-im4	Raw matrix element file using 4-byte integers from which to read input.	`-ir4=job27`
-oc	Output checkpoint file. Generally redundant with -y.	
-om	Output matrix element file.	`-om=job16new_mat`
-om4	Output matrix element file using 4-byte integers.	`-om4=job32mat`
-or	Output raw matrix element file.	`-or=job7_raw`
-or4	Output raw matrix element file using 4-byte integers.	`-or4=proteins_rawdat`

Note that the quotation marks are often required around option values to avoid modification by the shell.

4.8 Setting Defaults

Default values for the Link 0 commands mentioned here can be set in the *Default.Route* file. They can also be set via environment variables and on the command line. The following table lists these equivalences:

Link 0	Default.Route	Option	Env. Variable	Description
%CPU	-C-	-c=*list*	GAUSS_CDEF	Processor/core list for multiprocessor parallel jobs.
%GPUCPU	-G-	-g=*list*	GAUSS_GDEF	GPUs=Cores list for GPU parallel jobs.
%Mem	-M-	-m=*value*	GAUSS_MDEF	Memory amount for Gaussian jobs.
%NProcShared	-P-	-p=*n*	GAUSS_PDEF	Number of processors/cores for multiprocessor parallel jobs. Superseded in most cases by %CPU.
%UseRSH	-S- rsh	-s=rsh	GAUSS_SDEF=rsh	Use rsh to start workers for network parallel jobs.
%UseSSH	-S- ssh	-s=ssh	GAUSS_SDEF=ssh	Use ssh to start workers for network parallel jobs.
%LindaWorkers	-W-	-w=*list*	GAUSS_WDEF	List of hostnames for network parallel jobs.
%DebugLinda			GAUSS_LFLAGS=-v	Enable verbose Linda output.

Quotation marks will be required for command line argument values whenever special characters are included within them. Note that the environment variable specifications above use bash syntax.

Full documentation for all Link 0 command is available in Link0.

4.9 The Default.Route File

Depending on the characteristics of a particular computer system, it is sometimes necessary for performance reasons to override some of the defaults built into the program. This can be done by creating a customization file. On UNIX-based systems, this file is named *Default.Route*, residing in $g16root/g16. Under Windows, the Gaussian defaults file is Default.Rou, and it is located in the Gaussian 16W scratch subdirectory (e.g., C:\G16W\scratch). The format of the file is the same on all computer systems.

The following subsections describe the types of information which can be supplied in the defaults file.

4.9.1 Description

Route Defaults

These parameters are introduced by -#- and have the same form as normal route section commands. For example, this line will set the default SCF algorithm to the conventional (non-direct) algorithm:

```
-#- SCF=Conventional
```

There may be more than one -#- line in the file.

Commands listed in *Default.Route* change only the defaults; they are overridden by anything specified in the route section of an input file. Thus, if the *Default.Route* contains:

```
-#- MP2=NoDirect
```

and the route section contains the MP2 keyword, then the conventional MP2 algorithm will be used. However, if the route section contains the MP2=Direct keyword, then the direct algorithm will be used.

The directive -R- is a synonym for -#-.

Maximum Disk Usage

You will usually want to specify the amount of scratch disk space available via the MaxDisk keyword in the *Default.Route* file. For example, the following line sets MaxDisk to 2 GB:

```
-#- MaxDisk=2GB
```

The amount of available disk space is given in 8-byte words by default. This value may also be followed by KB, MB, GB, TB, KW, MW, GW, or TW (without intervening spaces) to specify units of kilo-, mega-, giga-, or tera-bytes/words.

The scratch disk space is set unlimited by default: i.e., MaxDisk=-1. In other words, it is assumed that enough disk space is available to perform a given calculation with no redundant work. Thus, specifying the amount of available memory and disk is by far the most important way of optimizing performance for your calculations. Doing so allows the program to decide between the various available algorithms, selecting the optimal one for your particular system configuration. Keep in mind that the more disk space available, the faster the evaluation, especially for MP2.

Memory Defaults

Gaussian jobs which unwisely use excessive memory can cause severe difficulties on the system. The memory directive -M- enforces a default dynamic memory limit. For example, the following line sets default memory use to 3GB:

```
-M- 3GB
```

The amount of available memory is given in 8-byte words (default); this value may also be followed by KB, MB, GB, TB, KW, MW, GW, or TW (without intervening spaces) to specify units of kilo-, mega-, giga-, or tera-bytes or words. The default memory size is 800 MB (100 MW). Note that this limit can be overridden with the %Mem Link 0 command.

The memory directive -U- sets default memory to use in utilities such as formchk and freqchk.

Organization and Host Names

The host name may be set with -H-; the value specifies the host name to be used in archive entries generated by Gaussian. The default is the current hostname.

The organization/site name may be set with -O-; the value specifies the site name to be used in archive entries generated by Gaussian. There is no default.

User Defaults Files

Gaussian users may set their own defaults by creating their own *Default.Route* file. Gaussian checks the current working directory for a file of this name when a job is initiated. Settings in the local file take precedence over those in the site-wide file, and options specified in the route section of the job take precedence over both of them.

Command Line Options & Environment Variable Equivalents

All of these directives can also be set via environment variables or UNIX/Linux command line arguments. The environment variable GAUSS_*x*DEF provides a command line equivalent to -X- in the *Default.Route* file. Similarly, the command line argument to g16 -*x*="value" specifies the same setting. For example, all of the following have the equivalent effect:

```
-M- 4GB                          in Default.Route
export GAUSS_MDEF=4GB
g16 -m="4GB"  ···
```

Order of Priority

The order of priority is:

- ◊ command line argument
- ◊ environment variable
- ◊ *Default.Route* setting
- ◊ internal program default

Default.Route Limitations

Not all route section keywords are honored in the *Default.Route* file. In general, the rule is that options which are specific to a specific job type are ignored. Thus, Integral=SuperFine, which changes the default integration grid, will be honored, while Freq=Raman, which specifies a setting for a specific job type, will be ignored.

4.9.2 Parallel Job Directives

Parallel Execution on Shared Memory Multiprocessors

Gaussian defaults to execution on only a single processor. If your computer system has multiple processors/cores, and parallel processing is supported in your version of Gaussian, you may the specific CPUs on which to run with the -C- directive. For example, the following directive specifies that the program should run on the first 5 cores of a hexacore system (reserving one core for other use):

```
-C- 0,1,2,3,4
```

The node list can also be specified as a range (e.g., 0-5). Ranges can also be followed by a suffix of the form /*n*, which says to use every *n*th processor in the range (e.g., /2 specifies every second processor/core).

The older -P- directive can be used to specify the total number of processors on which to execute (leaving the selection of processors to the operating system). Clearly, the number of processors requested should not exceed the number of processors available, or a substantial decrease in performance will result.

Network/Cluster Parallel Execution

You can specify the list of Linda workers in *Default.Route* via the -W- directive:

```
-W- dalton,lavoisier:2,priestley,agassiz:3,curie
```

This example will use the specified five nodes for parallel execution, placing 2 worker processes on *lavoisier*, 3 workers on *agassiz*, and one worker on each of the other systems. If the master node – the node where the job is started – is not one of these systems, a worker will also run on that system (making a total of six nodes).

This directive corresponds to – and can be overridden by – the Link 0 command %LindaWorkers.

The -S- directive specifies which program to use to start worker processes. It takes either rsh (the default) or ssh as its parameter.

Combining SMP and Network Parallelism

When -W- is combined with -C- or -P-, then an SMP parallel worker process is started on each node in the node list (or more than one such process when multiple workers are specified for that node). In other words, these individual worker processes run in parallel according to the SMP-related settings. If the settings are inappropriate for a listed computer, then the work will fail on that node. For example, the following directives will start workers as multiprocessors jobs on the specified systems:

```
-W- alpha,beta,gamma
-C- 0,1,2,3
```

However, if system *beta* is a dual-core workstation, that worker process will fail as there is no processor 2 or 3 present on that computer. This mechanism is designed primarily for cluster environments with uniform nodes.

Note that it is the *Default.Route* file on the master which is relevant; *Default.Route* files that may be present on worker systems are not used.

Parallel Execution on GPUs

The -G- directive sets the default list of GPUs to use. When using GPUs it is essential to have each GPU controlled by a specific CPU and much preferable if the CPU is physically close to the GPU it is controlling. The list uses the syntax *gpu#(s)=core#(s)* to indicate which GPUs to use and the core to which each is pinned.

For example, the following directive tells Gaussian to execute on GPU0 through GPU3:

```
-G- 0,1,2,3=0,1,16,17
```

GPU0 and GPU1 are pinned to cores 0 and 1, while GPU2 and GPU3 are pinned to cores 16 and 17 (respectively).

Ranges of GPUs and/or cores can be used in this directive. For example, the following is equivalent to the preceding:

```
-G- 0-3=0-1,16-17
```

4.9.3 formchk Directives

The -F- directive sets the default options for the formchk utility. By default formchk produces a formatted checkpoint file that is backwardly compatible with all previous versions of Gaussian.

The parameter to -F- can be -3 or -2 to create a formatted checkpoint file of the corresponding version (version 3 is the default). The parameter can also contain -c, which causes the molecular mechanics atom types to appear in the formatted checkpoint file as strings rather than integers.

The memory directive -U- sets default memory to use in utilities such as formchk and freqchk.

4.9.4 Examples

For a dual-core workstation with 8 GB memory and 1 TB of disk, the default algorithms and memory allocation are fine. However, MaxDisk and the default CPUs for execution need to be specified:

```
-C- 0,1
-#- MaxDisk=200GB
```

On a server with 4 octa-core processors and 256 GB of memory designed to be used for large jobs, 8 cores should be used by default: the first 8 odd numbered cores starting with core 0. Also, more memory should be given to each job (2 GB/core):

```
-M- 16GB
-C- 0-16/2
-#- MaxDisk=500GB
```

On some older Intel processors (Nehalem and before), there is not enough memory bandwidth to keep all the CPUs on a chip busy, and it is often preferable to use half the CPUs, each with twice as much memory as if all were used. For example, on such a machine with 4 12-core chips and 128 GBytes of memory, with CPUs 0-11 on the first chip, 12-23 on the second, etc., it is better to run using 24 processors (6 on each chip) and give them 72/24=3GByte memory each, rather than use all 48 with only 1.5GBytes of memory each. The appropriate directive would be

```
-M- 72GB
-C- 0-47/2
```

/2 says to use every other core: i.e., cores 0, 2, 4, 6, 8 and 10 on chip 0, cores 12, 14, 16, 18, 20 and 22 on chip 1, and so on.

4.9.5 Directive List

Default.Route	Link 0	Option	Env. Var.	Description
-#-		-r	GAUSS_RDEF	Route section keyword list.
-C-	%CPU	-c	GAUSS_CDEF	Processor/core list for multiprocessor parallel jobs.
-F-			GAUSS_FDEF	Options for the formchk utility.
-G-	%GPUCPU	-g	GAUSS_GDEF	GPUs=Cores list for GPU parallel jobs.
-H-			GAUSS_HDEF	Computer hostname.
-M-	%Mem	-m	GAUSS_MDEF	Memory amount for Gaussian jobs.
-O-			GAUSS_ODEF	Organization (site) name.
-P-	%NProcShared	-p	GAUSS_PDEF	Number of processors/cores for multiprocessor parallel jobs. Superseded in most cases by -C-.
-R-		-r	GAUSS_RDEF	Route section keyword list.
-S-	%UseSSH	-s	GAUSS_SDEF	Program to start workers for network parallel jobs: rsh or ssh.
-U-			GAUSS_UDEF	Memory amount for utilities.
-W-	%LindaWorkers	-w	GAUSS_WDEF	List of hostnames for network parallel jobs.

4.10 Running Gaussian Test Jobs

An extensive set of test jobs for Gaussian are provided, along with their corresponding output files. The input files are found in directory $g16root/g16/tests/com. One or more subdirectories of $g16root/g16/tests, named for computer architecture types (e.g., ia64), contain reference output for the various test jobs (gzipped log files). A script file which runs ranges of test jobs automatically is also provided (described below).

If you build the program from source code under a supported architecture and operating system and using the specified compiler, libraries and other software (if any), we recommend that you run a few of the test jobs to verify that the program has been built correctly. However, it is not necessary to run the entire test suite.

You do not need to run any test jobs for binary distributions.

Test job input files have names of the form test*nnnn*.com. Tests 1, 28, 94, 155, 194, 296, and 302 cover a range of Gaussian capabilities. Note that some test jobs are intended for fast hardware and are quite expensive on smaller, slower computer systems. The file $g16root/g16/tests/tests.idx lists what each test job does.

Rename Existing Default.Route File Before Running Test Jobs

If you choose to run some or all of the Gaussian test jobs, you will need to make sure that they run with the program's built-in default settings. Therefore, you'll need to rename both the site-wide *Default.Route* file (located in the $g16root/g16 directory) as well as any individual version of the defaults file that you may have prior to running any test job. Note that certain settings in this file can cause some test jobs to fail.

Examples

The script submit.csh can be used to run test jobs. It accepts two parameters: the numbers of the first and last jobs to run (by default, all of the tests are run). Note that you should run the test jobs from a separate directory to prevent them from clobbering the reference output.

The following commands illustrate the recommended procedure for running a test job, using the directory /chem/newtests as the test job execution area and test job 28 as an example:

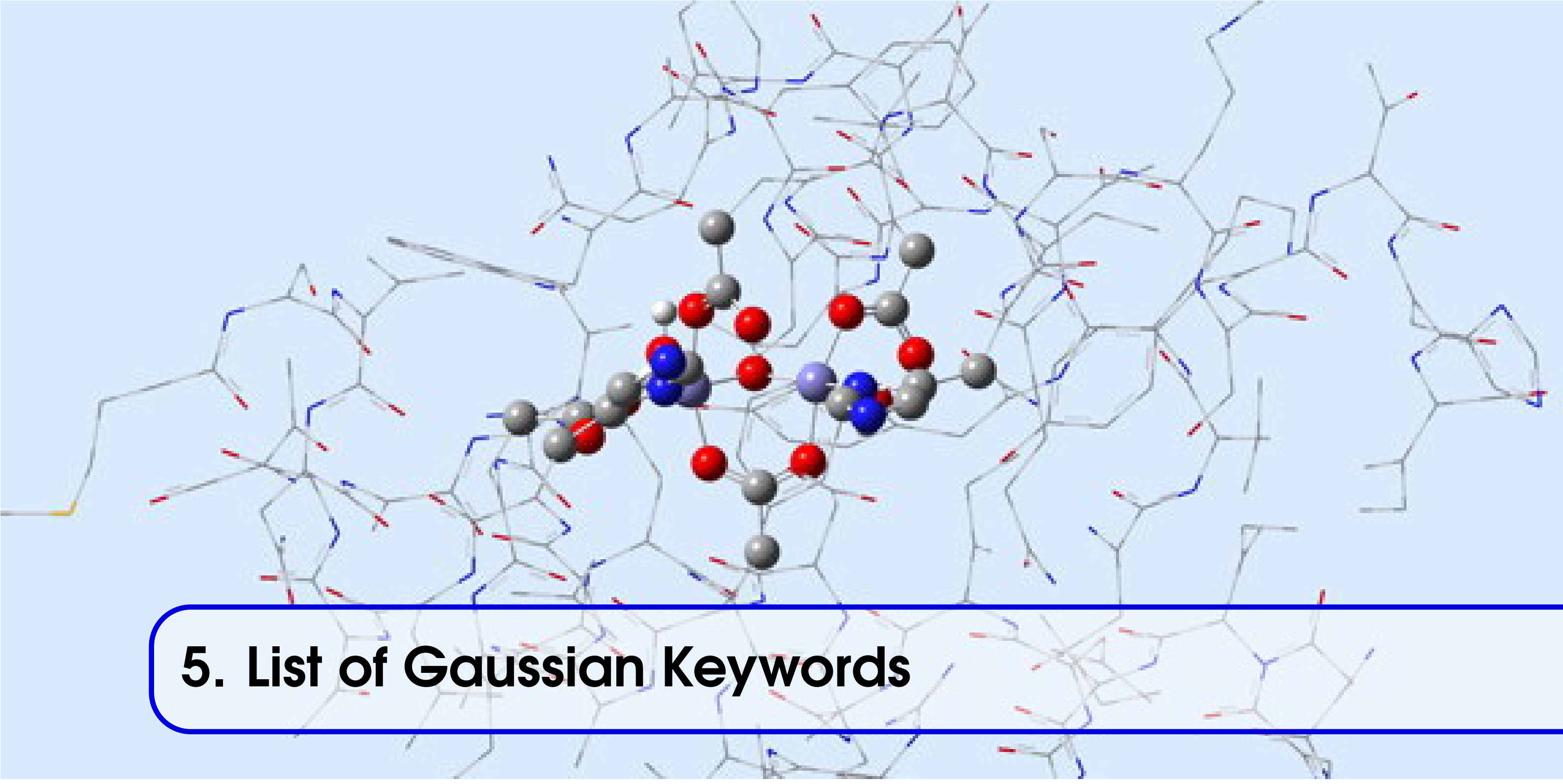
```
$ mkdir /chem/newtests; cd /chem/newtests
$ ln -s $g16root/g16/tests/com .
$ mkdir `gau-machine`
$ $g16root/g16/tests/submit.csh m  n &
```

After each test job finishes, verify that it completed successfully. Then, compare its current output with the reference output using the d1 script. For example:

```
$ $g16root/g16/tests/d1 m  n
```

The d1 script filters out insignificant differences from the output files for the specified test jobs and pipes the remaining output through more. The differences that appear should be limited to non-substantive items.

Part Two



5. List of Gaussian Keywords

5.1 Link 0 Commands

This section lists all Link 0 commands, which are optional and precede the route section.

Link 0 commands may be up to 500 characters in length.

%Mem=*N*

Sets the amount of dynamic memory used to *N* 8-byte words (default). This value may also be followed by KB, MB, GB, TB, KW, MW, GW or TW (without intervening spaces) to specify units of kilo-, mega-, giga-, or tera-bytes or words. The default memory size is 800 MB. A different default value can be set in *Default.Route* with the -M- directive.

%Chk=*file*

Locates and names the checkpoint file.

%OldChk=*file*

The contents of the checkpoint file specified by %OldChk are copied to the checkpoint file of the current job step at the start of the job step. This allows data to be picked up from a previous calculation without destroying anything on the checkpoint file from it.

%SChk=*file*

Saves a copy of the checkpoint file when the job step starts. One can usually do the same thing with %OldChk and %Chk but sometimes it might be convenient to save the intermediate results this way.

%RWF=*file*

Locates and names a single, unified read-write file (old-style syntax).

%RWF=*loc1,size1,loc2,size2,*···

An alternate syntax is provided for splitting the read-write file among two or more disks (or file systems). Each location is followed by a maximum size for the file segment at that location. The size of each file segment is given in 8-byte words (default). This value may also be followed by KB, MB, GB, TB, KW, MW, GW or TW (without intervening spaces) to specify units of kilo-, mega-, giga- or tera-bytes or words. A value of -1 for any size parameter indicates that any and all available space may be used, and a value of 0 indicates that an existing segment should retain its current size. The locations may be either directory locations, or full pathnames. Note that directory specifications *must* include terminal slashes (on UNIX systems).

%OldMatrix=*matfile*

Copy the data on the unformatted binary matrix element file *matfile* to the active checkpoint file at the start of the job step. This directive is similar %OldChk but takes the data from the specified file. In Rev. B.01 and later, the file name may be followed by the keyword i4lab to specify that the file uses 4-byte integers: e.g., %OldMatrix=(myfile,i4lab).

%OldRawMatrix=*matfile*

Copy the data on the raw binary matrix element file *matfile* to the active checkpoint file at the start of the job step. This directive is similar %OldChk but takes the data from the specified file. In Rev. B.01 and later, the file name may be followed by the keyword i4lab to specify that the file uses 4-byte integers: e.g., %OldRaw=(myfile,i4lab).

%Int=*spec*

Locates and names the two-electron integral file(s). The *spec* parameter may take on either of the forms used for the read-write file.

%D2E=spec

Locates and names the two-electron integral derivative file(s). The *spec* parameter may take on either of the forms used for the read-write file.

%KJob L*N* [*M*]

Tells the program to stop the run after the M^{th} occurrence of Link *N*. For example, %KJob L502 2 will cause the run to terminate after Link 502 has been run for the second time. *M* may be omitted; it defaults to 1.

%Save

Causes Link 0 to save scratch files at the end of the run. By default, all non-specified scratch files are deleted and all named scratch files are saved when the run completes successfully.

%ErrorSave

Causes Link 0 to delete scratch files at the end of a successful run, including any files that were named explicitly *preceding* this directive. In other words, if a file is named before %ErrorSave is encountered, it will not be saved. However, if the directive naming the file appears after the %ErrorSave directive, the file will be retained. If the job ends abnormally, all files are saved. %NoSave is a synonym for this directive.

If both %Save and %ErrorSave are specified, then the one appearing latest in the input file takes precedence.

%Subst L*N* *dir*

Tells Link 0 to take the executable (.exe file) for a link from an alternate directory. For example %SUBST L913 /user/chem will cause /user/chem/l913.exe to be run instead of the default executable (in $g16root). The directory specification should be in the usual format for the machine involved. Only the directory can be specified; the file name must have the standard form of l*nnnn*.exe, where *nnnn* is the Link number.

5.1.1 Parallel Job Directives

%CPU=*proc-list*

This directive contains a list of processor/core numbers for shared memory parallel processing, with multiple items are separated by commas. This directive is designed to replace the earlier %NProcShared and %NProc directives.

A default value can be set in *Default.Route* with the -C- directive.

%NProcShared=*N*

Requests that the job use up to *N* processors/cores on shared memory parallel execution on SMP multiprocessor computers. For most purposes, this directive is superceded by %CPU. The value of %NProcShared overrides the -P- directive in the *Default.Route* file.

%GPUCPU=*gpu-list=core-list*

This sets the default list of GPUs to use. When using GPUs it is essential to have each GPU controlled by a specific CPU and much preferable if the CPU is physically close to the GPU it is controlling. The list preceding the equals sign indicates the GPUs to use for the calculation. Corresponding items in the second list specify the core to which each CPU is pinned. For example, the following directive tells Gaussian to execute on GPU0 through GPU3:

```
%GPUCPU=0,1,2,3=0,1,16,17
```

GPU0 and GPU1 are pinned to cores 0 and 1, while GPU2 and GPU3 are pinned to cores 16 and 17 (respectively).

Ranges of GPUs and/or cores can be used in this directive. For example, the following is equivalent to the preceding:

```
%GPUCPU=0-3=0-1,16-17
```

A default value for %GPUCPU can be set in *Default.Route* with the -G- directive.

%LindaWorkers=*node1*[:*n1*] [,*node2*[:*n2*]] ···

This lists the TCP node name for each node to use. By default, one Linda worker is started on each node, but the optional value allows this to be varied. A worker is always started on the node where the job is started (the master node) whether or not it appears in the node list. A default value can be set in *Default.Route* with the -W- directive.

%LindaWorkers may be combined with %NProcShared. In this case, one or more parallel worker processes will be run on each node (the number still determined by the values in %LindaWorkers). The value to %NProcShared specifies the number of SMP processors/cores to use on each system in the worker node list.

Do not use the obsolete %NProcLinda directive. Gaussian will compute the total number of Linda workers based on the %LindaWorkers input.

%UseSSH

Start Linda workers using ssh rather than rsh. %UseRSH specifies the use of rsh, and it is the default. A different default value can be set in *Default.Route* with the -S- directive.

%DebugLinda

Report details concerning the starting and stopping of Linda workers.

5.1.2 Examples

These commands specify a name for the checkpoint file, and an alternate name and directory location for the read-write file, and cause only the checkpoint file to be saved at the conclusion of the Gaussian job:

```
%RWF=/chem/scratch2/water                 Files to be deleted go here.
%NoSave
%Chk=water                                Files to be saved go here.
```

The following directive causes a network parallel job to be run across the specified 5 nodes. Nodes *hamlet* and *ophelia* will each run two worker processes.

```
%LindaWorkers=hamlet:2,ophelia:2,laertes,horatio,lear
```

The following directives specify that a parallel job will be executed on hosts *norway*, *italy* and *spain*. Nodes *norway* and *italy* will each run one 4-way SMP parallel worker, and *spain* will run two such workers:

```
NProcShared=4                             Specifies four-way SMP parallelism.
%LindaWorkers=norway,italy,spain:2
```

These directives make sense when *norway* and *italy* are 4 processor/core computers, and *spain* is an 8 processor/core computer.

5.2

The route section of a Gaussian job is initiated by a pound sign (#) as the first non-blank character of a line. The remainder of the section is in free-field format. For most jobs, all of the information can be placed on this first line, but overflow to other lines (which may but need not begin with a # symbol) is permissible. The route section must be terminated by a blank line.

If no keywords are present in the route section, the calculation defaults to HF/STO-3G SP.

5.2.1 Alternate Forms

#N

Normal print level; this is the default.

#P

Additional output is generated. This includes messages at the beginning and end of each link giving assorted machine-dependent information (including execution timing data), as well as convergence information in the SCF.

#T

Terse output: output is reduced to essential information and results.

5.3 ADMP

This keyword requests a classical trajectory calculation [120–123] using the Atom Centered Density Matrix Propagation molecular dynamics model [124–126]. This method provides equivalent functionality to Born-Oppenheimer molecular dynamics (see the BOMD keyword) at considerably reduced computational cost [126].

ADMP belongs to the extended Lagrangian approach to molecular dynamics using Gaussian basis functions and propagating the density matrix. The best known method of this type is Car-Parrinello (CP) molecular dynamics [127], in which the Kohn-Sham molecular orbitals, ψ_i, are chosen as the dynamical variables to represent the electronic degrees of freedom in the system. CP calculations are usually carried out in a plane wave basis (although Gaussian orbitals are sometimes added as an adjunct [128–130]). Unlike plane wave CP, it is not necessary to use pseudopotentials on hydrogen or to use deuterium rather than hydrogen in the dynamics. Fictitious masses for the electronic degrees of freedom are set automatically [126] and can be small enough that thermostats are not required for good energy conservation.

ADMP can be performed with semi-empirical, HF, and pure and hybrid DFT models (see the *Availability* tab for more details). It can be applied to molecules, clusters and periodic systems. PBC calculations use only the Γ point (i.e., no K-integration).

5.3.1 Input

Although most jobs will not require it, ADMP calculations can accept some input. The first section below provides the optional initial Cartesian velocities for the ReadVelocity and ReadMWVelocity options.

Initial velocity for atom 1: x y z	*Optional initial Cartesian velocities*
Initial velocity for atom 2: x y z	*(ReadVelocity and ReadMWVelocity options)*
...	
Initial velocity for atom N: x y z	
...	

First, the initial velocity for each atom is read if the ReadVelocity or ReadMWVelocity option is included. Each initial velocity is specified as a Cartesian velocity in atomic units (Bohr/sec) or as a mass-weighed Cartesian velocity (in amu$^{1/2}$*Bohr/sec), respectively. One complete set of velocities is read for each requested trajectory computation.

This information (if present) may be immediately followed by the Morse parameters for each diatomic product (no intervening blank line):

Atom1, Atom2, E_0*, Len,* D_e*,* B_e
...

Terminate entire trajectory input subsection with a blank line.

The Morse parameter data is used to determine the vibrational excitation of diatomic fragments using the EBK quantization rules. It consists of the atomic symbols for the two atoms, the bond length between them (*Len*, in Angstroms), the energy at that distance (E_0 in Hartrees), and the Morse curve parameters D_e (Hartrees) and B_e (Angstroms^{-1}). This input subsection is terminated by a blank line.

5.3.2 Options

MaxPoints=*n*

Specifies the maximum number of steps that may be taken in each trajectory (the default is 50). If a trajectory job is restarted, the maximum number of steps will default to the number specified in the original calculation.

Lowdin

Use the Löwdin basis for the orthonormal set. The alternative is Cholesky, which uses the Cholesky basis and is the default.

NKE=*N*

Set the initial nuclear kinetic energy to N microHartrees. NuclearKineticEnergy is a synonym for this option. The default is 100000 (corresponding to 0.1 Hartree).

DKE=*N*

Set the initial density kinetic energy to N microHartrees. DensityKineticEnergy is a synonym for this option.

ElectronMass=*N*

Set the fictitious electron mass to $|N/10000|$ amu (the default is N=1000, resulting in a fictitious mass of 0.1 amu). EMass is a synonym for this option. If N<0, then uniform scaling is used for all basis functions. By default, core functions are weighted more heavily than valence functions.

FullSCF

Do the dynamics with converged SCF results at each point.

ReadVelocity

Read initial Cartesian velocities from the input stream. Note that the velocities must have the same symmetry orientation as the molecule. This option suppresses the fifth-order anharmonicity correction.

ReadMWVelocity

Read initial mass-weighted Cartesian velocities from the input stream. Note that the velocities must have the same symmetry orientation as the molecule. This option suppresses the fifth-order anharmonicity correction.

StepSize=*n*

Sets the step size in dynamics to *n**0.0001 femtoseconds. The default is 1000 (a step size of 0.1 femtoseconds).

BandGap

Whether to diagonalize the Fock matrix in order to report the band gap at each step. The default is NoBandGap.

Restart

Restart an ADMP calculation from the checkpoint file. Note that options set in the original job will continue to be in effect and cannot be modified.

ReadIsotopes

This option allows you to specify alternatives to the default temperature, pressure, frequency scale factor and/or isotopes – 298.15 K, 1 atmosphere, no scaling, and the most abundant isotopes (respectively). It is useful when you want to rerun an analysis using different parameters from the data in a checkpoint file. Be aware, however, that all of these can be specified in the route section (Temperature, Pressure and Scale keywords) and molecule specification (the Iso parameter), as in this example:

```
#T Method/6-31G(d) JobType Temperature=300.0 ···

...

0 1
C(Iso=13)
...
```

ReadIsotopes input has the following format:

temp pressure [scale]	*Values must be real numbers.*
isotope mass for atom 1	
isotope mass for atom 2	
...	
isotope mass for atom n	

Where *temp*, *pressure*, and *scale* are the desired temperature, pressure, and an optional scale factor for frequency data when used for thermochemical analysis (the default is unscaled). The remaining lines hold the isotope masses for the various atoms in the molecule, arranged in the same order as they appeared in the molecule specification section. If integers are used to specify the atomic masses, the program will automatically use the corresponding actual exact isotopic mass (e.g., 18 specifies ^{18}O, and Gaussian uses the value 17.99916).

5.3.3 Availability

Semi-empirical, HF, and DFT methods.

5.3.4 Related Keywords

BOMD

5.3.5 Examples

The following sample ADMP input file will calculate a trajectory for H_2CO dissociating to H_2 + CO, starting at the transition state:

```
# B3LYP/6-31G(d) ADMP Geom=Crowd
Dissociation of H2CO --> H2 + CO
0 1
C
O 1 r1
H 1 r2 2 a
H 1 r3 3 b 2 180.
r1 1.15275608
r2 1.74415774
r3 1.09413376
a 114.81897892
b 49.08562961
```

Final blank line

At the beginning of an ADMP calculation, the parameters used for the job are displayed in the output:

```
TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ
-------------------------------------------------------------------
INPUT DATA FOR L121
-------------------------------------------------------------------

General parameters:
Maximum Steps                 =      50
Random Number Generator Seed =   398465
Time Step                     =   0.10000 femtosec
Ficticious electronic mass    =   0.10000 amu
MW individual basis funct.    = True
Initial nuclear kin. energy   =   0.10000 hartree
Initial electr. kin. energy   =   0.00000 hartree
  Initial electr. KE scheme   =   0
Multitime step - NDtrC        =       1
Multitime step - NDtrP        =       1
No Thermostats chosen to control nuclear temperature

Integration parameters:

Follow Rxn Path (DVV)         = False
Constraint Scheme             =      10
Projection of angular mom.    = True
Rotate density with nuclei    = True
-------------------------------------------------------------------
```

```
TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ
```

The molecular coordinates and velocities appear at the beginning of each trajectory step (some output digits are truncated here):

```
Cartesian coordinates:
I= 1 X= -1.1971360D-01 Y=  0.0000000D+00 Z= -1.0478570D+00
I= 2 X= -1.1971360D-01 Y=  0.0000000D+00 Z=  1.1305362D+00
I= 3 X=  2.8718451D+00 Y=  0.0000000D+00 Z= -2.4313539D+00
I= 4 X=  4.5350603D-01 Y=  0.0000000D+00 Z= -3.0344227D+00
MW Cartesian velocity:
I= 1 X= -4.0368385D+12 Y=  1.4729976D+13 Z=  1.4109897D+14
I= 2 X=  4.4547606D+13 Y= -6.3068948D+12 Z= -2.2951936D+14
I= 3 X= -3.0488505D+13 Y=  6.0922004D+12 Z=  1.8527270D+14
I= 4 X= -1.3305097D+14 Y= -3.1794401D+13 Z=  2.4220839D+14
Cartesian coordinates after ADCart:
I= 1 X= -1.1983609D-01 Y=  4.2521779D-04 Z= -1.0437931D+00
I= 2 X= -1.1859803D-01 Y= -1.5769743D-04 Z=  1.1248052D+00
I= 3 X=  2.8688210D+00 Y=  6.0685035D-04 Z= -2.4129040D+00
I= 4 X=  4.4028377D-01 Y= -3.1670730D-03 Z= -3.0103048D+00
TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ
```

After the trajectory computation is complete, summary information is displayed in the output for each time step in the trajectory:

```
Trajectory summary for trajectory      1
Energy/Fock evaluations           51
Gradient evaluations                   51

Trajectory summary
 Time (fs) Kinetic (au) Potent (au)  Delta E (au)  Delta A (h-bar)
0.000000 0.1000000 -114.3576722 0.0000000 0.0000000000000000
0.100000 0.0988486 -114.3564837 0.0000371 -0.0000000000000081
0.200000 0.0967812 -114.3543446 0.0001088 -0.0000000000000104
0.300000 0.0948898 -114.3524307 0.0001313 -0.0000000000000115
...
```

You can also use GaussView or other visualization software to display the trajectory path as an animation.

5.4 BD

This method keyword requests a Brueckner Doubles calculation [131–133]. BD gradients are available [133].

5.4.1 Options

T

Requests a Brueckner Doubles calculation with a triples contribution [132] added. BD-T is a synonym for BD(T).

TQ

Requests a Brueckner Doubles calculation with triples and quadruples contributions [134] added.

FC

All frozen core options are available with this keyword; a frozen core calculation is the default. See the discussion of the FC options for full information.

MaxCyc=*N*

Specifies the maximum number of cycles.

Conver=*N*

Sets the convergence calculations to 10^{-N} on the energy and $10^{-(N-2)}$ on the wavefunction. The default is N=4 for single points and N=6 for gradients.

TWInCore

Whether to store amplitudes and products in memory during higher-order post-SCF calculations. The default is to store these if possible, but to run off disk if memory is insufficient. TWInCore causes the program to terminate if these can not be held in memory, while NoTWInCore prohibits in-memory storage.

InCore

Forces the in-memory algorithm. This is very fast when it can be used, but requires $N^4/4$ words of memory. It is normally used in conjunction with SCF=InCore. NoInCore prevents the use of the in-core algorithm.

SaveAmplitudes

Saves the converged amplitudes in the checkpoint file for use in a subsequent calculation (e.g., using a larger basis set). Using this option results in a very large checkpoint file, but also may significantly speed up later calculations.

ReadAmplitudes

Reads the converged amplitudes from the checkpoint file (if present). Note that the new calculation can use a different basis set, method (if applicable), etc. than the original one.

Read

Reads the initial orbitals from the checkpoint file rather than doing an HF calculation. Note that the new calculation can use a different basis set than the original one.

5.4.2 Related Keywords

OldFCBD

Requests old-style frozen-core BD (the core orbitals are never changed).

NewFCBD

Requests new-style frozen-core BD, in which the core orbitals are updated to conform to the BD condition T1=0, which means the BD Fock matrix is diagonal for these orbitals. This is the default.

5.4.3 Availability

Analytic energies and gradients for BD, numerical gradients for BD(T), and numerical frequencies for all methods. The options FC, T and TQ are not available with analytic gradients. Unrestricted open-shell calculations are available for BD energies and gradients.

5.4.4 Examples

The BD energy appears in the output labeled `E(Corr)`, following the final correlation iteration:

```
Wavefunction amplitudes converged. E(Corr)=     -75.001908213
```

The energy is given in Hartrees. If triples (or triples and quadruples) were requested, the energy including these corrections appears after the preceding:

```
Time for triples=        0.14 seconds.
T4(BD)= -0.12680940D-03
BD(T)= -0.75002034980D+02
```
Triples-corrected energy.

5.5 BOMD

This keyword requests a classical trajectory calculation [120–123] using a Born-Oppenheimer molecular dynamics model (first described in [135, 136]; see [137] for an extended review article). The implementation in Gaussian 16 [138–140] extends the usual methodology by using a very accurate Hessian-based algorithm that incorporates a predictor step on the local quadratic surface followed by a corrector step. The latter uses a fifth-order polynomial or a rational function fitted to the energy, gradient, and Hessian at the beginning and end points of each step. This method for generating the correction step enables an increase in the step size of a factor of 10 or more over previous implementations.

The selection of the initial conditions using quasi-classical fixed normal mode sampling and the final product analysis are carried out in the same manner as in the classical trajectory program VENUS [141]. Alternatively, initial Cartesian coordinates and velocities may be read in.

Note that the ADMP method provides equivalent functionality at substantially lower computational cost at the Hartree-Fock and DFT levels.

5.5.1 Required Input

All BOMD jobs must specify the number of dissociation paths; for many jobs, this value will be zero (a blank line is also allowed), and no other BOMD input will be used. In this case, the trajectory is integrated for a fixed number of steps, as specified by the program default of 100 or the value of the MaxPoints option.

If *NPath* is set to -1, the dissociation pathways will be detected automatically and a gradient criteria (Hartree/Bohr) will be used in place of the usual fragmentation pathway and stopping criteria.

When the number of dissociation paths is greater than zero, the full BOMD job input has the following general structure:

NPath	*Number of dissociation paths (maximum=20)*
$\text{IFrag}_1, \cdots, \text{IFrag}_{NAtoms}$	*Fragmentation information*
$\cdots$	*$\cdots$Repeated NPath times*
[R1, R2, R3, R4, G5, ITest, IAtom, JAtom, R6	*Optional stopping criteria (ReadStop option)*
$\cdots$]	*$\cdots$Repeated NPath times*
[Estart,DelE,SBeta,Ef,DPert,IFlag]	*Optional simulated annealing params. (SimAnneal)*
[Mode-num, VibEng(Mode-num), $\cdots$]	*Optional initial normal mode energies (NSample)*
[Initial velocity for atom 1: x y z	*Optional initial velocities (ReadVelocity, ReadMWVelocity)*
Initial velocity for atom 2: x y z	

…	
Initial velocity for atom N: x y z	
…]	*Entire section repeated NTraj times*
[Atom1, Atom2, E_0, Len, D_e, B_e	*Optional Morse parameters for each diatomic product*
…]	
	Terminate subsection with a blank line

The input line(s) following *NPath* define the fragmentation information for each path. The value in each position specifies that the corresponding atom belongs to the specified fragment number (i.e., atom i belongs to fragment number $IFrag_i$). Note that fragment information for each path must begin on a new line, but the ones for any individual path may be continued over as many lines as necessary.

Stopping criteria are specified next when the ReadStop option is included. Up to six stopping criteria may be specified for each path. A trajectory is terminated when all of the active criteria are satisfied. However, a value of zero for any parameter turns off testing for the corresponding stopping criteria. The stopping criteria tests are defined as follows (default parameter values are in parentheses):

◊ Minimum distance between the centers of mass for any pair of fragments > *R1* (18)
◊ Minimum distance between atoms located in different fragments > *R2* (20)
◊ Maximum distance between the center of mass and any atom in the same fragment < *R3* (0)
◊ The maximum distance between any pair of atoms in the same fragment < *R4* (0)
◊ Interfragment gradient < *G5* (10^{-6})
◊ If *ITest*=1, distance between atoms *IAtom* and *JAtom* > *R6* (0). Otherwise, distance between atoms *IAtom* and *JAtom* < *R6* (0)

All distances are specified in Bohr, and the units of the gradient *G5* are Hartrees/Bohr.

Parameters for simulated annealing/fragmentation follow the stopping criteria in the input stream when the SimAnneal option is specified:

◊ *Estart* is the desired initial kinetic energy (Hartrees).
◊ *DelE* is the energy gain/loss in Hartrees.
◊ *SBeta* is the Fermi-Dirac inverse temperature (1/Hartrees).
◊ *Ef* is the Fermi energy (wavenumbers): all modes corresponding to a frequency in wavenumbers below *Ef* will be enhanced, while those above *Ef* will be reduced. The reverse will happen if *SBeta* is negative.
◊ *DPert* is the size of the random perturbation.
◊ *IFlag* determines the algorithm for applying an energy perturbation for simulated annealing (i.e., adding/removing energy from the eigenmodes). It has the following possible values: 0 (weigh each eigencomponent according to its frequency), 1 (add DelE in a random fashion), 2 (combination of 0 and 1), 00 (if near a transition state, add all energy along that mode), and 10 (ignore any nearby transition state).

The next part of the input specifies how much energy is in each normal mode when the NSample option is used. For each mode, *VibEng* is the translational energy in kcal/mol in the forward direction along the transition vector. If *VibEng* < 0, then the initial velocity is in the reverse direction. (You can explicitly specify the forward direction using the Phase option.)

Next, the initial velocity for each atom is read if the ReadVelocity or ReadMWVelocity option is included. Each initial velocity is specified as a Cartesian velocity in atomic units (Bohr/sec) or as a mass-weighted

Cartesian velocity (in amu$^{1/2}$*Bohr/sec), respectively. One complete set of velocities is read for each requested trajectory computation.

Finally, Morse parameter data can be specified for each diatomic product. The Morse parameter data is used to determine the vibrational excitation of diatomic fragments using the EBK quantization rules. It consists of the atomic symbols for the two atoms, the bond length between them (*Len*, in Angstroms), the energy at that distance (E_0 in Hartrees), and the Morse curve parameters D_e (Hartrees) and B_e (Angstroms^{-1}). This input subsection is terminated by a blank line.

5.5.2 Options

MaxPoints=*n*

Specifies the maximum number of steps that may be taken in each trajectory (the default is 100). If a trajectory job is restarted, the maximum number of steps will default to the number specified in the original calculation.

Phase=(*N1*, *N2*[,*N3*[,*N4*]])

Defines the phase for the transition vector such that forward motion along the transition vector corresponds to an increase in the specified internal coordinate. If two atom numbers are given, the coordinate is a bond stretch between the two atoms; three atom numbers specify an angle bend and four atoms define a dihedral angle.

ReadVelocity

Read initial Cartesian velocities from the input stream. Note that the velocities must have the same symmetry orientation as the molecule. This option suppresses the fifth-order anharmonicity correction.

ReadMWVelocity

Read initial mass-weighted Cartesian velocities from the input stream. Note that the velocities must have the same symmetry orientation as the molecule. This option suppresses the fifth-order anharmonicity correction.

SimAnneal

Use simulated annealing (the initial velocity is randomly generated). Additional parameters are read in, as described above.

Only one of ReadVelocity, ReadMWVelocity and SimAnneal can be specified.

ReadStop

Read in alternative stopping criteria.

RTemp=*N*

Specifies the rotational temperature. The default is to choose the initial rotational energy from a thermal distribution assuming a symmetric top (the temperature defaults to 0 K).

NSample=*N*

Read in initial kinetic energies for the first *N* normal modes (the default is 0). The energies for the remaining modes are determined by thermal sampling by default.

NTraj=*N*

Compute *N* trajectories.

Update=*n*

By default BOMD does second derivatives at every point. Using the Update keyword causes the program to perform Hessian updates for n gradient points before doing a new analytic Hessian. GradOnly requests that only gradients be done and that the Hessian be updated all the time (full second derivatives are not computed). ReCalcFC is a synonym for this option.

RandomVelocity

Randomly generate initial velocities for dynamics without using any second derivative information. This is the default for GradientOnly dynamics.

StepSize=*n*

Sets the dynamic step size to n*0.0001, in the appropriate units. The default step size is 0.25 amu$^{1/2}$*Bohr except for *GradientOnly* calculations where it is 0.025 femtoseconds.

Sample=*type*

Specifies the type of sampling, where *type* is one of these keywords: Microcanonical, Fixed, and Local. The default is Fixed normal mode energy unless RTemp was specified, in which case Local mode sampling is implied.

Restart

Restart a trajectory calculation from the checkpoint file. Note that options set in the original job will continue to be in effect and cannot be modified.

ReadIsotopes

This option allows you to specify alternatives to the default temperature, pressure, frequency scale factor and/or isotopes – 298.15 K, 1 atmosphere, no scaling, and the most abundant isotopes (respectively). It is useful when you want to rerun an analysis using different parameters from the data in a checkpoint file. Be aware, however, that all of these can be specified in the route section (Temperature, Pressure and Scale keywords) and molecule specification (the Iso parameter), as in this example:

```
#T Method/6-31G(d) JobType Temperature=300.0 ···

...

0 1
C(Iso=13)
...
```

ReadIsotopes input has the following format:

temp pressure [scale]	*Values must be real numbers.*
isotope mass for atom 1	
isotope mass for atom 2	
...	
isotope mass for atom n	

Where *temp*, *pressure*, and *scale* are the desired temperature, pressure, and an optional scale factor for frequency data when used for thermochemical analysis (the default is unscaled). The remaining lines hold the isotope masses for the various atoms in the molecule, arranged in the same order as they appeared in the molecule specification section. If integers are used to specify the atomic masses, the program will automatically use the corresponding actual exact isotopic mass (e.g., 18 specifies ^{18}O, and Gaussian uses the value 17.99916).

5.5.3 Availability

All semi-empirical, HF, CASSCF, CIS, MP2 and DFT methods.

5.5.4 Related Keywords

ADMP

5.5.5 Examples

The following sample BOMD input file illustrates many of the available options. It will calculate a trajectory for H_2CO dissociating to H_2 + CO, starting at the transition state. There is one fragmentation pathway: C and O belong to fragment 1, and the two hydrogens belong to fragment 2.

Stopping criteria are also specified in this example job. The trajectory will be stopped if the distance between the centers of mass of H_2 and CO exceed 13 Bohr, the closest distance between H_2 and CO exceeds 11 Bohr, all atoms in a fragment are less than 1.3 Bohr from the center of mass of the fragment, any atom in the fragment is less than 2.5 Bohr from all other atoms in the fragment, the gradient for the separation of the fragments is less than 0.0000005 Hartree/Bohr, and the distance between atoms 1 and 3 is greater than 12.8 Bohr.

The initial kinetic energy along the transition vector is 5.145 kcal/mol, in the direction of the products (the forward direction is characterized by an increase in the larger C-H distance). The Morse parameters for H_2 and CO are specified to determine the vibrational excitation of the product diatomics; they were computed in a previous calculation. The calculation will be carried out at 300 K.

```
# HF/3-21G BOMD(Phase=(1,3),RTemp=300,NSample=1,ReadStop) Geom=Crowd

HF/3-21G dissociation of H2CO --> H2 + CO

0 1
C
O 1 r1
H 1 r2 2 a
H 1 r3 3 b 2 180.

r1 1.15275608
r2 1.74415774
r3 1.09413376
a 114.81897892
b 49.08562961

1
1 1 2 2
```

```
13.0 11.0 1.3 2.5 0.0000005 1 1 3 12.8
1 5.145
C O -112.09329898 1.12895435 0.49458169 2.24078955
H H -1.12295984 0.73482237 0.19500473 1.94603924
```
Final blank line

Note that all six stopping criteria are used here only for illustrative purposes. In most cases, one or two stopping criteria are sufficient.

At the beginning of a BOMD calculation, the parameters used for the job are displayed in the output:

```
TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ-TRJ
-------------------------------------------------------------------
INPUT DATA FOR L118
-------------------------------------------------------------------


General parameters:
Max. points for each Traj.   =     100
Total Number of Trajectories =       1
Random Number Generator Seed =  398465
Trajectory Step Size         =   0.250 sqrt(amu)*bohr

Sampling parameters:
Vib Energy Sampling Option   = Thermal sampling
  Vib Sampling Temperature   =     300.0 K
  Sampling direction         = Forward
Rot Energy Sampling Option   = Thermal distribution (symmetric top)
  Rot Sampling Temperature   =     300.0 K
Start point scaling criteria =       1.000D-05 Hartree
...

Reaction Path  1
****************
Fragment  1    center   1 ( C )   2 ( O )
Fragment  2    center   3 ( H )   4 ( H )
Termination criteria:
  The CM distances are larger than 13.000 bohr
  The min atomic distances among fragments are larger than 11.0 bohr
  The max atomic and CM distances in frags are shorter than 1.3 bohr
  The max atomic distances in fragments are shorter than 2.500 bohr
  The change of gradient along CM is less than 5.00D-07 Hartree/bohr
  Distance between atom center 1 ( C ) and 3 ( H ) is GE 12.800 bohr

Morse parameters for diatomic fragments:
```

```
            E0             Re             De             Be
 C  O   -112.0932990     1.1289544      0.4945817      2.2407896
 H  H     -1.1229598     0.7348224      0.1950047      1.9460392
---------------------------------------------------------------------
```

The initial kinetic energies for the normal modes appear at the beginning of each trajectory step:

```
 -------------------------------------------------------
          Thermal Sampling of Vibrational Modes
 Mode     Wavenumber     Vib. quant.#  Energy (kcal/mol)
 -------------------------------------------------------
    1     -2212.761                        5.14500
    2       837.330            0           1.19702
    3      1113.182            0           1.59137
    4      1392.476            0           1.99064
    5      2026.859            0           2.89754
    6      3168.689            0           4.52987
-------------------------------------------------------
```

After the trajectory computation is complete, summary information is displayed in the output:

```
 Trajectory summary for trajectory      1
 Energy/gradient evaluations           76
 Hessian evaluations                  76

 Trajectory summary
  Time (fs) Kinetic (au) Potent (au) Delta E (au)    Delta A (h-bar)
  0.000000  0.0214192  -113.0388912   0.0000000    0.0000000000000000
  1.169296  0.0293490  -113.0468302  -0.0000091    0.0000000000053006
  2.161873  0.0407383  -113.0582248  -0.0000144    0.0000000000045404
  ...
```

The information is given for each time step in the trajectory. In addition, the output includes geometrical parameters for the atoms in each fragment, the distances between fragments, and the relative mass-weighted velocities for each fragments and between fragments, all reported at each time step. You can also use GaussView or other visualization software to display the trajectory path as an animation.

5.6 CacheSize

This keyword specifies the default amount of cache per processor to use with various cache-blocking algorithms (in 8-byte words). Setting this value appropriately will maximize the efficiency of relevant algorithms by matching the batch size to the available cache.

Typically, this keyword is used in *Default.Route* files. You can use the testrt utility to determine the default value for the Gaussian binary installed on the current system. On Linux and most other UNIX-based systems, the cachesize script can be used to determine the optimal value for this parameter (see the examples).

When Gaussian 16 binaries are built, this value is set correctly for the current version of the various CPU types, but it may need to be modified for chips released later which might use a different cache-size or inter-core cache sharing scheme. If you build Gaussian 16 from source code, this value corresponds to the hardware on the system where the compile takes place.

5.6.1 Examples

The following testrt command output lists the value for the Gaussian 16 binaries:

$ testrt sp

```
 Revision:  EM64M-G16RevA.02  9-Aug-2016 NErtGn=1000 NextEG=955 MaxAtm=
   250000 MDCach=131072
```

Remainder of testrt output ···

The value of 131,072 corresponds to 128K working precision words, or 1 MB. This Mac OS X system has a quad-core 64-bit Intel processor with 6 MB of Level 3 cache. A reasonable value for cache use is 50% of the available cache, so a value of 0.75 MB/core would be a reasonable (50% of 1.5 MB): CacheSize=98304 (value is in 8-byte words).

On Linux systems, the cache size is recorded in /proc/cpuinfo. The cachesize script in the $g16root/g16/bsd subdirectory will display the total cache size:

$ $g16root/g16/bsd/cachesize

```
524288
```
bytes

This indicates that there is 512 KB of cache on this ancient dual-core system. In this case, due to the age of this computer, the per-processor cache size is reported. Giving any parameter to this script will display the CacheSize keyword to use:

$ $g16root/g16/bsd/cachesize 1

```
CacheSize=32768
```

This will use 0.25 MB of cache per core.

5.7 CASSCF

This method keyword requests a Complete Active Space Multiconfiguration SCF (MC-SCF) [142–150]. An MC-SCF calculation is a combination of an SCF computation with a full CI involving a subset of the orbitals; this subset is known as the *active space*. The number of electrons (N) and the number of orbitals (M) in the active space for a CASSCF *must* be specified following the keyword: CASSCF(N,M). Note that options may be interspersed with N and M in any order.

By default, the active space is defined assuming that the electrons come from the highest occupied orbitals in the initial guess determinant and that the remaining orbitals required for the active space come from the lowest virtuals of the initial guess. Thus, for a 4-electron, 6-orbital CAS – specified as CASSCF(4,6) – on a closed-shell system, the active space would consist of:

◊ Enough occupied orbitals from the guess to provide 4 electrons. Thus, the 2 highest occupied MOs would be included.
◊ Enough virtual orbitals to make a total of 6 orbitals. Since 2 occupied orbitals were included, the lowest 4 virtual orbitals would become part of the active space.

Similarly, a 4 electron, 6 orbital CAS on a triplet would include the highest 3 occupied orbitals (one of which is doubly occupied and two singly occupied in the guess determinant) and the lowest 3 virtual orbitals. In Gaussian 16, algorithmic improvements make an active space of up to about 16 orbitals feasible [151].

Normally, Guess=Alter or Guess=Permute is necessary to ensure that the orbitals which are selected involve the electrons of interest and that they are correlated correctly. A prior run with Guess=Only can be used to quickly determine the orbital symmetries (see the first example below). Alternatively, a full Hartree-Fock single point calculation may be done, and the subsequent job will include Guess=(Read,Permute) in order to retrieve and then modify the computed initial guess from the checkpoint file. You need to include Pop=Reg in the route section of the preliminary job in order to include the orbital coefficient information in the output (use Pop=Full for cases where you need to examine more than just the few lowest virtual orbitals). Alternatively, you may use Pop=NBOSave to save the NBOs, which are often the best choice for starting CAS orbitals. You may also choose to view the orbitals in a visualization package such as GaussView.

CAS is a synonym for CASSCF.

Use #P in the route section to include the final eigenvalues and eigenvectors in addition to the energy and one-electron density matrix in the CASSCF output.

Note: CASSCF is a powerful but advanced method with many subtleties. We strongly recommend that you study the cited references before attempting to run production CASSCF calculations. An overview of the CASSCF method is given in chapter 9 of *Exploring Chemistry with Electronic Structure Methods, 3rd ed.* [152]. Relatively straightforward example applications are discussed in references [145, 153–159].

5.7.1 Variations

◊ An MP2-level electron correlation correction to the CASSCF energy may be computed during a CASSCF calculation by specifying the MP2 keyword in addition to CASSCF within the route section [160].

◊ Calculations on excited states of molecular systems may be requested using the NRoot option. Note that a value of 1 specifies the ground state, not the first excited state (in contrast to usage with the CIS or TD keywords).

◊ State-averaged CASSCF calculations may be performed using the StateAverage and NRoot options to specify the states to be used.

◊ Conical intersections and avoided crossings may be computed by including Opt=Conical in the route section of a CASSCF job (see the examples) [161–163].

◊ Approximate spin orbit coupling between two spin states can be computed during CASSCF calculations by including the SpinOrbit option [164–170]. The method used in Gaussian 16 is based on [166]. It is available for the elements H through Cl. In order to compute the spin orbit coupling, the integrals are computed in a one-electron approximation involving relativistic terms, and then it uses effective charges that scale the Z value for each atom to empirically account for 2 electron effects. This value can be specified for each atom via the molecule specification nuclear parameters list (discussed earlier in this chapter). Finally, note that such calculations will be state-averaged by default.

◊ The Restricted Active Space variation (RASSCF) [171] is also supported [172]. It is selected via the RAS option. RASSCF calculations partition the molecular orbitals into five sections:

- The lowest-lying occupieds (doubly occupied in all configurations).
- The RAS1 space of doubly occupied MOs.
- The RAS2 space containing the most important orbitals for the problem.

- The RAS3 space of weakly occupied MOs.
- The remaining unoccupied orbitals.

Thus, the active space in CASSCF calculations is divided into three parts in a RAS calculations, and allowed configurations are defined by specifying the minimum number of electrons that must be present in the RAS1 space and the maximum number that may be in the RAS3 space, in addition to the total number of electrons in the three RAS spaces. See the discussion of the RAS option for a description of how to specify these values.

5.7.2 Options

NRoot=*j*

Requests that the jth root of the CI be used so that an excited state is obtained when $j > 1$. The option defaults to the ground state (j=1). The state specified by NRoot is referred to as the “state of interest.”

StateAverage

Used to specify a state-averaged CASSCF calculation. All states up to NRoot are averaged. This option requires the weighting for the various states to be input in format nF10.8 (no trailing blank line). StateAverage is not allowed in combination with Opt=Conical or CASSCF=SpinOrbit, both of which perform state-averaged calculations by default.

SpinOrbit

Compute approximate spin orbit coupling between two states, specified on a separate input line. Implies a state-averaged CASSCF calculation.

RAS=(*a*,*b*,*c*,*d*)

Requests a RASSCF calculation which allows up to *a* holes (i.e., excitations from RAS1 into RAS2 or RAS3) in the *b* orbitals in the RAS1 space, and to *c* particles in the *d* orbitals in the RAS3 space (i.e., excitations from RAS1 or RAS2 into RAS3). Thus, the minimum number of electrons in RAS2 is $2b$-a. Note that the two CASSCF keyword parameters specify the size of the *entire* active space: RAS1 + RAS2 + RAS3 (see the examples).

DavidsonDiag

Requests the use of the Davidson diagonalization method for the CI matrix instead of the Lanczos iterations. This is the default when there are more than six active orbitals.

LanczosDiag

Requests the use of Lanczos iterations when diagonalizing the CI matrix instead of the Davidson method. Lanczos is the default when there are 6 or fewer active orbitals.

FullDiag

Requests the use of the full (Jacobi) diagonalization method for the CI matrix instead of Lanczos or Davidson iterations. The default is full diagonalization if there are 6 or fewer active orbitals and Davidson otherwise. NoFullDiag suppresses the use of the full diagonalization method.

The full Jacobi diagonalization method must be used if quadratic convergence is required (see the QC option below) and when the CI eigenvector is unknown (in the latter case, specify FullDiag for calculations involving more than 6 active orbitals).

StateGuess=*k*

Set the starting vector for the Lanczos method to configuration *k*. For example, this option can be useful for selecting a configuration of the correct symmetry for a desired excited state (different from that of the ground state). In such cases, running a preliminary calculation to determine the orbital symmetries may be required.

k may also be set to the special value Read, which says to read in the entire eigenvector from the input stream (format: *NZ, (Ind(I), C(Ind(I)), I=1, NZ*).

The default diagonalization method is most efficient if the size of the CI problem is greater than about 50, or the user can identify one or more dominant components in the eigenvector from the onset of the calculation, via the initial trail vector. By default, the starting vector is initialized in *j*+1 positions, where *j* is the value given to the NRoot option (or its default value). The positions correspond to the lowest *j*+1 energy diagonal elements of the CI Hamiltonian. This usually results in good convergence for the lowest *j* roots.

The StateGuess option (below) may be used to change this default. CASSCF($\cdots$,StateGuess=*k*) sets C(*k*) to 1.0. The central requirement for this vector is that it should not be deficient in the eigenvector that is required. Thus, if the CI eigenvector is dominated by configuration k, setting the StateGuess option to *k* will generate a good starting vector (e.g., StateGuess=1 is appropriate if the CI vector is dominated by the SCF wavefunction). However, if the coefficient of configuration *k* is exactly zero (e.g., by symmetry) in the desired root, then that eigenvector will be missing, and the calculation will converge to a higher state.

OrbRot

OrbRot includes and NoCPMCSCF excludes the orbital rotation derivative contributions from the CP-MC-SCF equations in an Opt=Conical calculation. OrbRot is the default.

SlaterDet

Use Slater determinants in the CASSCF calculation. This option is needed to locate a conical intersection/avoided crossing between a singlet state and a triplet state.

SaveGEDensities

Saves ground- and excited-state alpha and beta total and transition density matrices (as is done for CIS). Forces the use of Slater determinants.

HWDet

Use Hartree-Waller determinants instead of Slater. This is the default for CAS calculations involving 10 or more orbitals. It implies NoFullDiag.

RFO

Requests the RFO quadratic step. At most, one of QC and RFO should be specified.

QC

Requests a quadratically convergent algorithm for the CAS. This option should be used with caution; it works well only with a very good guess. Only one of QC and RFO should be specified.

UNO

Requests that the initial orbitals for the CAS be produced from the natural orbitals generated from a previous UHF calculation [173, 174]. Normally used with Guess=Read.

The UNO guess must be used with caution. Often, some of the natural orbitals which have modest occupation are not the important ones for the process of interest. Consequently, unless the entire valence space is being correlated (which is usually prohibitively expensive), one normally runs one job which does a

UHF calculation with Pop=NaturalOrbitals and then examines the resulting orbitals. The orbitals which belong in the active space are then selected, and a single-point CASSCF(···,UNO) Guess=(Read, Alter) calculation is performed. The resulting converged orbitals are then examined to verify that the correct active space has been located, and finally an optimization can be run with CASSCF(···,UNO) Guess=Read. For singlets, this entire process depends on the user being able to coax the UHF wavefunction to converge to the appropriate broken spin-symmetry (non-RHF) result.

NPairs=*n*

Number of GVB pairs outside of the CAS active space in a CAS-GVB calculation [175].

5.7.3 Availability and Restrictions

Energies, analytic gradients, and analytic and numerical frequencies. CASSCF may not be combined with any semi-empirical method. Analytic gradients and frequencies are available only through f functions.

Analytic polarizabilities may not be performed with the CASSCF method. Use CASSCF Polar=Numer.

You can restart a CASSCF calculation by specifying SCF=Restart in the route section. In order to restart a CASSCF optimization, the keywords CASSCF Opt=Restart Extralinks=L405 must be included in the job's route section.

CASSCF frequencies with PCM solvation must be done numerically using Freq=Numer.

5.7.4 Related Keywords

Opt=Conical, MP2, Guess, Pop, SCF

5.7.5 Examples

We will consider several of the most important uses of the CASSCF method in this section.

Preliminary Examination of the Orbitals (Guess=Only). The following route section illustrates one method of quickly examining the orbitals in order to determine their symmetries and any alterations needed to produce the desired initial state. We include Pop=Reg to obtain the molecular orbital output in the population analysis section:

```
# HF/3-21G Guess=Only Pop=Reg Test
```

The molecule being investigated is 1,3-cyclobutadiene, a singlet with D_{2h} symmetry. We are going to run a 4×4 CAS, so there will be four orbitals in the active space: 2 occupied and 2 virtual. We want all four orbitals to be π orbitals.

The HOMO is orbital 14; therefore, orbitals 13 through 16 will comprise the active space. When we examine these orbitals, we see that only orbitals 14 and 15 are of the correct type. The molecule lies in the YZ-plane, so π orbitals will have significantly non-zero coefficients in the X direction. Here are the relevant coefficients for orbitals 10 and 13-16:

```
Molecular Orbital Coefficients
                     10        13        14        15        16
                      O         O         O         V         V
 3 1 C     2PX     0.29536   0.00000   0.34716   0.37752   0.00000
 7         3PX     0.16911   0.00000   0.21750   0.24339   0.00000
12 2 C     2PX     0.29536   0.00000   0.34716  -0.37752   0.00000
```

```
16          3PX      0.16911    0.00000    0.21750   -0.24339    0.00000
21  3  C    2PX      0.29536    0.00000   -0.34716   -0.37752    0.00000
25          3PX      0.16911    0.00000   -0.21750   -0.24339    0.00000
30  4  C    2PX      0.29536    0.00000   -0.34716    0.37752    0.00000
34          3PX      0.16911    0.00000   -0.21750    0.24339    0.00000
```

Orbital 10 is clearly also a π orbital. If we look at higher virtual orbitals, we will find that orbital 19 is also a π orbital. We have found our four necessary orbitals, and can now use Guess=Alter to move them into the active space. Here is the input file for the CASSCF calculation:

```
# CASSCF(4,4)/3-21G Guess=Alter Pop=Reg  Test

1,3-Cyclobutadiene Singlet, D2H, Pi 4×4 CAS

0 1
```

molecule specification

```
10,13               Interchange orbitals 10 and 13.
16,19               Interchange orbitals 16 and 19.
```

CASSCF Energy and the One-Electron Density Matrix. When we run this CASSCF calculation on cyclobutadiene, we will obtain a prediction for the energy. It appears in the CASSCF output as follows:

```
TOTAL                  -152.836259      ··· Energy at each iteration
 ITN=   9 MaxIt= 64 E=   -152.8402786733 DE=-1.17D-05 Acc= 1.00D-05
 ITN=  10 MaxIt= 64 E=   -152.8402826495 DE=-3.98D-06 Acc= 1.00D-05
 ...
 DO AN EXTRA-ITERATION FOR FINAL PRINTING
```

The value of E for the final iteration is the predicted energy: -152.8402826495 Hartrees in this case.

It is also important to examine the one-electron density matrix, which appears next in the output:

```
Final one electron symbolic density matrix:
             1             2             3             4
   1  0.191842D+01
   2 -0.139172D-05  0.182680D+01
   3  0.345450D-05  0.130613D-05  0.172679D+00
   4  0.327584D-06  0.415187D-05  0.564187D-06  0.820965D-01
 MCSCF converged.
```

The diagonal elements indicate the approximate occupancies for each successive orbital in the active space. If any of these values is (essentially) zero, then that orbital was empty throughout the calculation; similarly, if any of them is essentially 2, then that orbital was doubly occupied throughout the CAS. In either case, there were no excitations into or out of the orbital in question, and there is probably a problem with the CASSCF calculation. In our case, the two “occupied” orbitals have values less than 2, and the other two orbitals in the active space have non-zero occupancies, so things are fine.

CASSCF MP2 Energy. When you run a CASSCF calculation with dynamic correlation (CASSCF MP2 in the route section), the following additional lines will appear in the CASSCF output (with the first one coming significantly before the second):

```
MP2 correction to the MCSCF energy is computed        Indicates a CASSCF MP2 job.
...
E2 = -0.2635549296D+00 EUMP2 = -0.15310383973610D+03 Electron correlation-corrected energy.
```

The string EUMP2 labels the energy; in this case, the value is -153.1038397361 Hartrees.

CAS Configuration Information. The beginning of the CASSCF output lists the configurations, in the following format:

```
           Configuration          1 Symmetry 1 1100
           Configuration          2 Symmetry 2 1ab0
           Configuration          3 Symmetry 1 1010
           Configuration          4 Symmetry 1 a1b0
```

This is from a CAS(4,4) on a singlet reference, so each configuration indicates the occupation pattern for the 4 active orbitals. The first line is the reference configuration and in this case has the two lowest active orbitals doubly occupied, indicated with "1". In configuration 2, the first active orbital remains doubly occupied, while a β electron has been excited from the second to the third active orbital indicated by "a" for α and "b" for β. In configuration 3, the first and third active orbitals are doubly occupied, while configuration 4 shows excitation of the β electron from the first to the third active orbital. By default, all symmetry types are allowed, and the symmetry of each configuration is reported. Refer to the symmetry multiplication table printed before the configuration list for symmetry assignments of the orbitals.

Using CASSCF to Study Excited States. The following two-step job illustrates one method for studying excited state systems using the CASSCF method. The first step assumes that a preliminary Hartree-Fock single point calculation has been done in order to examine the orbitals; it takes advantage of the initial guess computation done by that job, which is retrieved from the checkpoint file:

```
%chk=CAS1
# CASSCF(2,4) 6-31+G(D) Guess=(Read,Alter) Pop=NaturalOrbital Test Geom=Check

Alter the guess so that the three LUMOs are all the desired symmetry, and run
   the CAS

0,1

orbital alterations

--Link1--
%chk=CAS1
%nosave
# CASSCF(2,4,NRoot=2) 6-31+G(D) Guess(Read) Pop(NaturalOrbital) Geom=Check Test

Excited state calculation
```

```
0,1
```

The second job step uses the NRoot option to CASSCF to specify the first excited state. The first excitation energy for the system will then be computed by taking the energy difference between the two states (see exercise 5 in chapter 9 of *Exploring Chemistry with Electronic Structure Methods* [176] for a more detailed discussion of this technique).

Predicting Conical Intersections. Including Opt=Conical keyword in the route section changes the job from an optimization of the specified state using CASSCF to a search for a conical intersection or avoided crossing involving that state. The optimized structure will be that of the conical intersection or avoided crossing. Distinguishing between these two possibilities may be accomplished by examining the final eigenvalues in the CASSCF output for the final optimization step (it precedes the optimized structure):

```
FINAL EIGENVALUES AND EIGENVECTORS
 VECTOR EIGENVALUES       CORRESPONDING EIGENVECTOR

     state energy
     1   -154.0503161       0.72053292          -0.48879229 ···
                          -0.16028934E-02       0.31874441E-02 ···
     2   -154.0501151       0.45467877           0.77417416 ···
```

If the two eigenvalues (the first entry in the lines labeled with a state number) are essentially the same, then the energies of the two states are the same, and it is a conical intersection. Otherwise, it is an avoided crossing.

Spin Orbit Coupling. Here is the output from a CASSCF calculation where the spin orbit coupling has been requested with the Spin option (the coupling is between the state specified to the NRoot option and the next lower state):

```
 ****************************
  spin-orbit coupling program
 ****************************
 Number of configs= 4
 1st state is 1  States for which spin orbit coupling is computed.
 2nd state is 2
 Transition Spin Density Matrix
             1             2
   1  .000000D+00  .141313D+01
   2  .553225D-01  .000000D+00
 magnitude in x-direction=     .0000000 cm-1
 magnitude in y-direction=     .0000000 cm-1
 magnitude in z-direction=   55.2016070 cm-1
 total magnitude=   55.2016070 cm-1  Spin orbit coupling.
 MCSCF converged.
```

The spin orbit coupling is broken down into X, Y, and Z components, followed by its total magnitude, which in this case is 55.2016070 cm^{-1}.

RASSCF example. Here is an example RASSCF calculation route section:

```
# CAS(16,18,RASSCF(1,2,3,4)) 6-31G(d)
```

If this molecule is a neutral singlet, then this route defines the following spaces: RAS1 with 2 orbitals, 3 or 4 electrons in all configurations; RAS2 with 12 orbitals, 12 electrons in the reference configuration; and RAS3 with 4 orbitals, 0-3 electrons in all configurations. Thus, the RAS2 space will have 9 to 13 electrons in all configurations. The orbitals taken from the reference determinant for the active space are (assuming a spin singlet) the 8 highest occupieds and 10 lowest virtuals: i.e., same orbitals as for a regular CAS(16,18).

5.8 CBS Methods

The following method keywords specify various Complete Basis Set (CBS) methods of Petersson and coworkers for computing very accurate energies [44, 45, 110, 177–181]:

- ◊ CBS-4M
- ◊ CBS-QB3
- ◊ CBS-APNO

The keywords refer to the modified version of CBS-4 [180, 181], CBS-Q//B3 [110, 181] and CBS-APNO [180] methods, respectively. No basis set should be specified with any of these keywords. The RO prefix may be added to CBS-QB3 to request the ROCBS-QB3 method [182].

These methods are complex energy computations involving several pre-defined calculations on the specified system. All of these distinct steps are performed automatically when one of these keywords is specified, and the final computed energy value is displayed in the output.

Either of the Opt=Maxcyc=*n*, QCISD=Maxcyc=*n* or CCSD=Maxcyc=*n* keywords may be used in conjunction with any of the these keywords to specify the maximum number of optimization, QCISD, or CCSD cycles, respectively.

5.8.1 Options

SP

Do only a single-point energy evaluation using the specified compound model chemistry. No zero-point or thermal energies are included.

NoOpt

Perform the frequencies and single-point energy calculation for the specified model chemistry at the input geometry. Freq=TProjected is implied. This option is not meaningful or accepted for methods such as G1, which use different geometries for the frequencies and the single-point steps. StartFreq is a synonym for NoOpt.

ReadIsotopes

This option allows you to specify alternatives to the default temperature, pressure, frequency scale factor and/or isotopes – 298.15 K, 1 atmosphere, no scaling, and the most abundant isotopes (respectively). It is useful when you want to rerun an analysis using different parameters from the data in a checkpoint file. Be aware, however, that all of these can be specified in the route section (Temperature, Pressure and Scale keywords) and molecule specification (the Iso parameter), as in this example:

```
#T Method/6-31G(d) JobType Temperature=300.0 ···
```

```
...

0 1
C(Iso=13)
...
```

ReadIsotopes input has the following format:

temp pressure [scale]	*Values must be real numbers.*
isotope mass for atom 1	
isotope mass for atom 2	
...	
isotope mass for atom n	

Where *temp*, *pressure*, and *scale* are the desired temperature, pressure, and an optional scale factor for frequency data when used for thermochemical analysis (the default is unscaled). The remaining lines hold the isotope masses for the various atoms in the molecule, arranged in the same order as they appeared in the molecule specification section. If integers are used to specify the atomic masses, the program will automatically use the corresponding actual exact isotopic mass (e.g., 18 specifies ^{18}O, and Gaussian uses the value 17.99916).

Restart

Restart from the checkpoint file from a previous CBS calculation. The new job will start after the last successful calculation of the previous (unfinished) run.

5.8.2 Availability

Energies only.

CBS-4M and CBS-QB3 are available for first and second row atoms; CBS-APNO is available for first row atoms only.

RO may be combined with CBS-4M and CBS-QB3.

The original CBS-4 model chemistry has been updated with both the new localization procedure and improved empirical parameters [181]. The new version, CBS-4M, (M referring to the use of Minimal Population localization) is recommended for new studies.

5.8.3 Examples

The output from each step of a CBS method calculation is included in the output file. The final section of the file contains a summary of the results of the entire run.

CBS Summary Output. Here is the output from a CBS-QB3 calculation on CH_2 (triplet state):

```
Complete Basis Set (CBS) Extrapolation:
M. R. Nyden and G. A. Petersson, JCP 75, 1843 (1981)
G. A. Petersson and M. A. Al-Laham, JCP 94, 6081 (1991)
G. A. Petersson, T. Tensfeldt, and J. A. Montgomery, JCP 94, 6091 (1991)
J. A. Montgomery, J. W. Ochterski, and G. A. Petersson, JCP 101, 5900 (1994)
```

```
 Temperature=          298.150000 Pressure=                 1.000000
 E(ZPE)=                 0.016991 E(Thermal)=               0.019855
 E(SCF)=               -38.936447 DE(MP2)=                 -0.114761
 DE(CBS)=               -0.011936 DE(MP34)=                -0.018720
 DE(CCSD)=              -0.002759 DE(Int)=                  0.004204
 DE(Empirical)=         -0.006404
 CBS-QB3  (0 K)=       -39.069832 CBS-QB3 Energy=         -39.066969
 CBS-QB3 Enthalpy=     -39.066025 CBS-QB3 Free Energy=    -39.088192
```

The temperature and pressure are given first, followed by the components terms of the CBS-QB3 energy. The second-to-last line gives the CBS-QB3 energy values (reading across): at 0 K and at the specified temperature (298.15 K by default). The final line gives the CBS-QB3 enthalpy (including the thermal correction for the specified temperature) and the Gibbs free energy computed via the CBS-QB3 method (i.e., the CBS-QB3 energy including the frequency job free-energy correction). All of the energies are in Hartrees.

Rerunning the Calculation at a Different Temperature. The following two-step job illustrates the method for running a second (very rapid) CBS calculation at a different temperature. This job computes the CBS-QB3 energy at 298.15 K and then again at 300 K:

The energy labels thus have the following meanings (CBS-QB3 is used as an example):

CBS-QB3 (0 K)	Zero-point-corrected electronic energy: $E_0 = E_{elec} + ZPE$
CBS-QB3 Energy	Thermal-corrected energy: $E = E_0 + E_{trans} + E_{rot} + E_{vib}$
CBS-QB3 Enthalpy	Enthalpy computed using the CBS-QB3 predicted energy: $H = E + RT$
CBS-QB3 Free Energy	Gibbs Free Energy computed using the CBS-QB3 predicted energy: $G = H - TS$

```
%Chk=cbs
# CBS-QB3 Test

CBS-QB3 on formaldehyde

0 1
```
molecule specification

```
--Link1--
%Chk=cbs
%NoSave
# CBS-QB3(Restart,ReadIso) Geom=AllCheck Test

300.0 1.0
```
isotope specifications

5.9 CBSExtrapolate

This keyword requests a general Complete Basis Set extrapolation of the MP2 energy [44, 45, 177, 178].

The method requires two parameters: the minimum number of pair natural orbitals and the integration grid, which are set with the CBSExtrapolate=NMin and Integral=Grid options, respectively.

The minimum number of pair natural orbitals defaults to 5 for the 6-31G**, 6-31G‡ and 6-311G** basis sets (with or without diffuse functions), and to 10 for the 6-311G basis set with (2df,p) or (3df,p) polarization functions (again, with or without diffuse functions). NMin *must* be specified in all other cases, or an error will result.

The default integration grid is the (99,590) grid; an alternate grid can be specified with the Integral=Grid keyword. The integration portion is a small part of the total CBS extrapolation computation, so this relatively large grid was chosen. See the description of the Integral keyword for a full discussion of the available grids.

5.9.1 Options

NMin=*N*

Specifies *N* as the minimum number of pair natural orbitals.

MinPopLocal

Use localization based on populations in minimal basis [181]. This is the default.

PopLocal

Use population localization as described in reference [183].

BoysLocal

Use Boys localization [184–186].

NoLocal

Do not use any localization.

NRPopLocal

Newton-Raphson population localization.

NRBoysLocal

Newton-Raphson Boys localization.

NRMinPopLocal

Use 2nd order minimal population analysis.

SaveOrbitals

Save the localized CBS orbitals to the read-write file. Note that they will replace the SCF orbitals.

5.9.2 Availability

Single point energy calculations only, using any electron correlation method.

5.9.3 Related Keywords

Int=Grid, CBS

5.10 CCD and CCSD

These method keywords request coupled cluster calculations [187], using double substitutions from the Hartree-Fock determinant for CCD [188], or both single and double substitutions for CCSD [189–192]. CC and QCID are synonyms for CCD. RO may be combined with CCSD for a restricted open-shell energy calculation [193].

5.10.1 Options

FC

All frozen core options are available with this keyword; a frozen core calculation is the default. See the discussion of the FC options for full information.

T

Include triple excitations non-iteratively [190, 194] (CCSD only). CCSD-T is a synonym for CCSD(T).

E4T

Used with the T option to request inclusion of triple excitations for both the complete MP4 as well as CCSD(T).

T1Diag

Computes the T1 diagnostic of T. J. Lee and coworkers [195] (CCSD only).

Conver=*N*

Sets the convergence calculations to 10^{-N} on the energy and $10^{-(N-2)}$ on the wavefunction. The default is N=7 for single points and N=8 for gradients.

MaxCyc=*N*

Specifies the maximum number of cycles for CCSD calculations.

TWInCore

Whether to store amplitudes and products in memory during higher-order post-SCF calculations. The default is to store these if possible, but to run off disk if memory is insufficient. TWInCore causes the program to terminate if these can not be held in memory, while NoTWInCore prohibits in-memory storage.

SaveAmplitudes

Saves the converged amplitudes in the checkpoint file for use in a subsequent calculation (e.g., using a larger basis set). Using this option results in a very large checkpoint file, but also may significantly speed up later calculations.

ReadAmplitudes

Reads the converged amplitudes from the checkpoint file (if present). Note that the new calculation can use a different basis set, method (if applicable), etc. than the original one.

5.10.2 Availability

Analytic energies and gradients for CCD and CCSD, numerical gradients for CCSD(T), and numerical frequencies for all methods.

The restricted open-shell (RO) method is available for CCSD and CCSD(T) energy calculations.

5.10.3 Related Keywords

MP4, Transformation

5.10.4 Examples

The Coupled Cluster energy appears in the output as follows following the final correlation iteration. The CCSD energy is given in the first line below, and the final line reports the energy with triples included:

```
Wavefunction amplitudes converged. E(Corr)=     -75.001924366
```

```
 ...
 CCSD(T)= -0.75002048348D+02
```

The CCSD energy is labeled `E(CORR)`, and the energy including the non-iterative triples contribution is given in the final line.

5.11 Charge

The Charge keyword requests that a background charge distribution be included in the calculation. The charge distribution is made up of point charges [196, 197].

5.11.1 Input

By default, the charges are read from the input stream, one per line, in this format:

```
x y z charge
```

where `x`,`y`,`z` are the coordinates in the standard orientation (in the units specified by the Units keyword and defaulting to Angstroms), and `charge` is the charge.

5.11.2 Options

Angstroms

Indicates that input charge locations are specified in Angstroms.

Bohrs

Indicates that input charge locations are specified in Bohrs.

Check

Reads the background charge distribution from the checkpoint file.

5.11.3 Availability

Single point energies, optimizations and frequencies. Not valid with semi-empirical methods or PBC.

5.11.4 Related Keywords

Units

5.11.5 Examples

To perform geometry optimizations in the presence of background charges, you must use Opt=Z-Matrix NoSymm keywords and define the input geometry either in traditional Z-matrix coordinates or symbolic Cartesian coordinates. Here is an example:

```
# B3LYP/6=31G(d) Opt=Z-Matrix Charge NoSymm

Water with point charges

0,1
O -0.75 -0.94 0.0
H  0.21 -0.94 0.0
H -1.07 -0.0  0.0
```

```
2.0  2.0 2.0 1.2
2.0 -2.0 2.0 1.1
```

5.12 ChkBasis

The ChkBasis keyword requests that the basis set be read from the checkpoint file. It is useful in compound jobs involving general basis sets by allowing them to have only one copy of the basis set in the input stream. Note, however, that ChkBasis can be used to retrieve *whatever* basis set exists in a checkpoint file, regardless of how it was originally specified. ECPs specified in the basis set are also retrieved, as are the choices for pure vs. Cartesian functions.

By default, ChkBasis will also retrieve any density fitting basis in the checkpoint file. See the examples for other possibilities.

Of course, no basis set keyword should be specified with ChkBasis.

CheckBasis, CheckPointBasis, ReadBasis, and RdBasis are all synonyms for ChkBasis.

5.12.1 Related Keywords

Gen, GenECP, Pseudo, ExtraBasis, ExtraDensityBasis

5.12.2 Examples

The following route section will retrieve the basis set and density fitting set (if any) from the checkpoint file and use them for the current job:

```
# B3LYP/ChkBasis
```

The following route section will retrieve only the basis set from the checkpoint file, and an automatically generated density fitting basis will be used:

```
# B3LYP/ChkBasis/Auto
```

The following route section will retrieve only the density fitting basis from the checkpoint file:

```
# B3LYP/6-31G(d)/ChkBasis
```

5.13 CID and CISD

These method keywords request a Hartree-Fock calculation followed by configuration interaction with all double substitutions (CID) or all single and double substitutions (CISD) from the Hartree-Fock reference determinant [198–200]. CI is a synonym for CISD.

5.13.1 Options

FC

All frozen core options are available with this keyword; a frozen core calculation is the default. See the discussion of the FC options for full information.

Conver=*N*

Sets the convergence calculations to 10^{-N} on the energy and $10^{-(N-2)}$ on the wavefunction. The default is *N*=7 for single points and *N*=8 for gradients.

MaxCyc=*n*

Specifies the maximum number of cycles for CISD calculations.

SaveAmplitudes

Saves the converged amplitudes in the checkpoint file for use in a subsequent calculation (e.g., using a larger basis set). Using this option results in a very large checkpoint file, but also may significantly speed up later calculations.

ReadAmplitudes

Reads the converged amplitudes from the checkpoint file (if present). Note that the new calculation can use a different basis set, method (if applicable), etc. than the original one.

5.13.2 Availability

Energies, analytic gradients, and numerical frequencies.

5.13.3 Related Keywords

Transformation

5.13.4 Examples

The CI energy appears in the output as follows:

```
DE(CI)=    -.48299990D-01        E(CI)= -.75009023292D+02
NORM(A) =   .10129586D+01
```

The output following the final CI iteration gives the predicted total energy. The second output line displays the value of Norm(A). Norm(A) - 1 gives a measure of the correlation correction to the wavefunction; the coefficient of the HF configuration is thus 1/Norm(A). Note that the wavefunction is stored in intermediate normalization; that is:

$$\Psi^{CISD} = \Psi^0 + \sum_{ia} T_{ia}\Psi(i \to a) + \sum_{iajb} T_{iajb}\Psi(ij \to ab)$$

Wavefunction in Intermediate Normalization

where Ψ^0 is the Hartree-Fock determinant and has a coefficient of 1 (which is what intermediate normalization means). Norm(A) is the factor by which to divide the wavefunction as given above to fully normalize it. Thus:

$$Norm(A) = \sqrt{\left(1 + \sum_{ia} T_{ia}T_{ia} + \sum_{ijab} T_{ijab}T_{ijab}\right)}$$

Fully Normalized Wavefunction

The coefficient of the Hartree-Fock determinant in the fully normalized wavefunction is then 1/Norm(A), the coefficient of singly-excited determinant $\Psi_{i\to a}$ is T_{ia}/Norm(A), and so on.

5.14 CIS

The CIS method keyword requests a calculation on excited states using single-excitation CI (CI-Singles) [201]. This implementation works for both closed-shell and open-shell systems.

CIS jobs can include the Density keyword. Without options, this keyword causes the population analysis to use the current (CIS) density rather than its default of the Hartree-Fock density. Note that Density cannot be used with CIS(D).

An energy range can be specified for CIS excitation energies using some options found under the procedure-related options below.

CIS(D) is used to request the related CIS(D) method (i.e. the D option) [202, 203]. You can also follow a CIS job with a CIS(D) job to compute the excitation energies for additional states (see the examples).

5.14.1 Options

State Selection Options

Singlets

Solve only for singlet excited states. This option only affects calculations on closed-shell systems, for which it is the default.

Triplets

Solve only for triplet excited states. This option only affects calculations on closed-shell systems.

50-50

Solve for half triplet and half singlet states. This option only affects calculations on closed-shell systems.

Root=*N*

Specifies the “state of interest” for which the generalized density is to be computed. The default is the first excited state (*N*=1).

NStates=*M*

Solve for *M* states (the default is 3). If 50-50 is requested, NStates gives the number of each type of state for which to solve (i.e., the default is 3 singlets *and* 3 triplets).

Instead of an integer, Read may be specified as this option’s parameter. In this case, the number of states to compute is read from the input stream. This is typically used in EET calculations.

Add=*N*

Read converged states off the checkpoint file and solve for an additional *N* states. This option implies Read as well. NStates cannot be used with this option.

Energy Range Options

GOccSt=*N*

Generate initial guesses using only active occupied orbitals *N* and higher.

GOccEnd=*N*

Generate initial guesses: if *N*>0, use only the first *N* active occupied orbitals; if *N*<0, do not use the highest |*N*| occupieds.

GDEMin=*N*

Generate guesses having estimated excitation energies $\geq N/1000$ eV.

DEMin=*N*

Converge only states having excitation energy $\geq N/1000$ eV; if *N*=-2, read threshold from input; if *N*<-2, set the threshold to |*N*|/1000 Hartrees.

IFact=*N*

Specify factor by which the number of states updated during initial iterations is increased.

WhenReduce=*M*

Reduce to the desired number of states after iteration *M*.

The default for IFact is $Max(4,g)$ where g is the order of the Abelian point group. The default for WhenReduce is 2. A larger value may be needed if there are many states in the range of interest.

Density-Related Option

AllTransitionDensities

Computes the transition densities between every pair of states.

Procedure- and Algorithm-Related Options

FC

All frozen core options are available with this keyword; a frozen core calculation is the default. See the discussion of the FC options for full information.

Direct

Forces solution of the CI-Singles equation using AO integrals which are recomputed as needed. CIS=Direct should be used only when the approximately $4O^2N^2$ words of disk required for the default (MO) algorithm are not available, or for larger calculations (over 200 basis functions).

MO

Requests that a CIS calculation use transformed integrals. This was the default for CIS in G09 but is never the default in G16.

AO

Forces solution of the CI-Singles equations using the AO integrals, avoiding an integral transformation. The AO basis is seldom an optimal choice, except for small molecules on systems having very limited disk and memory.

Conver=*N*

Sets the convergence calculations to 10^{-N} on the energy and $10^{-(N-2)}$ on the wavefunction. The default is N=4 for single points and N=6 for gradients.

Read

Reads initial guesses for the CI-Singles states off the checkpoint file. Note that, unlike for SCF, an initial guess for one basis set cannot be used for a different one.

Restart

Restarts the CI-Singles iterations off the checkpoint file. Also implies SCF=Restart.

RWFRestart

Restarts the CI-Singles iterations off the read-write file. Useful when using non-standard routes to do successive CI-Singles calculations.

EqSolv

Whether to perform equilibrium or non-equilibrium PCM solvation. NonEqSolv is the default except for excited state optimizations and when the excited state density is requested (e.g., with the Current or All options to the Density keyword).

NoIVOGuess

Forces the use of canonical single excitations for the guess. IVOGuess, which uses improved virtual orbitals, is the default.

NonAdiabaticCoupling

Requests that the ground-to-excited-state non-adiabatic coupling be computed. NAC is a synonym for this option. NoNonAdiabaticCoupling and NoNAC suppress this behavior. The default is NoNAC when computing energies or energies+gradients because the extra cost is non-trivial. The default is NAC during frequency calculations where the extra cost is negligible.

Debugging Options

ICDiag

Forces in-core full diagonalization of the CI-Singles matrix formed in memory from transformed integrals. This is mainly a debugging option.

MaxDiag=*N*

Limits the submatrix diagonalized in the Davidson procedure to dimension *N*. This is mainly a debugging option. MaxDavidson is a synonym for this option.

5.14.2 Availability

Energies, analytic gradients, and analytic frequencies for CIS (including open shell systems), and energies for CIS(D).

5.14.3 Related Keywords

ZIndo, TD, MaxDisk, Transformation, Density

5.14.4 Examples

CIS Output. There are no special features or pitfalls with CI-Singles input. Output from a single point CI-Singles calculation resembles that of a ground-state CI or QCI run. An SCF is followed by the integral transformation and evaluation of the ground-state MP2 energy. Information about the iterative solution of the CI problem comes next; note that at the first iteration, additional initial guesses are made, to ensure that the requested number of excited states are found regardless of symmetry. After the first iteration, one new vector is added to the solution for each state on each iteration.

The change in excitation energy and wavefunction for each state is printed for each iteration (in the #P output):

```
Iteration   3 Dimension    27
Root   1 not converged, maximum delta is    0.002428737687607
Root   2 not converged, maximum delta is    0.013107675296678
Root   3 not converged, maximum delta is    0.030654755631835
Excitation Energies [eV] at current iteration:
Root   1 :     3.700631883679401   Change is   -0.001084398684008
Root   2 :     7.841115226789293   Change is   -0.011232152003400
Root   3 :     8.769540624626156   Change is   -0.047396173133051
```

The iterative process can end successfully in two ways: generation of only vanishingly small expansion vectors, or negligible change in the updated wavefunction.

When the CI has converged, the results are displayed, beginning with this banner:

```
**********************************************************************
   Excited States From <AA,BB:AA,BB> singles matrix:
**********************************************************************
```

The transition dipole moments between the ground and each excited state are then tabulated. Next, the results on each state are summarized, including the spin and spatial symmetry, the excitation energy, the oscillator strength, and the largest coefficients in the CI expansion (use IOp(9/40=*N*) to request more coefficients: all that are greater than 10^{-N}):

```
Excitation energies and oscillator strengths:
```
Symmetry, excitation energy, oscillator strength
```
Excited State   1:   Singlet-A'     3.7006 eV  335.03 nm  f=0.0008
```
CI expansion coeffs. for each excitation:
```
         8 ->  9          0.69112
```
Orbital 8 to 9

```
This state for opt. and/or second-order corr.
```
This is the state of interest.
```
Total Energy, E(CIS) =  -113.696894498
```
CIS energy is repeated here for convenience.

CI expansion coefficients give the importance of excited determinants in the excited state wavefunction.

Normalization. For closed shell calculations, the sum of the squares of the expansion coefficients is normalized to total 1/2 (as the beta coefficients are not shown). For open shell calculations, the normalization sum is 1.

Finding Additional States. The following route will read the CIS results from the checkpoint file and solve for 6 additional states beyond those predicted in the previous calculation:

```
# CIS=(Read,Root=2,Add=6)
```

5.15 CNDO

This method keyword requests a semi-empirical calculation using the CNDO Hamiltonian [204]. No basis set keyword should be specified.

5.15.1 Availability

Energies, pseudo-analytic gradients, and numerical frequencies.

5.15.2 Examples

The CNDO energy appears in the output file as follows:

```
 SCF Done:  E(UCNDO) =  -8.08016620373     A.U. after   11 cycles
```

The energy is as defined by the CNDO model.

5.16 Complex

This keyword allows the molecular orbitals to become complex. It may only be used for closed-shell singlet states.

5.16.1 Availability

Analytic energies for Hartree-Fock and MP2 only, analytic HF gradients, and numerical HF frequencies.

5.16.2 Related Keywords

SCF

5.17 Constants

Specifies which set of physical constants to use. Note that using an older set should only be done in order to compare results with earlier versions of Gaussian.

5.17.1 Options

2010

Constants used in Gaussian 16, taken from [205, 206] and references therein. This is the default.

2006

Constants used in Gaussian 09, taken from [207] and references therein.

1998

Constants used in Gaussian 03 from [208] and references therein.

1986

Constants used in Gaussian 88 through Gaussian 98, from [209, 210].

1979

Constants used in Gaussian 80 through Gaussian 86, mostly from [211].

5.17.2 Current Values

Here are summarized various conversion factors and physical constants used by Gaussian 16 in converting from standard to atomic units. All quantities used in calculations inside Gaussian are in atomic units; the conversion factors are used only when processing input or generating printed output. Values without references are derived from the referenced quantities.

Raw Constants. The constants which are stored directly in the program are:

1 Bohr (a_0)	= 0.52917721092 Å[205, 206]
1 Atomic mass unit (amu, m_u)	= $1.660538921 \times 10^{-27}$ kilograms [205, 206]
1 Electron charge (e)	= $1.602176565 \times 10^{-19}$ Coulombs [205, 206]
	= 4.803204×10^{-10} ESU
Planck's constant (h)	= $6.62606957 \times 10^{-34}$ Joule-secs [205, 206]
Avogadro's number (N_A)	= $6.02214129 \times 10^{23}$ [205, 206]
1 Kcal-mol^{-1}	= 4.184 Joules-mol^{-1} [211]
1 Hartree(E_h)	= $4.35974434 \times 10^{-18}$ Joules [205, 206]
Speed of light (c)	= $2.99792458 \times 10^{10}$ cm-sec^{-1} [205, 206]
Boltzman constant (k)	= $1.3806488 \times 10^{-23}$ Joules-degree^{-1} [205, 206]
Inverse fine structure constant (α^{-1})	= 137.035999074 [205, 206]
Molar volume of ideal gas at 273.15 K (V_m)	= 0.022710953 m^3 [205, 206]
Proton rest mass (m_p)	= $1.672621777 \times 10^{-27}$ kg [205, 206]

Electron magnetic moment (μ_e)	= -9.28476430 $\times$ 10^{-24} J T^{-1} [205, 206]
Free electron g-factor (g_e)	= -2.00231930436153 (dimensionless) [205, 206]

Conversion Factors. The following useful conversion factors are included for your convenience:

Electron mass (m_e)	= 0.910938291 $\times$ 10^{-30} kg [205, 206]
Proton mass (m_p/m_e)	= 1836.15267245 electron mass [205, 206]
1 Atomic mass unit (amu)	= 1.660538921 $\times$ 10^{-27} kg [205, 206]
	= 1822.8885 electron mass
1 Electron volt (eV)	= 1.602176565 $\times$ 10^{-19} Joules [205, 206]
	= 23.06 kcal-mol^{-1}
1 Hartree	= 627.5095 kcal-mol^{-1}
	= 27.2114 eV
	= 219474.63 cm^{-1} (vibrational freq. au)
1 Debye^2-angstrom^{-2}-amu^{-1} (IR intensity unit)	= 42.2561 km-mol^{-1}
	= 5.82573 $\times$ 10^{-3} cm^{-2}-atm^{-1} at STP
Electric field: 1 au	= 5.14220652 $\times$ 10^{11} V-m^{-1} [205, 206]
Electric polarizability: 1 au	= 1.6487772754 $\times$ 10^{-41} C^2-m^2-J^{-1} [205, 206]
1 Bohr-electron (electric dipole moment au)	= 8.47835326 $\times$ 10^{-30} C-m [205, 206]
	= 2.541746 Debye = 2.541746 $\times$ 10^{-18} esu-cm

5.18 Counterpoise

Counterpoise corrections [212, 213] may be computed using the Counterpoise keyword, which can be used in an energy calculation, a geometry optimization, a frequency calculation or a BOMD calculation.

The Counterpoise keyword requires an integer value specifying the number of fragments or monomers in the molecular structure: e.g., Counterpoise=2.

We recommend the new syntax for defining fragments (see Overview of Molecule Specifications), and that is what is used in the examples.

Chapter 9 of *Exploring Chemistry with Electronic Structure Methods* [152, pages 440, 442–44, 454–56] provides an overview of counterpoise corrections, including several examples.

5.18.1 Options

NewGhost

Requests new-style ghost atoms for which integration grid points for DFT quadrature are included. NewBq is a synonym for NewGhost. This is the default and the recommended method.

OldGhost

Requests old-style ghost atoms. OldBq is a synonym for OldGhost. This option is only useful for comparison with previous results.

5.18.2 Availability

Cannot be used with ONIOM or SCRF. Counterpoise calculations cannot produce molecular orbitals.

5.18.3 Examples

Counterpoise Input. The following input is an example of a counterpoise calculation:

```
# UB3LYP/6-31G(d) Counterpoise=2

Counterpoise on water dimer

1,2 1,2 0,1
O(Fragment=1)  0.00  0.00  0.00
O(Fragment=2)  0.00  0.00  2.98
H(Fragment=1)  0.49  0.76 -0.29
H(Fragment=1)  0.49 -0.76 -0.29
H(Fragment=2) -0.91  0.00  3.24
H(Fragment=2) -0.01  0.00  2.03
```

The preceding job also illustrates the use of fragment-specific charge and spin multiplicity specifications. The first pair on the charge and spin line gives the values for the molecule as a whole; they are followed by the charge and spin for each fragment in fragment number order.

Here is an example counterpoise optimization using ECPs:

```
# B3LYP/LANL2DZ Counterpoise=2 NoSymm Opt

HBr + HF, optimization with counterpoise correction using ECP basis

0 1 0 1 0 1
H(Fragment=1)    0.00000000   0.00000000   0.58022808
Br(Fragment=1)   0.00000000   0.00000000  -0.83659185
F(Fragment=2)    0.00000000   0.00000000   2.77788358
H(Fragment=2)    0.00000000   0.00000000   3.69953441
```

Counterpoise Output. Typical output from a Counterpoise calculation follows:

```
Counterpoise corrected energy =    -622.483714884838
                  BSSE energy =       0.005664641553
              sum of fragments =    -622.333463706103
           complexation energy =     -97.84 kcal/mole (raw)
           complexation energy =     -94.28 kcal/mole (corrected)
```

These lines give the counterpoise corrected energy and basis set superposition errors, respectively.

5.19 CPHF

This keyword selects the algorithm used for solving the CPHF equations [214–224].

5.19.1 Options

Frequency-Dependent Calculations

RdFreq

Perform frequency-dependent (dynamic) CPHF, reading in the incident light frequency for the electromagnetic field perturbation. The desired frequency must be provided in the input stream. The default units for this value are Hartrees. Other units may be specified by including a suffix, one of cm (cm^{-1}) and nm (wavelength). This option is relevant for Freq and Polar jobs. It is the default for Freq=ROA.

InputFreq

Read in perturbation frequencies rather than take them from the checkpoint file when doing Geom=AllCheck.

Static

Automatically include the static perturbations when doing dynamic ones. This is the default except for Polar=OptRot and Freq=ROA. NoStatic says not to perform static perturbations in combination with dynamic via RdFreq.

Specifying the Integration Grid

Grid=*grid*

Specify the integration grid for the CPHF portion of the calculation. The syntax is the same as for the Int=Grid option. The argument to this option may be a grid keyword (Fine, UltraFine, and so on) or a specific grid.

The default grid is UltraFine. In this case, the default grid for the CPHF is SG1. When a specific grid is specified to the Int=Grid option, then that grid is also used for the CPHF. Finally, be aware that Fine is used in the CPHF as the default integration grid for a few DFT jobs including Polar=OptRot, Freq=Anharmonic and Freq=NNROA.

See the discussion of Int=Grid for full details on grid specification.

OneStep

DFT nuclear 2*nd* derivatives (ground- or excited-state) should use a grid for CPHF and CPTD one step smaller than the rest of the calculation.

TwoStep

DFT nuclear 2*nd* derivatives (ground- or excited-state) should use a grid for CPHF and CPTD two steps smaller than the rest of the calculation. This is the default.

TauOneStep

DFT 2*nd* derivatives (ground- or excited-state) should use a grid for CPHF and CPTD one step smaller than the rest of the calculation for tau-functionals but two steps smaller for GGAs.

PSCFOneStep

TDDFT 2*nd* derivatives should use a grid for CPHF and CPKS one step smaller than the rest of the calculation, but ground-state frequencies continue to default to 2 steps.

PSCFTauOneStep

TDDFT 2*nd* derivatives should use a grid for CPHF and CPKS one step smaller than the rest of the calculation when using tau functionals, but excited-state frequencies with GGAs and all ground-state frequencies continue to default to 2 steps.

The step options are primarily useful in *Default.Route*

Procedure-Related Options

Conver=*N*

Set the CPHF convergence criterion to 10^{-N}. $N \geq 10$ defaults to CPHF=Separate for the ground-state CPHF/CPKS. *N*>=9 defaults to CPHF=Separate for CPCIS/CPTD.

RecursiveDIIS

Solve reduced equations using recursive DIIS. This is the default when the number of right-hand sides is at least twice the dimension of the reduced matrix and the dimension of the reduced matrix is large (occurs only for ONIOM(MO:MM) using electronic embedding), or the limit set by MaxInv is exceeded. Otherwise, the default is NoRecursiveDIIS, which says to invert the reduced A-matrix.

MaxInv=*N*

Specifies the largest reduced space for in-core inversion during simultaneous solution (up to dimension *N*). Larger reduced problems are solved by a second level of DIIS. The default is 5000.

Simultaneous

Use one expansion space for all variables. This is faster than using separate spaces, but is slightly less accurate. This is the default except when multiple frequencies are specified with RdFreq (see below).

Separate

Use a separate expansion space for each variable in the CPHF (the opposite of Simultaneous). This is the default and only choice when multiple frequencies are specified with RdFreq.

AO

Solve CPHF in the atomic orbital basis [216, 221–223]. This is the default.

MO

Solve in the molecular orbital basis.

Canonical

Canonical CPHF, the default.

MOD

Use MOD orbital derivatives for SAC-CI gradients (which uses configuration selection).

5.19.2 Related Keywords

SCF

5.20 Density

By default, population and other analysis procedures use the SCF density (i.e., the Hartree-Fock density for post-SCF methods; the DFT density for DFT jobs, and the CASSCF density for CAS jobs). The generalized densities for the MP2, MP3, MP4(SDQ), QCISD, CCD, CCSD, CID, and CISD, BD, CIS, TD and SAC-CI methods are available. These are based on the Z-Vector [225–228], and hence yield multipole moments which are the correct analytical derivatives of the energy. The unrelaxed densities at second order (not the same as MP2) can also be used but are not recommended.

The options of the Density keyword select which density to analyze. The Density keyword without an option is equivalent to Density=Current.

5.20.1 Options

Current

Use the density matrix for the current method. This is the default when no option is given to Density.

All

Use all available densities. This is allowed for population analysis but not for electrostatics or density evaluation. Note that this option does not produce densities for all of the excited states in a CI-Singles calculation, only the density for the state of interest (see the examples below for a method of doing the former).

SCF

Use the SCF density. HF is a synonym for SCF.

MP2

Use the generalized density corresponding to the second-order energy.

Transition=*N* or (*N*,*M*)

Use the CIS transition density between state M and state N. M defaults to 0, which corresponds to the ground state.

AllTransition

Use all available CIS transition densities.

CI

Use the generalized density corresponding to the CI energy.

CC

Use the generalized density corresponding to the QCI (or coupled cluster) energy. QCI is a synonym for CC.

RhoCI

Use the one-particle density computed using the CI wavefunction for state N. This is not the same as the CI density [228], and its use is discouraged! Chapter 9 of *Exploring Chemistry with Electronic Structure Methods* discusses this issue [176].

Rho2

Use the density correct to second-order in Møller-Plesset theory. This is not the same as the MP2 density, and its use is discouraged! [228]

CIS=*N*

Use the total unrelaxed CIS density for state N. Note that this is not the same as the density resulting from CIS(Root=N,$\cdots$) Density=Current, which is to be preferred [228].

Checkpoint

Recover the density from the checkpoint file for analysis. Implies Guess=Only ChkBasis: the calculation does not recompute new integrals, SCF, and so on, and retrieves the basis set from the checkpoint file.

5.20.2 Related Keywords

Guess, ChkBasis

5.20.3 Examples

The following route section specifies a TD-DFT calculation which predicts the first six excited states of the molecule under investigation. The population and other analyses will use the TD-DFT density corresponding to the lowest excited state:

```
%Chk=benzene
# TD(NStates=6) B3LYP/6-31+G(d,p) Density=Current Pop=NBO
```

The following route section may be used to rerun the post-TD analyses for another excited state:

```
%Chk=benzene
# TD(Read,Root=3) B3LYP/6-31+G(d,p) Density=Current Pop=NBO
  Guess=Read Geom=AllCheck
```

This route picks up the converged TD density and wavefunction from the checkpoint file, and performs the necessary CPHF calculation to produce the relaxed density for state 3, which is then used in the population and other analyses.

5.21 DensityFit and NoDensityFit

Enables and controls density fitting for the Coulomb problem for calculations involving pure (non-hybrid) DFT functionals. Density fitting basis sets are specified as part of the model chemistry within the job's route section. A list of built-in density fitting sets is included in the discussion of *Basis Sets*. DenFit is a synonym for this keyword.

Including this keyword causes a fitting set to be used if possible for the specified model chemistry. If no specific fitting set is specified in the route section, then the program will select the standard fitting set corresponding to the specified basis set, if one exists, or generate one automatically. When DensityFit is included in the *Default.Route* file, then /Fit is implied for all (relevant) calculations not specifying a fitting set.

The NoDensity keyword turns off the use of density fitting in pure DFT calculations. Since density fitting is off by default in the program, this keyword's only use is to override any DensityFit in *Default.Route*. NoDenFit is a synonym for NoDensityFit.

If both a fitting set and NoDensityFit are included in a job's route section, the latter is ignored, and density fitting *is* used.

5.21.1 Options

Iterative

Controls whether a generalized inverse is formed or the fitting equations are solved iteratively. NonIterative is the default except for ADMP and PBC.

InvToler=*N*

Set the tolerance for a non-trivial eigenvalue of the generalized inverse of the fitting matrix to 10^{-N}.

Convergence=*N*

Specifies 10^{-N} as the convergence criterion for iterative solution of the fitting equations. Implies Iterative. The default is 10^{-6} for ADMP and 10^{-9} for BOMD.

Coulomb

Coulomb specifies the metric to be used for the fitting method. This is the default.

Overlap

Overlap specifies the metric to be used for the fitting method. The default is Coulomb.

JNormalization

Specifies that the read-in density basis has contraction coefficients corresponding to Coulomb normalization. This is the default.

AONormalization

Specifies that the read-in density basis has contraction coefficients corresponding to AO (overlap) normalization. JNormalization is the default.

5.21.2 Availability

Applies only to DFT calculations using pure (non-hybrid) functionals.

5.21.3 Related Keywords

Basis sets, ExtraDensityBasis, Gen, ChkBasis

5.22 DFT (Density Functional Theory) Methods

Gaussian 16 offers a wide variety of Density Functional Theory (DFT) [229–232] models (see also [233–245] for discussions of DFT methods and applications). Energies [246], analytic gradients, and true analytic frequencies [247–249] are available for all DFT models.

The self-consistent reaction field (SCRF) can be used with DFT energies, optimizations, and frequency calculations to model systems in solution.

Pure DFT calculations will often want to take advantage of density fitting. See the discussion in *Basis sets* for details.

The same optimum memory sizes given by freqmem are recommended for DFT frequency calculations.

Polarizability derivatives (Raman intensities) and hyperpolarizabilities are not computed by default during DFT frequency calculations. Use Freq=Raman to request them. Polar calculations do compute them.

Note: The double hybrid functionals are discussed with the MP2 keyword since they have similar computational cost.

Accuracy Considerations

A DFT calculation adds an additional step to each major phase of a Hartree-Fock calculation. This step is a numerical integration of the functional (or various derivatives of the functional). Thus in addition to the sources of numerical error in Hartree-Fock calculations (integral accuracy, SCF convergence, CPHF convergence), the accuracy of DFT calculations also depends on the number of points used in the numerical integration.

The UltraFine integration grid (corresponding to Integral=UltraFine) is the default in Gaussian 16. This grid greatly enhances calculation accuracy at reasonable additional cost. We do not recommend using any smaller grid in production DFT calculations. Note also that it is important to use the *same* grid for all calculations where you intend to compare energies (e.g., computing energy differences, heats of formation, and so on).

Larger grids are available when needed (e.g. tight geometry optimizations of certain kinds of systems). An alternate grid may be selected with the Integral=Grid option in the route section.

5.22.1 Background

In Hartree-Fock theory, the energy has the form:

E_{HF} = V + <hP> + 1/2<PJ(P)> - 1/2<PK(P)>

where the terms have the following meanings:

V	The nuclear repulsion energy.
P	The density matrix.
<hP>	The one-electron (kinetic plus potential) energy.
1/2<PJ(P)>	The classical coulomb repulsion of the electrons.
-1/2<PK(P)>	The exchange energy resulting from the quantum (fermion) nature of electrons.

In the Kohn-Sham formulation of density functional theory [230], the exact exchange (HF) for a single determinant is replaced by a more general expression, the exchange-correlation functional, which can include terms accounting for both the exchange and the electron correlation energies, the latter not being present in Hartree-Fock theory:

E_{KS} = V + <hP> + 1/2<PJ(P)> + E_X[P] + E_C[P]

where E_X[P] is the exchange functional, and E_C[P] is the correlation functional.

Within the Kohn-Sham formulation, Hartree-Fock theory can be regarded as a special case of density functional theory, with E_X[P] given by the exchange integral -1/2<PK(P)> and E_C=0. The functionals normally used in density functional theory are integrals of some function of the density and possibly the density gradient:

$E_X[P] = \int f\left(\rho_\alpha(r), \rho_\beta(r), \nabla\rho_\alpha(r), \nabla\rho_\beta(r)\right) dr$

where the methods differ in which function f is used for E_X and which (if any) f is used for E_C. In addition to pure DFT methods, Gaussian supports hybrid methods in which the exchange functional is a linear combination of the Hartree-Fock exchange and a functional integral of the above form. Proposed functionals lead to integrals which cannot be evaluated in closed form and are solved by numerical quadrature.

5.22.2 Keywords: Hybrid Functionals

A number of hybrid functionals, which include a mixture of Hartree-Fock exchange with DFT exchange-correlation, are available via keywords:

Becke Three-Parameter Hybrid Functionals

These functionals have the form devised by Becke in 1993 [250]:

A*E_X^{Slater} + (1-A)*E_X^{HF} + B*ΔE_X^{Becke} + E_C^{VWN} + C*$\Delta E_C^{non-local}$

where *A*, *B*, and *C* are the constants determined by Becke via fitting to the G1 molecule set.

There are several variations of this hybrid functional.

◊ B3LYP uses the non-local correlation provided by the LYP expression, and VWN functional III for local correlation (not functional V). Note that since LYP includes both local and non-local terms, the correlation functional used is actually:

  C*E_C^{LYP}+(1-C)*E_C^{VWN}

  In other words, VWN is used to provide the excess local correlation required, since LYP contains a local term essentially equivalent to VWN.

◊ B3P86 specifies the same functional with the non-local correlation provided by Perdew 86, and B3PW91 specifies this functional with the non-local correlation provided by Perdew/Wang 91.

◊ O3LYP is a three-parameter functional similar to B3LYP:
$A*E_X^{LSD}+(1-A)*E_X^{HF}+B*\Delta E_X^{OPTX}+C*\Delta E_C^{LYP}+(1-C)E_C^{VWN}$
where A, B and C are as defined by Cohen and Handy in reference [251].

Functionals Including Dispersion

◊ APFD requests the Austin-Frisch-Petersson functional with dispersion [252], and APF requests the same functional without dispersion.

◊ The wB97XD functional uses a version of Grimme's D2 dispersion model.

The standalone keyword EmpiricalDispersion also allows you to specify a dispersion scheme with various functionals.

Long-Range-Corrected Functionals

The non-Coulomb part of exchange functionals typically dies off too rapidly and becomes very inaccurate at large distances, making them unsuitable for modeling processes such as electron excitations to high orbitals. Various schemes have been devised to handle such cases. Gaussian 16 offers the following functionals which include long-range corrections:

◊ LC-wHPBE: Recommended version [253] of the long-range-corrected ωPBE functional [254–256]. LC-wPBE requests the original version.

◊ CAM-B3LYP: Handy and coworkers' long-range-corrected version of B3LYP using the Coulomb-attenuating method [257].

◊ wB97XD: The latest functional from Head-Gordon and coworkers, which includes empirical dispersion [258]. The wB97 and wB97X [259] variations are also available. These functionals also include long-range corrections.

In addition, the prefix LC- may be added to most pure functionals to apply the long correction of Hirao and coworkers [260]: e.g., LC-BLYP.

Other Hybrid Functionals

Functionals from the Truhlar Group

◊ MN15 requests the MN15 [261] functional.

◊ M11 [262], SOGGA11X [263], N12SX [264], and MN12SX [264] request these hybrid functionals from the Truhlar group.

◊ PW6B95 and PW6B95D3 [265].

◊ M08HX: The M08-HX functional [266].

◊ M06 hybrid functional of Truhlar and Zhao [267]. The M06HF [268, 269] and M062X [267] variations are also available.

◊ M05 [270] and M052X [271].

Functionals Employing PBE Correlation

◊ The 1996 pure functional of Perdew, Burke and Ernzerhof [272, 273] as made into a hybrid functional by Adamo [274]. The keyword is PBE1PBE. This functional uses 25% exact exchange and 75% DFT exchange. It is known in the literature as PBE0 [274] and as the PBE hybrid [275].

◊ HSEH1PBE: The recommended version of the full Heyd-Scuseria-Ernzerhof functional, referred to as HSE06 in the literature [253, 276–281].

◊ OHSE2PBE: The initial form of the HS06 functional, referred to as HSE03 in the literature.

◊ OHSE1PBE: The version of the HS06 functional prior to modification to support third derivatives.

◊ PBEh1PBE: Hybrid using the 1998 revised form of PBE pure functional (exchange and correlation) [282].

Becke One-Parameter Hybrid Functionals

The B1B95 keyword is used to specify Becke's one-parameter hybrid functional as defined in the original paper [283].

The program also provides other, similar one parameter hybrid functionals implemented by Adamo and Barone [284]. In one variation, B1LYP, the LYP correlation functional is used (as described for B3LYP above). Another version, mPW1PW91, uses Perdew-Wang exchange as modified by Adamo and Barone combined with PW91 correlation [285]; the mPW1LYP, mPW1PBE and mPW3PBE variations are available.

Revisions to B97

◊ Becke's 1998 revisions to B97 [286, 287]. The keyword is B98, and it implements fit 2c in reference [287].

◊ Handy, Tozer and coworkers modification to B97: B971 [288].

◊ Wilson, Bradley and Tozer's modification to B97: B972 [289].

Functionals with τ-Dependent Gradient-Corrected Correlation

◊ TPSSh: Hybrid functional using the TPSS functionals [290, 291].

◊ tHCTHhyb: Hybrid functional using the tHCTH functional [292].

◊ BMK: Boese and Martin's τ-dependent 2004 hybrid functional [293].

Older Functionals

◊ HISSbPBE requests the HISS functional [294].

◊ X3LYP: Functional of Xu and Goddard [295].

Half-and-Half Functionals

The following functionals, which are included for backward-compatibility only. Note that these are not the same as the "half-and-half" functionals proposed by Becke [296].

◊ BHandH: $0.5{*}E_X^{HF} + 0.5{*}E_X^{LSDA} + E_C^{LYP}$

◊ BHandHLYP: $0.5{*}E_X^{HF} + 0.5{*}E_X^{LSDA} + 0.5{*}\Delta E_X^{Becke88} + E_C^{LYP}$

User-Defined Hybrid Models

Gaussian 16 can use any model of the general form:

$P_2E_X^{HF} + P_1(P_4E_X^{Slater} + P_3\Delta E_x^{non-local}) + P_6E_C^{local} + P_5\Delta E_C^{non-local}$

The only available local exchange method is Slater (S), which should be used when only local exchange is desired. Any combinable non-local exchange functional and combinable correlation functional may be used (as listed previously).

The values of the six parameters are specified with various non-standard options to the program:

◊ IOp(3/76=*mmmmmnnnnn*) sets P_1 to *mmmmm*/10000 and P_2 to *nnnnn*/10000. P_1 is usually set to either 1.0 or 0.0, depending on whether an exchange functional is desired or not, and any scaling is accomplished using P_3 and P_4.

◊ IOp(3/77=*mmmmmnnnnn*) sets P_3 to *mmmmm*/10000 and P_4 to *nnnnn*/10000.

◊ IOp(3/78=*mmmmmnnnnn*) sets P_5 to *mmmmm*/10000 and P_6 to *nnnnn*/10000.

For example, IOp(3/76=1000005000) sets P_1 to 1.0 and P_2 to 0.5. Note that all values must be expressed using five digits, adding any necessary leading zeros.

Here is a route section specifying the functional corresponding to the B3LYP keyword:

```
#P BLYP IOp(3/76=1000002000) IOp(3/77=0720008000) IOp(3/78=0810010000)
```

The output file displays the values that are in use:

```
IExCor=  402 DFT=T Ex=B+HF Corr=LYP ExCW=0 ScaHFX=  0.200000
ScaDFX=  0.800000  0.720000  1.000000  0.810000
```

where the value of ScaHFX is P_2, and the sequence of values given for ScaDFX are P_4, P_3, P_6, and P_5.

5.22.3 Keyword: Pure Functionals

Names for the various pure DFT models are given by combining the names for the exchange and correlation functionals. In some cases, standard synonyms used in the field are also available as keywords. In order to specify a pure functional, combine an exchange functional component keyword with the one for desired correlation functional. For example, the combination of the Becke exchange functional (B) and the LYP correlation functional is requested by the BLYP keyword. Similarly, SVWN requests the Slater exchange functional (S) and the VWN correlation functional, and is known in the literature by its synonym LSDA (Local Spin Density Approximation). LSDA is a synonym for SVWN. Some other software packages with DFT facilities use the equivalent of SVWN5 when "LSDA" is requested. Check the documentation carefully for all packages when making comparisons.

Exchange Functionals

The following exchange functionals are available in Gaussian 16. Unless otherwise indicated, these exchange functionals must be combined with a correlation functional in order to produce a usable method.

- ◊ S: The Slater exchange, $\rho^{4/3}$ with theoretical coefficient of 2/3, also referred to as Local Spin Density exchange [229, 230, 297]. Keyword if used alone: HFS.
- ◊ XA: The XAlpha exchange, $\rho^{4/3}$ with the empirical coefficient of 0.7, usually employed as a standalone exchange functional, without a correlation functional [229, 230, 297]. Keyword if used alone: XAlpha.
- ◊ B: Becke's 1988 functional, which includes the Slater exchange along with corrections involving the gradient of the density [298]. Keyword if used alone: HFB.
- ◊ PW91: The exchange component of Perdew and Wang's 1991 functional [237, 241, 299–301].
- ◊ mPW: The Perdew-Wang 1991 exchange functional as modified by Adamo and Barone [285].
- ◊ G96: The 1996 exchange functional of Gill [302, 303].
- ◊ PBE: The 1996 functional of Perdew, Burke and Ernzerhof [272, 273].
- ◊ O: Handy's OPTX modification of Becke's exchange functional [304, 305].
- ◊ TPSS: The exchange functional of Tao, Perdew, Staroverov, and Scuseria [290].
- ◊ RevTPSS: The revised TPSS exchange functional of Perdew et. al. [306, 307].
- ◊ BRx: The 1989 exchange functional of Becke [308].
- ◊ PKZB: The exchange part of the Perdew, Kurth, Zupan and Blaha functional [309].
- ◊ wPBEh: The exchange part of screened Coulomb potential-based final of Heyd, Scuseria and Ernzerhof (also known as **HSE**) [253, 280, 310].
- ◊ PBEh: 1998 revision of PBE [282].

Correlation Functionals

The following correlation functionals are available, listed by their corresponding keyword component, all of which must be combined with the keyword for the desired exchange functional:

- ◊ VWN: Vosko, Wilk, and Nusair 1980 correlation functional(III) fitting the RPA solution to the uniform electron gas, often referred to as Local Spin Density (LSD) correlation [311] (functional III in this article).
- ◊ VWN5: Functional V from reference [311] which fits the Ceperly-Alder solution to the uniform electron gas (this is the functional recommended in [311]).
- ◊ LYP: The correlation functional of Lee, Yang, and Parr, which includes both local and non-local terms [312, 313].
- ◊ PL (Perdew Local): The local (non-gradient corrected) functional of Perdew (1981) [314].
- ◊ P86 (Perdew 86): The gradient corrections of Perdew, along with his 1981 local correlation functional [315].
- ◊ PW91 (Perdew/Wang 91): Perdew and Wang's 1991 gradient-corrected correlation functional [237, 241, 299–301].
- ◊ B95 (Becke 95): Becke's τ-dependent gradient-corrected correlation functional (defined as part of his one parameter hybrid functional [283]).
- ◊ PBE: The 1996 gradient-corrected correlation functional of Perdew, Burke and Ernzerhof [272, 273].
- ◊ TPSS: The τ-dependent gradient-corrected functional of Tao, Perdew, Staroverov, and Scuseria [290].
- ◊ RevTPSS: The revised TPSS correlation functional of Perdew et. al. [306, 307].
- ◊ KCIS: The Krieger-Chen-Iafrate-Savin correlation functional [316–319].
- ◊ BRC: Becke-Roussel correlation functional [308].
- ◊ PKZB: The correlation part of the Perdew, Kurth, Zupan and Blaha functional [309].

Correlation Functional Variations. The following correlation functionals combine local and non-local terms from different correlation functionals:

- ◊ VP86: VWN5 local and P86 non-local correlation functional.
- ◊ V5LYP: VWN5 local and LYP non-local correlation functional.

Standalone Pure Functionals

The following pure functionals are self-contained and are not combined with any other functional keyword components:

- ◊ VSXC: van Voorhis and Scuseria's τ-dependent gradient-corrected correlation functional [320].
- ◊ **HCTH/***: Handy's family of functionals including gradient-corrected correlation [288, 321, 322]. HCTH refers to HCTH/407, HCTH93 to HCTH/93, HCTH147 to HCTH/147, and HCTH407 to HCTH/407. Note that the related HCTH/120 functional is not implemented.
- ◊ tHCTH: The τ-dependent member of the HCTH family [292]. See also tHCTHhyb below.
- ◊ B97D: Grimme's functional including dispersion [323]. B97D3 requests the same but with Grimme's D3BJ dispersion [324].
- ◊ M06L [325], SOGGA11 [326], M11L [327], MN12L [328] N12 [329] and MN15L [330] request these pure functionals from the Truhlar group.

5.22.4 Empirical Dispersion

The EmpiricalDispersion keyword enables empirical dispersion. It takes the following options:

PFD

Add the Petersson-Frisch dispersion model from the APFD functional [252].

GD2

Add the D2 version of Grimme's dispersion [323]. The table below gives the list of functionals in

Gaussian 16 for which GD2 parameters are defined. The functionals highlighted in bold include this dispersion model by default when the indicated keyword is specified (e.g., B2PLYPD). For the rest of the functionals, dispersion is requested with EmpiricalDispersion=GD2.

Functional	S6	SR6
B97D	1.2500	1.1000
B2PLYPD	0.5500	1.1000
mPW2PLYPD	0.4000	1.1000
PBEPBE	0.7500	1.1000
BLYP	1.2000	1.1000
B3LYP	1.0500	1.1000
BP86	1.0500	1.1000
TPSSTPSS	1.0000	1.1000

The damping function used by this model also contains a D6 parameter with a fixed value of 6.0.
You can use this empirical dispersion method with other functionals by defining the values of the SR6 and S6 parameters (the value of SR6 is always 1.1). This is done using an environment variable with the name GAUSS_DFTD3_S6. The value of the environment variable sets the corresponding parameter to *value*/1,000,000. For example, the command:

```
export GAUSS_DFTD3_S6=1200000
```

sets the value of S6 to 1200000/1000000=1.2.
The wB97XD functional – specified as an independent keyword – uses a version of this dispersion model with values of S6 and SR6 of 1.0 and 1.1, respectively. This functional uses a similar damping function to that used by the GD3 model, with D6 and IA6 having fixed values of 6.0 and 12, respectively.

GD3

Add the D3 version of Grimme's dispersion with the original D3 damping function [331]. The table below gives the list of functionals in Gaussian 16 for which GD3 parameters are defined. For these functionals, dispersion is requested with EmpiricalDispersion=GD3.

Functional	S6	SR6	S8
B2PLYPD3 [332]	0.6400	1.4270	1.0220
B97D3	1.0000	0.8920	0.9090
B3LYP	1.0000	1.2610	1.7030
BLYP	1.0000	1.0940	1.6820
PBE1PBE	1.0000	1.2870	0.9280
TPSSTPSS	1.0000	1.1660	1.1050
PBEPBE	1.0000	1.2170	0.7220
BP86	1.0000	1.1390	1.6830
BPBE	1.0000	1.0870	2.0330
B3PW91	1.0000	1.1760	1.7750
BMK	1.0000	1.9310	2.1680
CAM-B3LYP	1.0000	1.3780	1.2170
LC-wPBE	1.0000	1.3550	1.2790

M05	1.0000	1.3730	0.5950
M052X	1.0000	1.4170	0.0000
M06L	1.0000	1.5810	0.0000
M06	1.0000	1.3250	0.0000
M062X	1.0000	1.6190	0.0000
M06HF	1.0000	1.4460	0.0000

This model also uses an SR8 parameter with a fixed value of 1.0. The damping function used by this model also contains D6, IA6, D8, and IA8 parameters with fixed values of 6.0, 14, 6.0, and 16, respectively.

You can use this empirical dispersion method with other functionals by defining the values of the SR6 and S8 parameters (the value of S6 is always 1.0). This is done using environment variables with names of the form GAUSS_DFTD3_*param*, where *param* is one of the parameter names. The value of the environment variable sets the corresponding parameter to *value*/1,000,000. For example, the command:

```
export GAUSS_DFTD3_S8=1375000
```

sets the value of S8 to 1375000/1000000=1.375.

GD3BJ

Add the D3 version of Grimme's dispersion with Becke-Johnson damping [324]. The table below gives the list of functionals in Gaussian 16 for which GD3BJ parameters are defined. The functionals highlighted in bold include this dispersion model by default when the indicated keyword is specified (e.g., B2PLYPD3). For the rest of the functionals, dispersion is requested with EmpiricalDispersion=GD3BJ.

Functional	**S6**	**S8**	**ABJ1**	**ABJ2**
B2PLYPD3 [332]	0.6400	0.9147	0.3065	5.0570
B97D3	1.0000	2.2609	0.5545	3.2297
B3LYP	1.0000	1.9889	0.3981	4.4211
BLYP	1.0000	2.6996	0.4298	4.2359
PBE1PBE	1.0000	1.2177	0.4145	4.8593
TPSSTPSS	1.0000	1.9435	0.4535	4.4752
PBEPBE	1.0000	0.7875	0.4289	4.4407
BP86	1.0000	3.2822	0.3946	4.8516
BPBE	1.0000	4.0728	0.4567	4.3908
B3PW91	1.0000	2.8524	0.4312	4.4693
BMK	1.0000	2.0860	0.1940	5.9197
CAM-B3LYP	1.0000	2.0674	0.3708	5.4743
LC-wPBE	1.0000	1.8541	0.3919	5.0897

You can use this empirical dispersion method with other functionals by defining the values of the S8, ABJ1 and ABJ2 parameters (the value of S6 is always 1.0). This is done using environment variables with names of the form GAUSS_DFTD3_*param*, where *param* is one of the parameter names. The value of the environment variable sets the corresponding parameter to *value*/1,000,000. For example, the command:

```
export GAUSS_DFTD3_S8=2375000
```

sets the value of S8 to 2375000/1000000=2.375.

5.22.5 Availability

Energies, analytic gradients, and analytic frequencies; ADMP calculations.

Third order properties such as hyperpolarizabilities and Raman intensities are not available for functionals for which third derivatives are not implemented: the exchange functionals G96, P86, PKZB, wPBEh and PBEh; the correlation functional PKZB; the hybrid functionals OHSE1PBE and OHSE2PBE.

5.22.6 Related Keywords

IOp, Int=Grid, Stable, TD, DenFit, B2PLYP, mPW2LYP

5.22.7 Examples

The energy is reported in DFT calculations in a form similar to that of Hartree-Fock calculations. Here is the energy output from a B3LYP calculation:

```
 SCF Done:  E(RB3LYP) =  -75.3197099428     A.U. after    5 cycles
```

5.23 DFTB and DFTBA

Requests a density-functional-based tight-binding semi-empirical calculation, a method which is parametrized via the results of DFT calculations:

◊ DFTB uses the tabulated matrix elements as in the original implementation of Elstner and coworkers [333, 334].

◊ DFTBA is a version that uses analytic expressions for the matrix elements rather than tabulated ones [335].

See [336–340] for review articles and calibration studies.

5.23.1 Options

Read

Read values for parameters from the input stream. This is the default.

ChkParameters

Read parameters from the checkpoint file.

5.23.2 Availability

Energies, gradients, and frequencies.

Analytic second derivatives are available when the analytic DFTB parameters (provided with the program) are used, but are not available when the tabulated parameters from `dftb.org` are used. This is because the linear interpolation that is done using the tabulated parameters give discontinuous first derivatives at first points, so the second derivatives do not always exist. If you want to ignore this potential problem and compute second derivatives using the tabulated parameters, then Freq=Numer must be specified in the job's route section.

Parameter Files

DFTBA is parametrized for all pairs of H, C, N, and O. It is also parametrized for the metals Sc, Ti, Fe, Co, and Ni but only with H, C, N, and O. That is, Fe_5CO and Sc_5CO are supported, but Fe_4ScCO is not.

The DFTB parameter files are copyright by Professor Elstner and must be obtained from him.

5.23.3 Examples

The following input file format runs a DFTBA calculation using the parameter set provided with Gaussian:

```
# DFTBA OPT FREQ

Ala3 DFTB frequencies

0,1
C,0,-4.5929012011,1.0163256276,1.6498020765
O,0,-5.6641782096,0.9622594116,2.2369288649
H,0,-5.788876035,3.2375262156,-2.1703220199
N,0,-4.4446298947,1.4038535552,0.3517633631
```

Molecule specification continues ···

```
@GAUSS_EXEDIR:dftba.prm
```

For DFTB, the same format of parameter files is used as in other programs: one file for each pair of elements, with the order of the two elements being significant. Accordingly, a calculation on H_2CO would use a parameter input section something like this:

```
@cc.prm
@oo.prm
@hh.prm
@co.prm
@oc.prm
@ch.prm
@hc.prm
@oh.prm
@ho.prm
```

The energy appears in the output as follows:

```
 SCF Done:  E(RDFT-SCTBA) =  -33.9465130617     A.U. after   11 cycles
```

5.23.4 Modifying Slater-Koster Files

DFTB Parameter (.skf) Files and Gaussian

The handling of DFTB input files has been modified for compatibility with the files provided by Elstner: HTML data at the end of the file is ignored. Also, multipliers – e.g. 10*1.0 – are now accepted.

Modifying Slater-Koster files (.skf) from dftb.org for use with Gaussian

The first line of each file must be edited to identify the two elements involved. For example, in the file *H-S.skf* the first line should contain the atomic numbers, so it goes from `dftb.org` format:

```
 2.000000000000E-02, 500
```

to Gaussian format:

```
 0.02, 500 1 16   Add atomic numbers for hydrogen and sulfur.
```

The second field in the first line of each file should contain the number of lines in the file containing grid points for the Hamiltonian and overlap integrals. However, these values are often just placeholders of 500 (as above) or 600. Many of the files – especially older ones – actually supply some other number of points. If you count the number of lines between the one with the first point and the last line before `Spline`, this will yield the required value:

```
 2.000000000000E-02,  601 1 6       The next line is line 1.
 0.000000000000E+00  0.000000000000E+00  0.000000000000E+00  0.000000000000E+00
 0.000000000000E+00  0.000000000000E+00  0.000000000000E+00  0.000000000000E+00
-1.668269422592E-05  1.066481969438E-06  0.000000000000E+00  0.000000000000E+00
...
 0.000000000000E+00  0.000000000000E+00  0.000000000000E+00  0.000000000000E+00
 0.000000000000E+00  0.000000000000E+00  4.881740699644E-05 -7.526317479277E-06
Spline                                The preceding line is the last one to count.
30  3.5
```

You would then edit the first line to reflect the actual number of points:

`0.02, 601 1 16` Some of the files involving hydrogens are links rather than separate files because they are the same as the corresponding file with the order of the elements reversed. For example, *H-C.skf* is a link to *C-H.skf*. However, Gaussian expects both files to be provided, differing only in the order of the atomic numbers on the first line.

5.24 EET

Energy transfer from a photoexcited donor molecule to a nearby ground-state acceptor molecule is a process of fundamental interest in many fields of science, including polymer photophysics, surface photochemistry, photochemical synthesis and molecular device engineering. It is usually known as electronic energy transfer (EET) or resonance energy transfer (RET). The fundamental theoretical treatment was presented by Förster in 1948 [341], and it EET analysis computes the excitation energy transfer rate between molecules (or parts of molecules) from the overlap of the fluorescence spectrum of the donor molecule/fragment with the absorption spectum of the acceptor molecule/fragment. However, not all energy transfers are described well by this treatment. Accordingly, there have been many extensions to Förster's theory, beginning with Dexter [342]. In recent years, a variety of new models have built upon these foundations; see [343] for a review.

In Gaussian 16, the EET analysis is a quantum mechanical model for EET based on a DFT description of the wavefunction, incorporating a time-dependent variational approach [344, 345]. EET is available in the gas phase and in solution. Indeed, Förster's original theory recognizes the importance of solvent effects. The implementation in solution in Gaussian 16 is the formulation of Iozzi, Mennucci, Tomasi and Cammi [346], a model that differs from its predecessors (e.g., [347]) in that it incorporates solvent effects by adding the appropriate operators to the Hamiltonian and the linear response equations; in this way, solvation is present in all steps of the quantum mechanical calculation [348–351]. The solvation cavity for this model is the same for other employments of IEFPCM [352–354] (rather than a simplistic sphere or multipolar expansion).

The EET keyword performs an excitation energy transfer calculation using the results of the CIS/TDA/TD-DFT calculation or of an EOM-CCSD calculation. This type of calculation uses the same setup of Guess

(Fragment=···) but it can also process ONIOM-like link-atom input information to cap the fragments. An excited-state calculation is performed on each fragment, and all the coupling among all the resulting states are computed. Solvent effects can be introduced using PCM and a single cavity, or a fragment-pair cavity can be used to evaluated the solvent-mediated coupling.

5.24.1 Options

Fragment=*N*

Set up *N* fragments.

FullSystemCavity

Use the PCM cavity for the whole system when evaluating the EET coupling. This is the default.

FragmentCavity

Set up a PCM cavity for each pair of fragments when evaluating the EET coupling.

NonEqSolv

Force the use of non-equilibrium solvation for the solvent-mediated term of the coupling.

EqSolv

Force the use of equilibrium solvation for the solvent-mediated term of the coupling.

The default choice of equilibrium vs non-equilibrium is such that the EET is consistent with what has been done in L913/L914.

5.24.2 Examples

The following simple input file performs an EET calculation on formaldehyde dimer, treating each molecule as a separate fragment. This job models the EET for a single excited state.

```
# td(nstates=1) b3lyp/6-31G(d) eet(fragment=2)

EET in gas phase (Closed shell fragments done as closed shell)
Full   : H2CO ... H2CO
Frag 1: H2CO
Frag 2:          H2CO

0 1 0 1 0 1
C(fragment=1)       0.000000   -0.542500    0.000000
O(fragment=1)       0.000000    0.677500    0.000000
H(fragment=1)       0.000000   -1.082500    0.935307
H(fragment=1)       0.000000   -1.082500   -0.935307
C(fragment=2)       2.000000   -0.542500    0.000000
O(fragment=2)       2.000000    0.677500    0.000000
H(fragment=2)       2.000000   -1.082500    0.935307
H(fragment=2)       2.000000   -1.082500   -0.935307
```

Here is the key part of the EET output:

```
 ===============================================================================

  Electronic Coupling for Excitation Energy Tranfer
```

```
===============================================================================
Separate basis set information will be generated for each pair of fragments.
Using analytical method for overlap contributions to EET.
```

The minimum distance is a measure of how far the electron moves.

```
Frag=  2  1 min distance between atoms 5 1 = 2.00 Angs (sum of Rcov=1.520)
                   excluding hydrogens 5 1 = 2.00 Angs (sum of Rcov=1.520)
```

End of next line gives value of ω for these two fragments & states.

```
Frag=  2 State=  1 (w=  3.9574 eV)  Frag=  1 State=  1 (w=  3.9574 eV)

  delta-w                     =  0.000000000 eV
  Coulomb                     = -0.035983294 eV
  Exact-exchange              =  0.014046399 eV
  Exchange-correlation        =  0.006375024 eV
  w-avg*Overlap               = -0.000357708 eV (w-avg=3.957 eV, Ovlp=-0.90389D-04)
  Total coupling              = -0.015919579 eV
```

Energy difference between the two states. (delta-w)

Parameter used to compute the rate of EET. (Total coupling)

This output will be repeated for each interaction requested by the calculation.

5.25 EOMCCSD

Requests an excited state calculation using the EOM-CCSD method [355–365]. EOM-CCSD is an extension of CCSD for modeling excited states. It provides CCSD-level accuracy for excited-state calculations and requires comparable computational cost (scaling as N^6 like CCSD) and additional disk space. This method uses a preliminary CIS calculation to generate the initial guess for the states followed by an EOM-CCSD analysis.

Note: The EOM-CCSD method exploits abelian symmetry (and not higher point groups).

The various solvation methods for EOM developed by Caricato [360, 366, 367] are available; see the SCRF=PTED option for details.

5.25.1 Options

State Selection and Specification

NStates=*N*

Try to solve for the lowest *N* states in EOM. It is a good idea to set *N* to be larger than the desired number of states to take account of likely state reordering between the CIS and EOM portions.

NStPIR=*K*

Number of states per symmetry type to solve for in the EOM. The default is 2. Note that the symmetry types correspond to the largest abelian subgroups. If *K* is less than zero, then a separate blank line-terminated input section is read specifying the number of states for each symmetry type (irreducible representation). The symmetry ordering can be determined quickly by running a preliminary job with the %KJob L301 Link 0 command. We recommend that you also specify NCISState with a reasonable number of states for the CIS (see below).

Only one of NState and NStPIR should be used to specify the desired number of states. If both are specified, then NState takes precedence. If nothing is specified, then NStPIR=2 is the default.

Singlets

Solve for singlet excited states. This option only affects calculations on closed-shell systems, for which it is the default.

Triplets

Solve for triplet excited states. This option only affects calculations on closed-shell systems. Must be combined with Singlets to solve for both kinds of states.

NCISState=*M*

Total number of states to be generated as guesses by CIS. The default with NState is *N*Irr.Reps.*; with NStPIR, it is *(K+2)*Irr.Reps.*

Root=*N*

Specifies the state of interest. The default is the first excited state (*N*=1).

Procedure-Related Options

MaxCyc=*N*

Specifies the maximum number of cycles for the calculation.

Convergence=*N*

Sets the convergence calculations to 10^{-N} on the energy and $10^{-(N-2)}$ on the wavefunction. The default is *N*=7.

CCConvergence=*N*

Use 10^{-N} as the convergence on the CCSD and ground-state Z-vector iterations. CCSDConvergence is a synonym for this option. The default is *N*=8.

LRTransitionDensities

Requests linear response transition densities [355, 357, 358] in addition to EOM-style (unrelaxed) ones. This formalism is more rigorous than the default EOM-CCSD, but it is also computationally more expensive. Note that the two formalisms are equivalent when CCSD provides the exact wavefunction (i.e., the two electron system). Applies only to singlet closed shell and open shell systems.

EnergyOnly

Save time by computing only right eigenvectors, which are sufficient for excitation energies but not for transition densities.

Reading/Saving Amplitudes

Amplitudes are saved by default for use in a subsequent calculation. They may be optionally read-in from a previous calculation. The number of states can be increased in the subsequent calculation. The CIS for the guess also reads in vectors and automatically adds states if more guesses are required (provided there is no change in the basis set).

TWInCore

Forces the program to store amplitudes and products in memory during higher-order post-SCF calculations. The default is to do so if possible, but to run off disk if memory is insufficient. TWInCore causes the program to terminate if these can not be held in memory, while NoTWInCore prohibits in-memory storage.

SaveAmplitudes

Saves the converged amplitudes in the checkpoint file for use in a subsequent calculation (e.g., using a larger basis set). Using this option results in a very large checkpoint file, but also may significantly speed up later calculations.

ReadAmplitudes

Reads the converged amplitudes from the checkpoint file (if present). Note that the new calculation can use a different basis set, method (if applicable), etc. than the original one.

ReadGroundStateAmplitudes

Reads in only the ground-state (and Z-vector) amplitudes and not the excited state amplitudes. This option is useful when going from an EOM calculation on singlets to one on triplets. ReadGSAmplitudes is a synonym for this option.

NewCIS

Do a new CIS calculation from scratch when reading EOM amplitudes. This option is needed when reading in singlet states but calculating both singlets and triplets. It is also needed when using a different basis set than was used for a prior calculation retrieved with ReadAmplitudes.

5.25.2 Availability

Energies and gradients.

5.25.3 Examples

Using EOM-CCSD. It is often useful to perform a preliminary, smaller EOM-CCSD calculation which solves for a large number of states, and then run a more accurate calculation on the states of interest. The following route sections illustrate this approach:

First calculation:

```
%Chk=my_eom
# EOMCCSD(NStates=10,EnergyOnly)/Aug-CC-PVDZ
```

Second calculation:

```
%Chk=my_eom
# EOMCCSD(NStates=2,ReadAmplitudes,NewCIS)/Aug-CC-PVQZ
```

Here is some example output from an EOM-CCSD calculation. This header introduces the results section:

```
 ==============================================


        EOM-CCSD transition properties


 ==============================================
```

Next comes the transition electric dipole moment, separated into left and right sections. The dipole and oscillator strengths reported at the end of each line are identical in the two sections as the former is the product of the two:

```
 Ground to excited state transition electric dipole moments (Au):
      state          X           Y           Z          Dip. S.      Osc.
        1         0.0000      0.0000     -0.3969      0.1601      0.0614
        2         0.0000      0.3963      0.0000      0.1638      0.0756
        3         0.0000      1.3681      0.0000      1.9183      1.0604
 Excited to ground state transition electric dipole moments (Au):
      state          X          Y          Z         Dip. S.     Osc.
        1         0.0000     0.0000    -0.4034      0.1601     0.0614
        2         0.0000     0.4133     0.0000      0.1638     0.0756
        3         0.0000     1.4022     0.0000      1.9183     1.0604
```

For each state, a separate section lists the CI expansion coefficients for excitation along with the corresponding orbital abelian symmetry type, divided by left and right, and then by excitation type:

```
Excited State   1:      Singlet-A1    15.6603 eV   79.17 nm  f=0.0614
 Right Eigenvector
 Alpha Singles Amplitudes
     I     SymI     A     SymA     Value
     4       1      6       1     0.675597     Excitation from orbital 4 (occ.) to 6 (virt.).
     3       4      7       4     0.122684
 Beta  Singles Amplitudes
     I     SymI     A     SymA     Value
     4       1      6       1     0.675597
     3       4      7       4     0.122684
 Alpha-Beta  Doubles Amplitudes    Similar information for a double excitation.
     I     SymI     J     SymJ     A     SymA     B     SymB     Value
     4       1      4       1      6       1      6       1    -0.118378
 Left  Eigenvector
 Alpha Singles Amplitudes
     I     SymI     A     SymA     Value
     4       1      6       1     0.676418
     3       4      7       4     0.121856
 Beta  Singles Amplitudes
     I     SymI     A     SymA     Value
     4       1      6       1     0.676418
     3       4      7       4     0.121856
 Alpha-Beta  Doubles Amplitudes
     I     SymI     J     SymJ     A     SymA     B     SymB     Value
     4       1      4       1      6       1      6       1    -0.107806
 Total Energy, E(EOM-CCSD) =  -74.4340926881   Total energy reported for state of interest.
```

5.25.4 Related Keywords

CCSD, CIS, SAC-CI, SCRF

5.26 EPT

This method keyword requests an electron propagator theory [368] calculation of correlated electron affinities and ionization potentials [369–383]. Gaussian 16 includes the renormalized partial third order approximation – P3+ – method of Ortiz [384]. It also includes algorithmic improvements for significant speedup of the diagonal, second-order self-energy approximation (D2) component of composite electron propagator (CEP) methods as described in [385]. These models combine a relatively inexpensive D2-level calculation using a large basis set (e.g., augmented quadruple or triple zeta) with a more expensive P3+ or OVGF calculation with a smaller basis set (e.g., triple or double zeta) to produce high accuracy predictions.

For reviews of EPT methods and applications, see [343, 386].

EPT calculations default to storage of <ia||bc> integrals, but can be run with Trans=Full to save CPU time at the expense of disk usage or with Trans=IJAB to save on disk space at the expense of CPU time. In the latter case, electron affinities are not computed.

By default, only ionization potentials which are < 20 eV are computed.

Use the ReadOrbitals option to specify the starting and ending orbitals to refine as input.

OVGF is a synonym for this keyword.

5.26.1 Options

OVGF

Use the Outer Valence Green's Function propagator. This is the default.

P3

Use the P3 and P3+ propagators.

OVGF+P3

Use both propagator methods.

D2

Perform the second-order electron propagator using code that is very efficient for this case. D2 does both attachment (electron affinities) and detachment (ionization potentials) from all orbitals, while the D2IA does only ionization from active (non-frozen-core) orbitals, (which is even less expensive). Often used as part of a compound method in conjection with a higher-order EPT calculation using a smaller basis set. EP2 and EP2IA are synonyms for these options (respectively).

FC

All frozen core options are available with this keyword; a frozen core calculation is the default. See the discussion of the *FC options* for full information.

ReadOrbitals

Specify starting and ending orbitals to refine, in a separate, blank-terminated input section. For unrestricted calculations, separate ranges are specified for alpha and beta orbitals (on the same input line). Orbital numbering starts with the first active orbital after the frozen core. For example, using the default frozen core for ethene, the input 3 8 will skip the two 2s valence orbitals (in addition to the two frozen 1s cores), resulting in the HOMO being labeled as orbital 6 in the output.

ForceSort

Forces sorting of intermediate quantities to be done even when it is not necessary. This option appears in some Gaussian test jobs, but it is not useful for production calculations.

5.26.2 Availability

Single-point energy calculations using HF for the reference method.

5.26.3 Examples

For EPT=OVGF calculations, the results for each orbital appear as follows:

```
3 OVGF renormalized results based on the 3rd order
Method   Orbital  HF-eigenvalue 3rd-order Pole strength
  A:        2    -14.81277     -14.16283    0.93047    OVGF-A results
  B:        2    -14.81277     -14.26825     0.92594    OVGF-B results
  C:        2    -14.81277     -14.28318     0.92723    OVGF-C results

Summary of results for alpha spin-orbital   2  OVGF:
Koopmans theorem:            -0.54436D+00 au  -14.813 eV
Converged second order pole: -0.51494D+00 au  -14.012 eV  0.921 (PS)
Converged third  order pole: -0.52613D+00 au  -14.317 eV  0.929 (PS)
OVGF-von Niessen (OVGF-N) result:
Outer Valence Approximation: -0.52435D+00 au  -14.268 eV  0.926 (PS)
```

The second section gives the estimate of ionization potential/electron affinity for the specified orbital (which property is given depends on whether the orbital is occupied or not, respectively) with the specified propagator. The pole strength is a measure of how easy it is to make this excitation, with 1.0 as the maximum value. Note that orbitals are listed in the output in order of symmetry (and not necessarily in numerical order).

For EPT=P3 calculations, the results for each orbital appear as follows:

```
 Summary of results for alpha spin-orbital   5    P3:
 Koopmans theorem:             0.16053D+00 au    4.368 eV
 Converged second order pole:  0.13657D+00 au    3.716 eV  0.978 (PS)
 Converged 3rd order P3 pole:  0.13578D+00 au    3.695 eV  0.974 (PS)    P3 propagator result
 Renormalized (P3+)  P3 pole:  0.13584D+00 au    3.696 eV  0.974 (PS)    P3+ propagator result
```

Results are given for both the original and renormalized formulations of P3.

For EPT=D2 calculations, the results for each orbital appear as follows:

```
 Converged 2nd order pole for alpha spin-orbital 2  -0.85777 au  -23.341 eV
 Converged 2nd order pole for alpha spin-orbital 3  -0.51494 au  -14.012 eV
 Converged 2nd order pole for alpha spin-orbital 4  -0.51494 au  -14.012 eV
 Converged 2nd order pole for alpha spin-orbital 5  -0.51494 au  -14.012 eV
 ...
```

5.27 External

Requests a calculation using an external program. This mechanism is primarily intended to facilitate the use of external programs to provide the low-level calculations in ONIOM calculations, but can also be used to conduct geometry optimizations using Gaussian's optimizer with external programs providing the function values and derivatives.

Gaussian uses a standardized interface to run an external program to produce an energy (and optionally a dipole moment or forces) at each geometry. A text file is produced with the current structure, and a script named *Gau_External* is run by default (see below for information on specifying an alternate script). This script, which is provided by the user, is expected to:

◊ Convert the text file – referred to as the "input file" – into input for another program.

◊ Run that program.

◊ Convert the results into a standard text form for recovery by Gaussian. The converted file for use by Gaussian is referred to as the "output file."

You may specify a different script by including its name as the option to the External keyword: e.g., External=MyScript.

5.27.1 Script Invocation

By default, the *Gau_External* script is passed six parameters:

$ Gau_External *layer InputFile OutputFile MsgFile FChkFile MatElFile*

The parameters are defined as follows:

layer	A key letter indicating whether the computation is being performed on the real system (R), the model system of a 2-layer ONIOM or the middle layer of a 3-layer ONIOM (M), or the model system of a 3-layer ONIOM (S).
InputFile	The name of the file Gaussian has prepared as input for the external program.
OutputFile	The name of the file which should be read in after the external program completes.
MsgFile	The name of a file for messages; if the script creates this file, then its contents are copied to the Gaussian output file.
FChkFile	A formatted checkpoint file. If the appropriate options are set to link 402, then this file is created from the read-write file before starting the external script, and may be read to import results after the script finishes instead of Gaussian input being provided via *OutputFile*. The output formatted checkpoint file can contain an initial two blank lines plus the data to be updated in the usual format; it does not need to contain any information which is to remain unchanged.
MatElFile	Matrix element file. This is a Fortran unformatted file designed to export data such as the overlap and Core Hamiltonian matrix and two-electron integrals in an extensible format. The structure is documented in *Interfacing to Gaussian 16*.

All of these files are deleted by Gaussian once the results have been recovered.

Additional arguments to the script may also be included:

External="RunTink Amber"

In this example, the actual command would be:

$ RunTink Amber *layer InputFile OutputFile MsgFile FChkFile MatElFile*

The specified script is always passed the parameters mentioned above as its final six arguments.

5.27.2 File Formats

Input File Format

The input file has the following format:

#atoms derivatives-requested charge spin

atomic# x y z MM-charge *Repeated for each atom.*

The first line specifies the number of atoms in the molecule, what derivatives are to be computed (0=energy only, 1=first derivatives, 2=second derivatives), and the molecule's charge and spin multiplicity (format 4I10). The remaining lines specify the atomic number, coordinates, and molecular mechanics charge for each atom (format I10, 4F20.12).

Output File Format

The output file is in fixed format and has the following data (all in atomic units):

Items **Pseudo Code** **Line Format**

energy, dipole-moment (xyz)	E, Dip(I), I=1,3	4D20.12
gradient on atom (xyz)	FX(J,I), J=1,3; I=1,NAtoms	3D20.12
polarizability	Polar(I), I=1,6	3D20.12
dipole derivatives	DDip(I), I=1,9*NAtoms	3D20.12
force constants	FFX(I), I=1,(3*NAtoms*(3*NAtoms+1))/2	3D20.12

The second section is present only if first derivatives or frequencies were requested, and the final section is present only if frequencies were requested. In the latter case, the Hessian is given in lower triangular form: α_{ij}, i=1 to N, j=1 to i. The dipole moment, polarizability, and dipole derivatives can be zero if none are available.

5.27.3 Options

It is also possible to provide one-electron or one- and two-electron integrals and other matrix elements to an external program and to recover results such as MOs or densities from the other program. Full details and examples are in the *g16/doc* subdirectory (*doc* folder on Windows). Options must follow the name of the script. Test job 769 serves as an example of some of these options.

InUnf

A Fortran unformatted file will be provided to the external program containing coordinates and one-electron matrix elements (overlap, core Hamiltonian, etc.). Refer to *g16/doc/unfdat.txt* for details on the contents of the file and to *g16/doc/rdmat.F* for a sample program which reads the file and prints its contents. 1Elintegrals is a synonym for this option.

2ElIntegrals

The Fortran unformatted file should also contain two-electron integrals. This option implies SCF=Conventional.

InputFchk

A formatted checkpoint file should be generated and provided to the external program.

OutputUnf

A Fortran unformatted file will be provided to the external program and an updated or replaced file with the same structure will be read by G16 for the results, in lieu of the default text output file expected from the external program/script.

IOFchk

A formatted checkpoint file will be generated and provided to the external program, and a new *.fchk* file will be read to import results afterwards.

ReadInputSection

This option can be used to alter the content of the external text input file that Gaussian 16 automatically generates for the external script. When the data transfer between Gaussian 16 and the external script is handled using one of the options above (e.g. IOFchk), the default external text input file is not needed. With this option, a section (delimited by the usual blank lines) will be read from the Gaussian 16 input file. The text in this section will be placed in the external text input file instead of the usual content of such file. This provides additional flexibility to provide extra instructions to the external script.

Files

Include the contents of the specified internal Gaussian file within the generated matrix element file. For example, the following option:

```
External=Files=(123,(456,offset=1,integer=27))
```

will cause the contents of internal file 123 (assumed by default to be real values) to be included in the matrix element file (labeled as “File 123”). The second item in the file list specifies the 27 integers in internal file 456 starting after the first (8-byte) word (labeled as “File 456 integers section 001”), as well as any real values following the integers (labeled “File 456 reals section 002”).

5.27.4 Related Info

External scripts may also be specified as one of the models for the ONIOM keyword (see the examples).

The Gaussian stand-alone mm utility program can be run with the **-external** switch, which causes it to read and write data in the formats used by the External interface.

See *Interfacing to Gaussian 16* for information about exchanging data with other programs.

5.27.5 Examples

The following route section specifies an external script for the low layer of a 3 layer ONIOM calculation:

```
# ONIOM(B3LYP/6-31G(d):AM1:External="RunTink Amber") Opt
```

The following route section specifies an external script for the high accuracy layer of a 2 layer ONIOM job:

```
# ONIOM(External="RunCC SDT":B3LYP/6-31G(d)) Opt
```

5.28 ExtraBasis & ExtraDensityBasis

These keywords indicate that additional basis functions are to be added to the basis set or density fitting basis set specified in the route section for the calculation (respectively). These basis functions appear in a separate section in the input stream, using any of the valid formats (which are described in detail in the discussion of the Gen keyword).

ExtraBasis is most useful for supplying basis functions for elements undefined in a standard basis set. It cannot be used to replace a definition within a built-in basis set, and attempting to do so will result in an error. All basis functions specified with this keyword are added to the ones in the basis set specified in the route section. For these reasons, Gen is often easier to use than ExtraBasis; consult the description for that keyword before deciding to use this one.

ExtraDensityBasis is ignored if no density fitting basis is specified in the route.

5.28.1 Related Keywords

Gen, Pseudo, GenECP, GFInput, GFPrint

5.28.2 Example

The following job uses the 6-31G(d,p) basis set along with an additional diffuse function on all of the carbon atoms:

```
# HF/6-31G(d,p) ExtraBasis ···
```

title section

molecule specification

```
C 0
SP 1 1.00
 0.4380000000D-01  0.1000000000D+01  0.1000000000D+01
****
```

The following job supplies additional functions for both the basis set and for density fitting:

```
#p rblyp/6-31g*/dga1 extrabasis extradensitybasis 6d

HCl using the internally stored 6-31g* AO basis & DGA1 fitting set,
adding f functions to the AO basis, and f & g fitting functions

0,1
cl
h,1,1.29

! here are some extra AO polarization functions
cl 0       Additional basis set functions
 F   1 1.00       0.000000000000
      0.7500000000D+00  0.1000000000D+01
 ****
h 0
 p   1 1.00       0.000000000000
      0.1612777588D+00  0.1000000000D+01
 ****

! here are some extra fitting functions.
cl 0      Additional density fitting functions
f 1
  1.5
g 1
  1.5
 ****
h 0
spd 1
 0.32
 ****
```

The following job reads in extra basis set data from an external file:

```
# B3LYP/6-31++G(d) ExtraBasis

B3LYP/6-31++G(d) with extra diffuse functions
```

molecule specification

```
@tripleplus.gbs
```

5.29 Field

The Field keyword requests that a finite field be added to a calculation. In Gaussian, the field can either involve electric multipoles (through hexadecapoles) or a Fermi contact term. Field requires a parameter in one of these two formats: $M \pm N$ or *F(M)N*, where *M* designates a multipole, and *F(M)* designates a Fermi contact perturbation for atom *M* (following the ordering in the molecule specification section of the input file). *N**0.0001 specifies the magnitude of the field in atomic units in the first format and specifies the magnitude of the Fermi contact perturbation in the second format.

Thus, Field=X+10 applies an electric dipole field in the X direction of 0.001 au, while Field=XXYZ-20 applies the indicated hexadecapole field with magnitude 0.0020 au and direction opposite to the default (which is determined by the standard orientation). Similarly, Field=F(3)27 applies a perturbation of 0.0027 times the spin density on atom 3.

Note that the coefficients are those of the Cartesian operator matrices; care must be taken regarding the choice of sign convention when interpreting the results.

All parameters are in the input orientation.

The field specification parameter may be placed among any other options as desired. Archiving is disabled when Field is specified.

5.29.1 Options

Read

Reads the coefficients of 34 electric multipole components from the input stream in free format.

OldRead

Reads the coefficients of 35 electric multipole components from the input stream, in the old style format (including the monopole term): using format 3D20.10 (the first component is a charge).

RWF

Takes the 35 multipole components from the read-write file.

ERWF

Extracts only the three electric dipole field components from the read-write file.

Checkpoint

Reads the 35 multipole components from the checkpoint file. Chk is a synonym for Checkpoint. Checkpoint is the default with Geom=Check.

NoChK

Prevents the reading of the field from the checkpoint file.

EChk

Extracts only the three electric dipole field components from the checkpoint file.

5.29.2 Availability

Single-point energy, geometry optimizations, frequencies, and Force and Scan calculations.

5.29.3 Limitations

Note that if symmetry is left on during a GVB calculation, the finite field may not lead to correct numerical derivatives if the selected field breaks molecular symmetry. To be safe, use Guess=NoSymm whenever using Field with GVB.

In general, if the electric field causes the wavefunction to have different symmetry than the original molecule, incorrect numerical derivatives can result. Accordingly, you might want to use NoSymm when doing numerical derivatives with Field.

5.29.4 Examples

To perform geometry optimizations in the presence of an electric field, you must use Opt=Z-Matrix NoSymm keywords and define the input geometry either in traditional Z-matrix coordinates or symbolic Cartesian coordinates. Here is an example using a Z-matrix:

```
# RHF/3-21G Field=x+60 Opt=Z-Matrix NoSymm

Z-Matrix optimization

 0  1
 C
 H   1   B1
 H   1   B2    2   A1
 H   1   B3    2   A2    3    D1
 H   1   B4    2   A3    3    D2

   B1             1.070000
   B2             1.070000
   B3             1.070000
   B4             1.070000
   A1           109.471203
   A2           109.471203
   A3           109.471231
   D1           120.000015
   D2          -119.999993
```

Here is an example using symbolic Cartesian coordinates:

```
# HF/6-31G(d) Opt=Z-Matrix Field=z-50 NoSymm

Symbolic Cartesian coordinates optimization
```

```
0  1
O 0  x1 y1 z1
H 0  x2 y2 z2
H 0  x3 y3 z3

x1=0.0
y1=0.0
z1=0.12
x2=0.0
y2=0.75
z2=-0.46
x3=0.0
y3=-0.75
z3=-0.46
```

5.30 FMM

Force the use of the fast multipole method [280, 387–394] if possible. The use of FMM is automated in Gaussian 16. Gaussian 16 generally turns on the FMM facility when using it provides even a modest performance gain (say, 1.2x). FMM is enabled for nonsymmetric molecules with 60 atoms or more for both Hartree-Fock and DFT. For molecules with high symmetry, FMM is enabled for Hartree-Fock and hybrid DFT above 240 atoms and for pure DFT above 360 atoms. For molecules with low (but non-zero) symmetry, intermediate thresholds are used. You will begin to see substantial performance improvements (2x or better) for systems that are twice as large.

Of course, the exact results will vary from case to case (compact systems show the least speedup; stretched out linear ones the most), but the defaults are very unlikely to enable FMM when it has a negative effect on performance and are also as unlikely to fail to enable it when it would be worth a factor of 1.5x or more. Thus, users are unlikely to need to control FMM by hand except for some very unusual special cases, such as nearly linear polypeptides and long carbon nanotubes.

The options to FMM are described in *Program Development-Related Keywords*.

5.30.1 Availability

Energies, gradients and frequencies for HF, pure and hybrid DFT. This keyword may also be used within method specifications for ONIOM layers.

5.31 Force

This calculation-type keyword requests a single calculation of the forces on the nuclei (i.e., the gradient of the energy). The dipole moment is also computed (as a proper analytic derivative of the energy for MP2, CC, QCI and CI) [200, 228].

5.31.1 Options

EnOnly

Compute the forces by numerically differentiating the energy once. It is the default for all methods for which analytic gradients are unavailable. Note that this procedure exhibits some numerical instability, so care must be taken that an optimal step size is specified for each case.

Restart

Restarts numerical evaluation of the forces.

StepSize=*N*

Sets the step size used in numerical differentiation to 0.0001**N*. The units are Angstroms by default unless Units=Bohr has been specified. The default step size is 0.01 Å. StepSize is valid only in conjunction with EnOnly.

NoStep

Can be used with large MM force calculations to avoid the O(N^3) work involved in computing the putative geometry optimization step.

5.31.2 Availability

Analytic gradients are available for all SCF wavefunctions, all DFT methods, CIS, MP2, MP3, MP4(SDQ), CID, CISD, CCD, CCSD, QCISD, BD, CASSCF, SAC-CI and all semi-empirical methods. For other methods, the forces are determined by numerical differentiation.

5.31.3 Examples

The forces on the nuclei appear in the output as follows (this sample is from a calculation on water):

```
***** AXES RESTORED TO ORIGINAL SET *****
-------------------------------------------------------------------
Center     Atomic                   Forces (Hartrees/Bohr)
Number     Number              X               Y              Z
-------------------------------------------------------------------
1          8           -.049849321     .000000000    -.028780519
2          1            .046711997     .000000000    -.023346514
3          1            .003137324     .000000000     .052127033
-------------------------------------------------------------------
  MAX      .052127033     RMS      .031211490
-------------------------------------------------------------------
  Internal Coordinate Forces (Hartree/Bohr or radian)
Cent Atom N1     Length/X     N2     Alpha/Y      N3     Beta/Z   J
-------------------------------------------------------------------
1  O
2  H     1   -.023347(   1)
3  H     1   -.023347(   2)  2   -.088273(   3)
-------------------------------------------------------------------
   MAX      .088272874     RMS      .054412682
```

The forces are determined in the standard orientation, but are restored to the original (Z-matrix) set of axes before printing (as noted in the output). This is followed by the corresponding derivatives with respect to the internal coordinates (lengths and angles used in the Z-matrix) when internal coordinates are in use. The forces are followed in each case by their maximum and root-mean-square values.

5.32 Freq

This calculation-type keyword computes force constants and the resulting vibrational frequencies. Intensities are also computed. By default, the force constants are determined analytically if possible, by single numerical differentiation for methods for which only first derivatives are available, and by double numerical differentiation for those methods for which only energies are available.

Vibrational frequencies are computed by determining the second derivatives of the energy with respect to the Cartesian nuclear coordinates and then transforming to mass-weighted coordinates. *This transformation is only valid at a stationary point.* Thus, it is meaningless to compute frequencies at any geometry other than a stationary point for the method used for frequency determination.

For example, computing 6-311G(d) frequencies at a 6-31G(d) optimized geometry produces meaningless results. It is also incorrect to compute frequencies for a correlated method using frozen core at a structure optimized with all electrons correlated, or vice-versa. The recommended practice is to compute frequencies following a previous geometry optimization using the same method. This may be accomplished automatically by specifying both Opt and Freq within the route section for a job.

Note also that the CPHF (coupled perturbed SCF) method used in determining analytic frequencies is not physically meaningful if a lower energy wavefunction of the same spin multiplicity exists. Use the Stable keyword to test the stability of Hartree-Fock and DFT wavefunctions.

5.32.1 Calculation Variations

Additional related properties may also be computed during frequency calculations, including the following:

◊ When frequencies are done analytically, polarizabilities are also computed automatically; when numerical differentiation is required (or requested with Freq=Numer), polarizabilities must be explicitly requested using the Polar keyword (e.g., CCSD Freq Polar).

◊ The VCD option may be used to compute the vibrational circular dichroism (VCD) intensities in addition to the normal frequency analysis at the Hartree-Fock and DFT levels [395].

◊ The ROA option computes analytic Raman optical activity intensities [396–401]. However, see Polar=ROA for the recommended method and model chemistries for predicting ROA spectra.

◊ Pre-resonance Raman intensities may be computed by specifying one of the Raman options, and also including CPHF=RdFreq within the route and specifying the desired frequency in the input file (see the examples for additional information).

◊ Frequency-dependent polarizabilities and hyperpolarizabilities may be computed by including CPHF=RdFreq within the route (subject to their usual availability restrictions).

◊ Vibrational-rotational coupling can be computed using Freq=VibRot [402–410].

◊ The Anharmonic option performs numerical differentiation to compute anharmonic frequencies and zero-point energies [402–406, 408, 409, 411, 412] and anharmonic vibrational-rotational couplings [407, 410, 413–416] (as requested). This option is only available for methods with analytic second derivatives: Hartree-Fock, DFT, CIS and MP2. Full anharmonic IR intensities are computed [416, 417]. The DCPT2 [418, 419] and HDCPT2 [419] methods support resonance-free computations of anharmonic frequencies and partition functions. Anharmonic VCD and ROA spectra can also be predicted [420]. Calculations in solution are supported [421].

◊ There are several options for performing an analysis for an electronic excitation using the Franck-Condon [422–443], Herzberg-Teller method [422, 440–447] or combined Franck-Condon/Herzberg-Teller [440–443] methods (see the *Options* and additional input sections). They can be used to predict vibronic spectra and intensities, as well as resonance Raman spectra [448, 449]. Vibronic computations support chiral spectroscopies as well (ECD and CPL) [450, 451]. For a tutorial review, see [452].

The keyword Opt=CalcAll requests that analytic second derivatives be done at every point in a geometry optimization. Once the requested optimization has completed all the information necessary for a frequency analysis is available. Therefore, the frequency analysis is performed and the results of the calculation are archived as a frequency job.

5.32.2 Input

The SelectNormalModes and SelectAnharmonicModes options require additional input. The modes to select are specified in a separate blank-line terminated input section. The initial mode list is always empty.

Integers and integer ranges without a keyword are interpreted as mode numbers; although,This option is only available for methods with the [not]mode keywords may be used. The keywords atoms and notatoms can be used to define an atom list whose modes should be included/excluded (respectively). Atoms can also be specified by ONIOM layer via the [not]layer keywords, which accept these values: real for the real system, model for the model system in a 2-layer ONIOM, middle for the middle layer in a 3-layer ONIOM, and small for the model layer of a 3-layer ONIOM. Atoms may be similarly included/excluded by residue with residue and notresidue, which accept lists of residue names or numbers. Both keyword sets function as shorthand forms for atom lists.

Here are some examples:

`2-5`	*Includes modes 2 through 5.*
`atoms=O`	*Includes modes involving oxygen atoms.*
`1-20 atoms=Fe`	*Includes modes 1 through 20 and any modes involving iron atoms.*
`layer=real notatoms=H`	*Includes modes for heavy atoms in low layer (subject to default threshold).*

5.32.3 Options

Retrieving Force Constants

ReadFC

Requests that the force constants from a previous frequency calculation be read from the checkpoint file, and the mode and thermochemical analysis be repeated, presumably using a different temperature, pressure, or isotopes, at minimal computational cost. Note that since the basis set is read from the checkpoint file, no general basis should be input. If the Raman option was specified in the previous job, then do not specify it again when using this option.

Requesting Specific Spectra

Raman

Compute Raman intensities in addition to IR intensities. This is the default for Hartree-Fock. It may be specified for DFT and MP2 calculations. For MP2, Raman intensities are produced by numerical differentiation of dipole derivatives with respect to the electric field Raman is equivalent to NRaman for this method).

NRaman

Compute polarizability derivatives by numerically differentiating the analytic dipole derivatives with respect to an electric field. This is the default for MP2 if Freq=Raman.

NNRaman

Compute polarizability derivatives by numerically differentiating the analytic polarizability with respect to nuclear coordinates.

NoRaman

Skips the extra steps required to compute the Raman intensities during Hartree-Fock analytic frequency calculations, saving 10-30% in CPU time.

VCD

Compute the vibrational circular dichroism (VCD) intensities in addition to the normal frequency analysis. This option is valid for Hartree-Fock and DFT methods. This option also computes optical rotations (see Polar=OptRot).

ROA

Compute dynamic analytic Raman optical activity intensities using GIAOs [401]. This procedure requires one or more incident light frequencies to be supplied in the input to be used in the electromagnetic perturbations (CPHF=RdFreq is the default with Freq=ROA). This option is valid for Hartree-Fock and DFT methods. Note that the Polar=ROA keyword is often a better choice. NNROA says to use the numerical ROA method from Gaussian 03; this is useful only for reproducing the results of prior calculations.

Anharmonic Frequency Analysis

Anharmonic

Do numerical differentiation along modes to compute zero-point energies, anharmonic frequencies, and anharmonic vibrational-rotational couplings if VibRot is also specified. This option is only available for methods with analytic second derivatives: Hartree-Fock, DFT, CIS, and MP2.

ReadAnharm

Read an input section with additional parameters for the vibrational-rotational coupling and/or anharmonic vibrational analysis (VibRot or Anharmonic options). Available input options are documented below following the examples.

ReadHarmonic

Read the central point force constants and normal modes from a previous harmonic frequency calculation and avoid repeating the calculation at the central point.

ReadDifferentharmonic

Read the central point energy, forces, and force constants from a previous calculation and then compute 3^{rd} and 4^{th} derivatives at the current (presumably lower) level of theory for anharmonic spectra.

SelectAnharmonicModes

Read an input section selecting which modes are used for differentiation in anharmonic analysis. The format of this input section is discussed above. SelAnharmonicModes is a synonym for this option.

Vibronic Spectra: Franck-Condon, Herzberg-Teller and FCHT

The following options perform an analysis for an electronic excitation using the corresponding method; these jobs use vibrational analysis calculations for the ground state and the excited state to compute the amplitudes for electronic transitions between the two states. The vibrational information for the ground state is taken from the current job (Freq or Freq=ReadFC), and the vibrational information for the excited state is taken from a checkpoint file, whose name is provided in a separate input section (enclose the path in quotes if it contains internal spaces). The latter will be from a CI-Singles or TD-DFT Freq=SaveNormalModes calculation.

The ReadFCHT option can be added to cause additional input to be read to control these calculations (see below). In the latter case, the excited state checkpoint file would typically have been generated with Freq=(SelectNormalModes, SaveNormalModes) with the same modes selected.

FranckCondon

Use the Franck-Condon method [422–439, 441] (the implementation is described in [438–441]). FC is a synonym for this option. Transitions for ionizations can be analyzed instead of excitations. In this case, the molecule specification corresponds to the neutral form, and the additional checkpoint file named in the input section corresponds to the cation.

HerzbergTeller

Use the Herzberg-Teller method [422, 440, 444–447] (the implementation is described in [440]). HT is a synonym for this option.

FCHT

Use the Franck-Condon-Herzberg-Teller method [440].

Emission

Indicates that emission rather than absorption should be simulated for a Franck-Condon and/or Herzberg-Teller analysis. In this case, within the computation, the initial state is the excited state, and the final state is the ground state (although,This option allows you to specify alternatives to the default temperature, pressure, frequency scale factor the sources of frequency data for the ground and excited state are as described above: current job=ground state, second checkpoint file=excited state).

ReadFCHT

Read an input section containing parameters for the calculation. Available input options are documented below following the examples. This input section precedes that for ReadAnharmon if both are present.

Other Calculation Variations and Properties

VibRot

Analyze vibrational-rotational coupling.

Projected

For a point on a mass-weighted reaction path (IRC), compute the projected frequencies for vibrations perpendicular to the path. For the projection, the gradient is used to compute the tangent to the path. Note that this computation is very sensitive to the accuracy of the structure and the path [453]. Accordingly, the geometry should be specified to at least 5 significant digits. This computation is not meaningful at a minimum.

TProjected

Perform a projected harmonic frequency analysis if the RMS force is $\geq$ 1.d-3 Hartree/Bohr and perform regular harmonic analysis if the RMS force is smaller.

HinderedRotor

Requests the identification of internal rotation modes during the harmonic vibrational analysis [454–456]. If any modes are identified as internal rotation, hindered or free, the thermodynamic functions are corrected. The identification of the rotating groups is made possible by the use of redundant internal coordinates. Because some structures, such as transition states, may have a specific bonding pattern not automatically recognized, the set of redundant internal coordinates may need to be altered via the Geom=Modify keyword. Rotations involving metals require additional input via the ReadHinderedRotor option (see below).

If the force constants are available on a previously generated checkpoint file, additional vibrational/internal rotation analyses may be performed by specifying Freq=(ReadFC, HinderedRotor). Since Opt=CalcAll automatically performs a vibrational analysis on the optimized structure, Opt=(CalcAll, HinderedRotor) may also be used.

ReadHinderedRotor

Causes an additional input section to be read containing the rotational barrier cutoff height (in kcal/mol) and optionally the periodicity, symmetry number and multiplicity for rotational modes. Rotations with barrier heights larger than the cutoff value will be automatically frozen. If the periodicity value is negative, then the corresponding rotor is also frozen. You must provide the periodicity, symmetry and spin multiplicity for all rotatable bonds contain metals. The input section is terminated with a blank line, and has the following format:

VMax-value

Atom1 Atom2 periodicity symmetry spin *Repeated as necessary.*

...

Normal Modes

HPModes

Include the high precision format (to five figures) vibrational frequency eigenvectors in the frequency output in addition to the normal three-figure output.

InternalModes

Print modes as displacements in redundant internal coordinates. IntModes is a synonym for this option.

SaveNormalModes

Save all modes in the checkpoint file. SaveNM is a synonym for this option. NoSaveNormalModes, or NoSaveNM, is the default.

ReadNormalModes

Read saved modes from the checkpoint file. ReadNM is a synonym for this option. NoReadNormalModes, or NoReadNM, is the default.

SelectNormalModes

Read input selecting the particular modes to display. SelectNM is a synonym for this option. NoSelectNormalModes, or NoSelectNM, is the default. AllModes says to include all modes in the output. The format of this input section is discussed above. Note that this option does *not* affect the functioning of

SaveNormalModes, which always saves all modes in the checkpoint file.

SortModes

Sort modes by ONIOM layer in the output.

ModelModes

Display only modes involving the smallest model system in an ONIOM calculation.

MiddleModes

Display only modes involving the two model systems in a 3-layer ONIOM.

PrintDerivatives

Print normal mode derivatives of the dipole moment, polarizability, and so on.

PrintFrozenAtoms

By default, the zero displacements for frozen atoms are not printed in the mode output. This option requests that all atoms be listed.

NoPrintNM

Used to suppress printing of the normal mode components during a frequency calculation. The frequencies and intensities are still reported for each mode.

Geometry-Related Options

ModRedundant

Read-in modifications to redundant internal coordinates (i.e., for use with InternalModes). Note that the same coordinates are used for both optimization and mode analysis in an Opt Freq, for which this is the same as Opt=ModRedundant. See the discussion of the Opt keyword for details on the input format.

ReadIsotopes

This option allows you to specify alternatives to the default temperature, pressure, frequency scale factor and/or isotopes – 298.15 K, 1 atmosphere, no scaling, and the most abundant isotopes (respectively). It is useful when you want to rerun an analysis using different parameters from the data in a checkpoint file. Be aware, however, that all of these can be specified in the route section (Temperature, Pressure and Scale keywords) and molecule specification (the Iso parameter), as in this example:

```
#T Method/6-31G(d) JobType Temperature=300.0 ···

...

0 1
C(Iso=13)
...
```

ReadIsotopes input has the following format:

temp pressure [scale]	*Values must be real numbers.*
isotope mass for atom 1	
isotope mass for atom 2	
...	
isotope mass for atom n	

Where *temp*, *pressure*, and *scale* are the desired temperature, pressure, and an optional scale factor for frequency data when used for thermochemical analysis (the default is unscaled). The remaining lines hold the isotope masses for the various atoms in the molecule, arranged in the same order as they appeared in the molecule specification section. If integers are used to specify the atomic masses, the program will automatically use the corresponding actual exact isotopic mass (e.g., 18 specifies ^{18}O, and Gaussian uses the value 17.99916).

Algorithm Variations and Execution Options

Analytic

This specifies that the second derivatives of the energy are to be computed analytically. This option is available only for RHF, UHF, CIS, CASSCF, MP2, and all DFT methods, and it is the default for those cases.

Numerical

This requests that the second derivatives of the energy are to be computed numerically using analytically calculated first derivatives. It can be used with any method for which gradients are available and is the default for those for which gradients but not second derivatives are available. Freq=Numer can be combined with Polar=Numer in one job step.

FourPoint

Do four displacements instead of two for each degree of freedom during numerical frequencies, polarizabilities, or Freq=Anharm. This gives better accuracy and less sensitivity to step size at the cost of doing twice as many calculations.

DoubleNumer

This requests double numerical differentiation of energies to produce force constants. It is the default and only choice for those methods for which no analytic derivatives are available. EnOnly is a synonym for DoubleNumer.

Cubic

Requests numerical differentiation of analytic second derivatives to produce third derivatives. Applicable only to methods having analytic frequencies but no analytic third derivatives.

Step=*N*

Specifies the step-size for numerical differentiation to be 0.0001**N* (in Angstoms unless Units=Bohr has been specified). If Freq=Numer and Polar=Numer are combined, *N* also specifies the step-size in the electric field. The default is 0.001 Å for Hartree-Fock and correlated Freq=Numer, 0.005 Å for GVB and CASSCF Freq=Numer, and 0.01 Å for Freq=EnOnly. For Freq=Anharmonic or Freq=VibRot, the default is 0.025 Å.

Restart

This option restarts a frequency calculation after the last completed geometry. A failed frequency job may be restarted from its checkpoint file by simply repeating the route section of the original job, adding the Restart option to the Freq=Numer keyword/option. No other input is required.

Analytic frequencies can be restarted with the Restart keyword provided that the read-write file was named and saved from the failed job. See the description of that keyword for more information and an example.

DiagFull

Diagonalize the full $(3N_{atoms})^2$ force constant matrix – including the translation and rotational degrees of freedom – and report the lowest frequencies to test the numerical stability of the frequency calculation. This precedes the normal frequency analysis where these modes are projected out. Its output reports the lowest 9 modes, the upper 3 of which correspond to the 3 smallest modes in the regular frequency analysis. Under ideal conditions, the lowest 6 modes reported by this analysis will be very small in magnitude. When they are significantly non-zero, it indicates that the calculation is not perfectly converged/numerically stable. This may indicate that translations and rotations are important modes for this system, that a better integration grid is needed, that the geometry is not converged, etc. The system should be studied further in order to obtain accurate frequencies. See the examples section below for the output from this option. DiagFull is the default; NoDiagFull says to skip this analysis.

TwoPoint

When computing numerical derivatives, make two displacements in each coordinate. This is the default. FourPoint will make four displacements but only works with Link 106 (Freq=Numer). Not valid with Freq=DoubleNumer.

NFreq=*N*

Requests that the lowest N frequencies be solved for using Davidson diagonalization. At present, this option is only available for ONIOM(QM:MM) model chemistries.

WorkerPerturbations

During numerical frequencies using Linda parallelism, run separate displacements on each worker instead of parallelizing each energy+derivative evaluation across the cluster. This strategy is more efficient, but it requires specifying an extra worker on the master node. It is the default if at least 3 Linda workers were specified. NoWorkerPerturbations suppresses this behavior.

5.32.4 Availability

Analytic frequencies are available for the AM1, PM3, PM3MM, PM6, PDDG, DFTB, DFTBA, HF, DFT, MP2, CIS, TD and CASSCF methods.

Numerical frequencies are available for MP3, MP4(SDQ), CID, CISD, CCD, CCSD, EOM-CCSD and QCISD.

Raman is available for the HF, DFT and MP2 methods.

VCD and ROA are available for HF and DFT methods.

Anharmonic is available for HF, DFT, MP2 and CIS methods.

Freq and NMR can both be on the same route for HF and DFT.

5.32.5 Related Keywords

Polar, Opt, Stable, NMR.

5.32.6 Examples

Frequency Output. The basic components of the output from a frequency calculation are discussed in detail in chapter 4 of *Exploring Chemistry with Electronic Structure Methods* [152].

New Gaussian users are often surprised to see that the final part frequency calculation output that looks that of a geometry optimization at the beginning of a frequency job:

```
GradGradGradGradGradGradGradGradGradGradGradGradGradGradGrad
Berny optimization.
Initialization pass.
```

Link 103, which performs geometry optimizations, is executed at the beginning and end of all frequency calculations. This is done so that the quadratic optimization step can be computed using the correct second derivatives. Occasionally an optimization will complete according to the normal criterion using the approximate Hessian matrix, but the step size is actually larger than the convergence criterion when the correct second derivatives are used. The next step is printed at the end of a frequency calculation so that such problems can be identified. If you think this concern is applicable, use Opt=CalcAll instead of Freq in the route section of the job, which will complete the optimization if the geometry is determined not to have fully converged (usually, given the full second derivative matrix near a stationary point, only one additional optimization step is needed), and will automatically perform a frequency analysis at the final structure.

Specifying #P in the route section produces some additional output for frequency calculations. Of most importance are the polarizability and hyperpolarizability tensors (the latter in Raman calculations only); although, they still may be found in the archive entry in normal print-level jobs. They are presented in lower triangular and lower tetrahedral order, respectively (i.e., α_{xx}, α_{xy}, α_{yy}, α_{xz}, α_{yz}, α_{zz} and β_{xxx}, β_{xxy}, β_{xyy}, β_{yyy}, β_{xxz}, β_{xyz}, β_{yyz}, β_{xzz}, β_{yzz}, β_{zzz}), in the standard orientation:

```
Dipole         = 2.37312183D-16 -6.66133815D-16 -9.39281319D-01
Polarizability= 7.83427191D-01  1.60008472D-15  6.80285860D+00
               -3.11369582D-17  2.72397709D-16  3.62729494D+00
HyperPolar     = 3.08796953D-16 -6.27350412D-14  4.17080415D-16
                 5.55019858D-14 -7.26773439D-01 -1.09052038D-14
               -2.07727337D+01  4.49920497D-16 -1.40402516D-13
               -1.10991697D+01
```

#P also produces a bar-graph of the simulated spectra for small cases.

Thermochemistry analysis follows the frequency and normal mode data:

```
Zero-point correction=                   .023261 (Hartree/Particle)
Thermal correction to Energy=            .026094
Thermal correction to Enthalpy=          .027038
Thermal correction to Gibbs Free Energy= .052698
Sum of electronic and zero-point Energies=    -527.492585   E0=Eelec+ZPE
Sum of electronic and thermal Energies=       -527.489751   E= E0+ Evib+ Erot+Etrans
Sum of electronic and thermal Enthalpies=     -527.488807   H=E+RT
Sum of electronic and thermal Free Energies= -527.463147   G=H-TS
```

The raw zero-point energy correction and the thermal corrections to the total energy, enthalpy, and Gibbs free energy (all of which include the zero-point energy) are listed, followed by the corresponding corrected energy. The analysis uses the standard expressions for an ideal gas in the canonical ensemble. Details can be found in McQuarrie [457] and other standard statistical mechanics texts. In the output, the various quantities are labeled as follows:

```
 E (Thermal)         Contributions to the thermal energy correction
```

CV	*Constant volume molar heat capacity*
S	*Entropy*
Q	*Partition function*

The thermochemistry analysis treats all modes other than the free rotations and translations as harmonic vibrations. For molecules having hindered internal rotations, this can produce slight errors in the energy and heat capacity at room temperatures and can have a significant effect on the entropy. The contributions of any very low frequency vibrational modes are listed separately so that their harmonic contributions can be subtracted from the totals and their correctly computed contributions included should they be group rotations and high accuracy is required. Expressions for hindered rotational contributions to these terms can be found in Benson [458]. The partition functions are also computed, with both the bottom of the vibrational well and the lowest (zero-point) vibrational state as reference.

Pre-resonance Raman. This calculation type is requested with one of the Raman options in combination with CPHF=RdFreq. The frequency specified for the latter should be chosen as follows:

◊ Determine the difference in frequency between the peak of interest in the UV/visible absorption spectrum and the incident light used in the Raman experiment.

◊ Perform a TD calculation using a DFT method in order to determine the predicted location of the same peak.

◊ Specify a frequency for CPHF=RdFreq which is shifted from the predicted peak by the same amount as the incident light differs from the observed peak.

Pre-resonance Raman results are reported as additional rows within the normal frequency tables:

```
Harmonic frequencies (cm**-1), IR intensities (KM/Mole), Raman
scattering activities (A**4/AMU), depolarization ratios for plane
and unpolarized incident light, reduced masses (AMU), force constants
(mDyne/A), and normal coordinates:
                    1
                   B1
Frequencies --  1315.8011
Red. masses --     1.3435
Frc consts  --     1.3704
IR Inten    --     7.6649
Raman Activ --     0.0260
Depolar (P) --     0.7500
Depolar (U) --     0.8571
RamAct Fr= 1--     0.0260   Additional output lines begin here.
 Dep-P Fr= 1--     0.7500
 Dep-U Fr= 1--     0.8571
RamAct Fr= 2--     0.0023
 Dep-P Fr= 2--     0.7500
 Dep-U Fr= 2--     0.8571
```

Vibration-Rotation Coupling Output. If the VibRot option is specified, then the harmonic vibrational-rotational analysis appears immediately after the normal thermochemistry analysis in the output, introduced by this header:

```
Harmonic Vibro-Rotational Analysis
```

If anharmonic analysis is requested as well (i.e., VibRot and Anharmonic are both specified), then the anharmonic vibrational-rotational analysis results follow the harmonic ones, introduced by the following header:

```
Second-order Perturbative Anharmonic Analysis
```

Anharmonic Frequency Calculations. Freq=Anharmonic jobs produce additional output following the normal frequency output. (It follows the vibrational-rotational coupling output if this was specified as well.) We will briefly consider the most important items.

The output displays the equilibrium geometry (i.e., the minimum on the potential energy surface), followed by the anharmonic vibrationally averaged structure at 0 K:

```
Internal coordinates for the Equilibrium structure (Se)

                         Interatomic distances:
                 1          2          3          4
    1  C    0.000000
    2  O    1.206908   0.000000
    3  H    1.083243   2.008999   0.000000
    4  H    1.083243   2.008999   1.826598   0.000000
                         Interatomic angles:
     O2-C1-H3=122.5294      O2-C1-H4=122.5294      H3-C1-H4=114.9412
     O2-H3-H4= 62.9605
                            Dihedral angles:
     H4-C1-H3-O2= 180.

Internal coordinates for the vibrationally average structure at 0K (Sz)

                         Interatomic distances:
                 1          2          3          4
    1  C    0.000000
    2  O    1.210431   0.000000
    3  H    1.097064   2.024452   0.000000
    4  H    1.097064   2.024452   1.849067   0.000000
                         Interatomic angles:
     O2-C1-H3=122.57        O2-C1-H4=122.57        H3-C1-H4=114.8601
     O2-H4-H3= 62.8267
                            Dihedral angles:
     H4-C1-H3-O2= 180.
```

Note that the bond lengths are slightly longer in the latter structure. The predicted coordinates at STP follow in the output.

The anharmonic zero point energy is given shortly thereafter in the output:

```
Anharmonic Zero Point Energy
----------------------------
Harmonic       : cm-1 =  5008.40626 ; Kcal/mol =  14.320 ; KJ/mol =  59.914
Anharmonic Pot.: cm-1 =   -53.31902 ; Kcal/mol =  -0.152 ; KJ/mol =  -0.638
```

```
Watson+Coriolis: cm-1 =   -12.83227 ; Kcal/mol =  -0.037 ; KJ/mol =  -0.154
Total Anharm    : cm-1 =  4942.25496 ; Kcal/mol =  14.131 ; KJ/mol =  59.122
```

The anharmonic frequencies themselves appear just a bit later in this table, in the column labeled `E(anharm)`:

```
    ==================================================
              Anharmonic Infrared Spectroscopy
    ==================================================

Units: Transition energies (E) in cm^-1
       Integrated intensity (I) in km.mol^-1

Fundamental Bands
-----------------
  Mode(n)                  E(harm)   E(anharm)        I(harm)        I(anharm)
     1(1)                  2938.531   2788.983     55.17567187     55.41312200
     2(1)                  1888.862   1864.231    101.42877427    104.63741421
     ...

Overtones
---------
  Mode(n)                  E(harm)   E(anharm)                       I(anharm)
     1(2)                  5877.061   5517.149                      0.00211652
     2(2)                  3777.724   3710.383                      3.68324904
     ...

Combination Bands
-----------------
  Mode(n)      Mode(n)     E(harm)   E(anharm)                       I(anharm)
     2(1)         1(1)     4827.393   4654.114                      1.74785224
     3(1)         1(1)     4490.139   4271.343                      0.04557003
     ...
```

The harmonic frequencies are also listed for convenience.

Vibronic Analysis. The following input file predicts the vibronic spectrum:

```
%OldChk=excited                          Excited state calculation.
%Chk=fcht
# Freq=(ReadFC,FCHT,ReadFCHT) Geom=AllCheck ···

TimeIndependent                          ReadFCHT additional input.
Output=Matrix=JK                         Output Duschinsky matrix and shift vector.
     final blank line
```

The molecule specification is taken from the checkpoint file from the excited state, as are the force constants for the excited states.

FCHT analysis produces many results. The final Duschinsky (state overlap) matrix appears as follows:

```
Final Duschinsky matrix
```

```
-----------------------
Note: The normal coordinates of the final state (columns) are expressed
      in the basis set of the normal coordinates of the initial state (rows)
           1             2             3             4             5
1  -0.539484D+00  0.839747D+00  0.139916D-01 -0.147815D-01  0.167387D-02
2  -0.594185D+00 -0.373849D+00 -0.647845D+00  0.757424D-01 -0.627709D-02
3   0.303582D-01  0.276954D-01  0.572527D-02  0.354162D+00 -0.933518D+00
...
```

Note that this output reports the value of J_{ij} for each pair of states. Generally, what is plotted is J^2. The locations and intensities of the predicted bands are reported as follows:

```
    ==================================================
                Information on Transitions
    ==================================================

 Energy of the 0-0 transition:  31327.1976 cm^(-1)

 NOTE: The energy (transition energy) refers to the relative energy,
       with respect to the 0-0 transition energy.
       The intensity is the line intensity.
       DipStr is the dipole strength.

 Energy =      0.0000 cm^-1: |0> -> |0>
   -> Intensity =  7003.     (DipStr = 0.9135E-01)

 Energy =    457.9310 cm^-1: |0> -> |9^1>
   -> Intensity =  650.2     (DipStr = 0.8360E-02)
 ...
```

Frequency and transition (states).

Location is 31875 cm^{-1}.
Intensity in dm^{3}cm^{-1}mol^{-1};
Dipole strength in au.

The final predicted spectrum follows in a form suitable for plotting:

```
    ==================================================
                      Final Spectrum
    ==================================================

 Band broadening simulated by mean of Gaussian functions with
 Half-Widths at Half-Maximum of  135.00 cm^(-1)

 Legend:
 -------
 1st col.: Energy (in cm^-1)
 2nd col.: Intensity at T=0K
 Intensity: Molar absorption coefficient (in dm^3.mol^-1.cm^-1)
 -----------------------------
    30327.1976    0.000000D+00
    ...
    31319.1976    0.699549D+04
    31327.1976    0.701428D+04
    31335.1976     0.699927D+04
    ...
```

Resonance Raman Spectra. The following input file computes the resonance Raman intensities from two previously run frequency calculations.

```
%Chk=S0_freq                                             Ground state checkpoint file.
# Freq=(FC,ReadFC,ReadFCHT) Geom=AllCheck ···

TimeIndependent                                          ReadFCHT additional input.
Spectroscopy=ResonanceRaman                              Predict resonance Raman spectrum.
Spectrum=(Lower=800.,Upper=2800.,Broadening=Stick)       Spectrum specifications.
Intermediate=Source=Chk                                  Get second state data from checkpoint file (named below).
RR=(OmegaMin=55000,OmegaMax=56000,OmegaStep=100)         RR analysis parameters: ϱ range and step size.

S2_freq.chk                                              Excited state checkpoint file.
```

See the section on Freq=ReadFCHT for details about the additional input. For each of the Raman modes, the following output appears for each point in the specified range of incident energies (omega):

```
    ==================================================
                Information on Transitions
    ==================================================

 Energy of the 0-0 transition:  54854.2397 cm^(-1)

 Alp2: alpha^2, BsAl: beta_s(alpha)^2, BaAl: beta_a(alpha)^2

 Energy =      0.0000 cm^-1: |0> -> |0>          Relative energy and involved states.
   -> Omega =  55000.0 cm^-1, Sigma =   1.1332
      Alp2 =  0.33009E+02, BsAl =  0.29859E+03, BaAl =  0.00000E+00
```

Following this output, the same data is presented in a tabular form:

```
    ==================================================
                      Final Spectrum
    ==================================================

 No band broadening applied (stick spectrum)

 Legend:
 -------
 1st col.: Raman shift (in cm^-1)
 2nd col.: Intensity at T=0K for incident energy:  55000.00 cm^-1
 3rd col.: Intensity at T=0K for incident energy:  55100.00 cm^-1
 4th col.: Intensity at T=0K for incident energy:  55200.00 cm^-1
 5th col.: Intensity at T=0K for incident energy:  55300.00 cm^-1
 Raman scattering intensity in cm^3.mol^-1.sr^-1
 ------------------------------------------------------------------------------
 ...
 1188.0000    0.000000D+00    0.000000D+00    0.000000D+00    0.000000D+00
 1190.0000    0.134622D-21    0.213038D-21    0.358179D-21    0.644832D-21
 1192.0000    0.000000D+00    0.000000D+00    0.000000D+00    0.000000D+00
  ...
```

Since no spectral broadening was requested here (Spectrum=Broadening=Stick), the only rows with non-zero intensities correspond to the Raman active frequencies.

Examining Low-Lying Frequencies. The output from the full force constant matrix diagonalization (the default Freq=DiagFull), in which the rotational and translational degrees of freedom are retained, appears as following in the output:

```
 Low frequencies ---  -19.9673   -0.0011   -0.0010    0.0010   14.2959
 Low frequencies ---   25.6133  385.4672  988.9028 1083.0692
```

This output is from an Opt Freq calculation on methanol. Ignoring sign, there are 3 low-lying modes, located at around 14, 19, and 25 wavenumbers (in addition to the three that are ∼0). However, if we rerun the calculation using tight optimization criteria (Opt=Tight) and a larger integration grid, the lowest modes become:

```
 Low frequencies ---   -7.4956   -5.4813   -2.6908    0.0003    0.0007
 Low frequencies ---    0.0011  380.1699  988.1436 1081.9083
```

The low-lying modes are now quite small, and the lowest frequencies have moved slightly as a result.

This analysis is especially important for molecular systems having frequencies at small wavenumbers. For example, if the lowest reported frequency is around 30 and there is a low-lying mode around 25 as above, then the former value is in considerable doubt (as is whether the molecular structure is even a minimum).

Rerunning a Frequency Calculation with Different Thermochemistry Parameters. The following two-step job contains an initial frequency calculation followed by a second thermochemistry analysis using a different temperature, pressure, and selection of isotopes:

```
%Chk=freq
# B3LYP/6-311+G(2d,p) Freq

Frequencies at STP

molecule specification

-Link1-
%Chk=freq
%NoSave
# B3LYP/6-311+G(2d,p) Freq(ReadIso,ReadFC) Geom=Check

Repeat at 300 K

0,1

300.0 1.0
```

```
16
 2
 3
...
```

Note also that the freqchk utility may be used to rerun the thermochemical analysis from the frequency data stored in a Gaussian checkpoint file.

5.32.7 ReadAnharm Input

The following keywords for specifying various aspects of Freq=Anharm calculations are included as additional input within the Gaussian input file. They control various aspects of anharmonic frequency analyses. Note that these keywords are completely different from those supported in Gaussian 09 (a few of these changes were introduced in Gaussian 09 revision D.01).

Data Sources and Format

The *DataSrc*, *DataAdd* and *DataMod* input items locate the various data required by the anharmonic frequency analysis. They each take a list of parameters and associated values which specify locations from which to retrieve different data items. In general, parameters specify what data is to be read and their values specify the location of that data. The available options for the latter are listed below. Generally, they may be optionally followed by a format suffix.

Source keywords are used to specify where the data is located:

◊ Src: Use data from the RWF file for the current job. RWF is a synonym.

◊ Chk: Retrieve data from the current checkpoint file (as defined with %Chk or %OldChk).

◊ In: Read data from the input stream.

◊ InChk*n*: Retrieve data from the *n*th file in the file list (see the discussion of additional input sections below). Valid values of n run from 1 to 6.

Format Suffixes are appended directly to the source item, and they specify a non-default format for various read-in data. For example, InQMW says to read derivative data from the input stream in mass-weighted normal coordinates. The following suffixes are available:

◊ QMW: Derivatives are with respect to normal modes in mass-weighted normal coordinates. Q is a synonym for this format suffix.

◊ QMWX: Harmonic derivatives are with respect to normal modes in Cartesian coordinates, and anharmonic derivatives are with respect to normal modes in mass-weighted normal coordinates. X is a synonym for this format suffix.

◊ QRedX: Harmonic derivatives are with respect to normal modes in Cartesian coordinates, and anharmonic derivatives are with respect to normal modes in dimensionless normal coordinates.

By default, QMWX is tried first, followed by QMW.

DataSrc=*param*: Specify the source(s) of various read-in data. The parameter consists of a keyword indicating the data to which it applies and a source keyword indicating its location (and possibly its format). The available parameters are:

◊ *source*: Sets the data source for all data.

◊ Harm=*source*: Sets the data source for harmonic data. The default is taken from the source file.

◊ Anharm=*source*: Sets the data source for harmonic data. The default is taken from the source file.

◊ Coriolis=*source*: Sets the data source for the Coriolis couplings. At present, the only supported items are Src and In, and format suffixes may not be used.

◊ NMOrder=*ordering*: Specifies the order normal modes are stored in the input source, selected from the following list. This item may be specified in addition to a source item.

- AscNoIrrep: Ascending order. Do not sort by irreducible representation. This is the default.
- Asc: Ascending order. Sort by irreducible representation if possible.
- Desc: Descending order. Sort by irreducible representation if possible.
- DescNoIrrep: Descending order. Do not sort by irreducible representation.
- Print: Use same order as for printing.

The following DataSrc items are deprecated, and are included only for backward similarity to Gaussian 09 (where they functioned as top-level additional input items).

◊ InDerAU: Use data from the input stream in atomic units.

◊ InDerAJ: Use data from the input stream in attoJoules

◊ InDerRed: Use data from the input stream in reduced form. Reduced is an alternate name for this item.

◊ InDerGau: Use data from the input stream with the layout of the Gaussian output. InGauDer is an alternate name for this item.

DataAdd=*params*: Read alternate data to replace or complete the original data. Using this option will replace the already existing information in the original data with the data specified here.

◊ Freq: Replace harmonic frequencies with values given in the input stream (in cm^{-1}). A data source may also be specified as a parameter: Freq=*source*, but format suffixes are not allowed.

◊ PESFull=*source*: Read force constraints from specified specified source.

◊ PESHarm=*sources*: Read harmonic force constants from specified specified source.

◊ PESAnh=*sources*: Read anharmonic force constants from specified specified source.

◊ EDipFull=*sources*: Read the electric dipole from the specified specified source.

◊ EDipHarm=*sources*: Read the harmonic components of the electric dipole from the specified specified source.

◊ EDipAnh=*sources*: Read the anharmonic components of the electric dipole from the specified specified source.

◊ MDipFull=*sources*: Read the magnetic dipole from the specified source.

◊ MDipHarm=*sources*: Read the harmonic components of the magnetic dipole from the specified source.

◊ MDipAnh=*sources*: Read the anharmonic components of the magnetic dipole from the specified source.

◊ PolFull=*sources*: Read the polarizability tensor from the specified source.

◊ PolHarm=*sources*: Read the harmonic components of the polarizability tensor from the specified source.

◊ PolAnh=*sources*: Read the anharmonic components of the polarizability tensor from the specified source.

◊ MagFFull=*sources*: Read the magnetic-field properties from the specified source.

◊ MagFHarm=*sources*: Read the harmonic components of the magnetic-field properties from the specified source.

◊ MagFAnh=*sources*: Read the anharmonic components of the magnetic-field properties from the specified source.

◊ FreqDepPFull=*sources*: Read the frequency-dependent properties from the specified source.

◊ FreqDepPHarm=*sources*: Read the harmonic components of the frequency-dependent properties from the specified source.

◊ FreqDepPAnh=*sources*: Read the anharmonic components of the frequency-dependent properties from the specified source.

DataMod=*params*: Modify the data in various manners.

◊ ScHarm=*value*: Scales harmonic frequencies with a constant scaling factor (default is 1.0).

◊ NoCor: Discards Coriolis couplings in calculations. By default, all couplings will be retained.

◊ DerOrder=*N*: Selects the derivatives order to keep. E.g. DerOrder=123 discards all quartic force constants.

◊ DerIndex=*N*: Sets the maximum number of independent indexes for a derivative. E.g. DerIndex=2 keeps k_{iij} but discards k_{ijk}.

◊ SkipPT2=*what*: Selectively removes derivatives based on the parameter, whose possible values are listed below:

- No: Do not remove the data. This is the default option.
- Modes: Removes the derivatives with respect to any of the normal modes given in the input stream.
- Constants: Removes the derivatives based on the indexes given in the input stream. The input explicitly specifies the force constants (energy derivatives) to be removed. Each line specifies the involved normal modes, with the derivative order implied by the number of indexes. For example, to remove the third derivatives with respect to normal coordinates $Q_1Q_2Q_5$, the input line would be:
  ```
  1 2 5
  ```
- OptModes: Modify derivatives based on additional instructions in the input stream (see input ordering section below).

Tolerances=*data*: Modify the tolerance threshold to include/discard derivative data.

◊ Gradient=*value*: Threshold for the energy first derivatives (default is 3.7074×10^{-3}).

◊ Hessian=*value*: Threshold for the energy second derivatives (default is 3.7074×10^{-5}).

◊ Cubic=*value*: Threshold for the energy third derivatives (default is 3.7074×10^{-5}).

◊ Quartic=*value*: Threshold for the energy fourth derivatives (default is 3.7074×10^{-5}).

◊ Coriolis=*value*: Threshold for the Coriolis couplings (default is 1.0×10^{-3}).

◊ Inertia=*value*: Threshold for the principal moments of inertia (default is 1.0×10^{-4}Å^2).

◊ Symm=*value*: Tolerance for anharmonic data with respect to symmetry rules (default is 2%).

Output Control

This section specifies the contents and destination of the calculation output.

Print=*items*: Include items in the output file. Available items are the following:

◊ InDataX: Include data compatible with DataSrc=InQMWX. The form Print=InDataX=Ext writes the data to the external file **input_data.dat**.

◊ InDataNM: Include data compatible with DataSrc=InQMW. The form Print=InDataNM=Ext writes the data to the external file **input_data.dat**.

◊ YMatrix: Include the Y matrix (a variant of the χ matrix).

◊ Verbosity=*n*: Specify the verbosity level. The default is 0.

◊ ITop=*rep*: Selects the representation used for rotational spectroscopy. By default, it is defined automatically by Gaussian from the principal moments of inertia. Available representations are:

- Ir: Ir Representation: $I_z < I_x < I_y$
- IIr: IIr Representation: $I_y < I_z < I_x$
- IIIr: IIIr Representation: $I_x < I_y < I_z$
- Il: Il Representation: $I_z < I_y < I_x$
- IIl: IIl Representation: $I_x < I_z < I_y$
- IIIl: IIIl Representation: $I_y < I_x < I_z$

◊ ZAxisSymm=*axis*: Sets the Eckart axis to be used as Z for the definition of the reduced Hamiltonians for the vibrorotational analysis. Available choices are:

- X: Z collinear with X.
- Y: Z collinear with Y.
- Z: Z collinear with Z.

◊ NMOrder=*ordering*: Specifies the order in which normal modes are listed:

- Asc: Ascending order. Sort by irreducible representation if possible.
- Desc: Descending order. Sort by irreducible representation if possible. This is the default.
- AscNoIrrep: Ascending order. Do not sort by irreducible representation.
- DescNoIrrep: Descending order. Do not sort by irreducible representation.

◊ PT2VarEVec: Include the eigenvector matrix from the diagonalization of the variational matrix.

◊ PT2VarStates: Include the projection of the variational states on the deperturbed ones.

◊ PT2VarProj: Include the projection of the DVPT2 states on the new variational states.

◊ InDataAU: Write data compatible with DataSrc=InDerAU (deprecated).

◊ Polymode: Write data to use in the Polymode program.

Reduced-Dimensionality Schemes

RedDim=*items*: Specifies which normal modes are active in the analysis. Items are:

◊ Active=*n*: Activate the *n* modes specified in the input stream. By default, all modes are active.

◊ Inactive=*n*: Read list of *n* inactive modes from the input stream.

◊ Frozen=*n*: Read *n* modes to be frozen from the input stream.

◊ MinFreqAc=*freq*: Sets the normal modes with a frequency above the specified value to be active (default is 0). Only valid if MaxFreqAc>MinFreqAc.

◊ MaxFreqAc=*freq*: Sets the normal modes with a frequency below the specified value to be active (default is infinity). Only valid if MaxFreqAc>MinFreqAc.

◊ MinFreqIn=*freq*: Sets the normal modes with a frequency above the specified value to be inactive (default is 0). Only valid if MaxFreqIn>MinFreqIn.

◊ MaxFreqIn=*freq*: Sets the normal modes with a frequency below the specified value to be active (default is infinity). Only valid if MaxFreqIn>MinFreqIn.

◊ MinFreqFr=*freq*: Sets the normal modes with a frequency above the specified value to be frozen (default is 0). Only valid if MaxFreqFr>MinFreqFr.

◊ MaxFreqFr=*freq*: Sets the normal modes with a frequency below the specified value to be frozen (default is infinity).Only valid if MaxFreqFr>MinFreqFr.

Second-Order Vibrational Perturbation Theory (VPT2)

PT2Model=*data*: Sets the VPT2 model to use. The default is GVPT2.

◊ HDCPT2: Use the Hybrid Degeneracy-Corrected 2nd-order Perturbation Theory.

◊ VPT2: Use the original 2nd-order Vibrational Perturbation Theory. Vibrational spectroscopy intensities are available for this model.

◊ DVPT2: Use the Deperturbed 2nd-order Vibrational Perturbation Theory. Vibrational spectroscopy intensities are available for this model. The form DVPT2=all selects all possibly resonant terms as Fermi resonances, and it is equivalent to Resonances=(DFreqFrm=∞,DPT2Var=0).

◊ GVPT2: Use the Generalized 2nd-order Vibrational Perturbation Theory. This is the default. It is similar to DVPT2, but the removed terms are treated variationally in a second step. Vibrational spectroscopy intensities are available for this model. The form GVPT2=all selects all possibly resonant terms as Fermi resonances, and it is equivalent to Resonances=(DFreq12=∞,K12Min=0).

◊ DCPT2: Use the Degeneracy-Corrected 2nd-order Perturbation Theory.

HDCPT2=*params*: Set the parameters for the model with the Alpha and Beta options (i.e., HDCPT2=Alpha=*value*), which specify values for the corresponding variables in the expression for Λ: $\Lambda = \left[\tanh\alpha \times \left(\sqrt{k^2\varepsilon^2} - \beta\right) + 1\right]/2$

Resonances=*params*: Set resonance thresholds and parameters for DVPT2 (Fermi-related items only) and GVPT2 calculations.

◊ DFreq12=*value*: Sets the maximum frequency for 1 – 2 Fermi resonances ($\omega_i - (\omega_j + \omega_k)$). The default is 200 cm^{-1}.

◊ DFreq22=*value*: Sets the maximum frequency for 2 – 2 Darling-Dennison resonances ($2\omega_i - (\omega_j + \omega_k)$ and $2\omega_i - 2\omega_j$). The default is 100 cm^{-1}.

◊ DFreq11=*value*: Sets the maximum frequency for 1 – 1 Darling-Dennison resonances ($\omega_i\omega_j$). The default is 100 cm^{-1}.

◊ DFreq13=*value*: Sets the maximum frequency for 1 – 3 Darling-Dennison resonances ($\omega_i - (\omega_j + \omega_k + \omega_l)$). The default is 100 cm^{-1}.

◊ K12Min=*value*: Sets the maximum allowed difference between the VPT2 and model variational results (Martin test). The default is 1 cm^{-1}.

◊ K22Min=*value*: Sets the minimum value for off-diagonal 2 – 2 Darling-Dennison term. The default is 10 cm^{-1}.

◊ K11Min=*value*: Sets the minimum value for off-diagonal 1 – 1 Darling-Dennison term. The default is 1 cm^{-1}.

◊ K11MinI=*value*: Sets the minimum value for the secondary 1 – 1 resonance test, intended to detect critical cases specific to intensity calculations. The default is 1 cm^{-1}.

◊ K13Min=*value*: Sets the minimum value for off-diagonal 1 – 3 Darling-Dennison term. The default is 10 cm^{-1}.

◊ K13MinI=*value*: Sets the minimum value for the secondary 1 – 3 resonance test, intended to detect critical cases specific to intensity calculations. The default is 0.25 cm^{-1}.

◊ HDCPT2=*value*: Sets the minimum value for the HDCPT2/VPT2 difference test. The default is 0.1.

◊ NoFermi: Deactivates the search for 1 – 2 Fermi resonances. No12Res is a synonym for this item.

◊ NoDarDen: Deactivates the search for Darling-Dennison (2 – 2, 1 – 1 and 1 – 3) resonances.

◊ No22Res: Deactivates the search for 2 – 2 Darling-Dennison resonances.

◊ No11Res: Deactivates the search for 1 – 1 Darling-Dennison resonances.

◊ No13Res: Deactivates the search for 1 – 3 Darling-Dennison resonances. 1 – 3 resonances are only available for 3-quanta transitions.

◊ List=*action*: Tells Gaussian to read resonance cases from the input stream. *action* is optional; it defaults to Replace. Otherwise, it controls the use of the input resonances. The available *action* keywords are:

- Replace: Discards automatic analysis and only use resonances in the input. This is the default.
- Add: Augments automatic results with input data. A simpler form of this option is Resonances=Add.
- Delete: Remove resonances in the input list from the results of the automatic analysis. A simpler form of this option is Resonances=Delete.
- Modify: Add or remove resonances starting from the list obtained from the automatic analysis. An action keyword – ADD or DEL – must precede the resonance data on each input line.

See the *Specifying Resonances* subsection below for more information.

Spectroscopy

Spectro=MaxQuanta=*quanta*: Compute transition integrals to states with up to the specified quanta. The default is 2.

ROA=*params*: Options related to Raman optical activity. If the keyword is specified, then only those scatterings explicitly requested will be computed. If the ROA is not specified, then intensities are computed for all supported scatterings.

◊ ICP0: Compute ROA intensity for ICP forward scattering.

◊ ICP90x: Compute ROA intensity for incident circular polarization (ICP) right-angle scattering (polarized).

◊ ICP90z: Compute ROA intensity for ICP right-angle scattering (depolarized).

◊ ICP90*: Compute ROA intensity for ICP right-angle scattering (magic angle).

◊ ICP180: Compute ROA intensity for ICP backward scattering.

◊ SCP0: Compute ROA intensity for scattered circular polarization(SCP) forward scattering.

◊ SCP90x: Compute ROA intensity for SCP right-angle scattering (polarized).

◊ SCP90z: Compute ROA intensity for SCP right-angle scattering (depolarized).

◊ SCP90*: Compute ROA intensity for SCP right-angle scattering (magic angle).

◊ SCP180: Compute ROA intensity for SCP backward scattering.

◊ DCP180: Compute ROA intensity for double circular polarization (DCP) backward scattering.

◊ All: Compute the ROA intensity for all supported scatterings.

Freq=ReadAnharm Additional Input Ordering

The potential input sections for the various Freq=ReadAnharm additional input items should follow the keyword list section in the following order. Blank lines separate input sections, but each section and its terminating blank line should be included only when the corresponding keyword is specified.

Freq=ReadAnharm input keywords

blank line

checkpoint file list for DataSrc=InChk*n* or DataAdd=···=InChk*n*

blank line

data for DataSrc=In or DataAdd=···=In (harmonic followed immediately by anharmonic) blank line

data for DataAdd=Freq

blank line

n modes for RedDim=Active

blank line

n modes for RedDim=Inactive

blank line

n modes for RedDim=Frozen

blank line

keyword for DataMod=SkipPT2=OptModes (see below)

modes for DataMod=SkipPT2=Modes or Constants or DataMod=SkipPT2=OptModes

blank line

data for Resonances=List

blank line

DataMod=SkipPT2=OptModes Keywords: Removal of Contributions from Selected Normal Modes

One or more options are specified on a single line with no blank line following. The options available are:

◊ MinInd=*n*: Controls the minimum number of times a selected normal mode must appear to discard the derivative. The default is 1. For example, a value of 2 means that k_{ijk} is kept, but k_{iij} and k_{iii} are removed.

◊ EnOrd=*mask*: Controls the energy derivative orders to consider. The default is 1234 which says to include all energy derivatives. The value 14 says to only treat the first and fourth energy derivatives and to ignore the second and third energy derivatives.

Here is an example calculation using DataMod=SkipPT2=OptModes:

```
# Freq=(Anharm,ReadAnharm) ···

Formaldehyde

0 1
  C      -0.6067825443565   -0.0000000216230    0.0000000000000
  O       0.6033290944914    0.0000000215000    0.0000000000000
  H      -1.1752074085613    0.9201232113261    0.0000000000000
  H      -1.1752073429832   -0.9201232950844    0.0000000000000

PT2Model=GVPT2 Resonances=(NoDarDen,NoFermi)
DataMod=SkipPT2=OptModes

MinInd=2 EnOrd=34
5 6
```

Control keywords: keep only k_{ijk} for third and fourth derivatives

Modes for which to remove derivatives

terminating blank line

Specifying Resonances

For Resonances=Add, Resonances=List=Replace and Resonances=Delete, each line contains the type of resonance and the indexes of the normal modes involved in the resonances. Supported types are:

◊ 1-2 or 12: 1-2 Fermi resonance. 3 indexes needed, given in the order: $\omega_1 \approx \omega_2 + \omega_3$

◊ 1-1 or 11: 1-1 Darling-Dennison resonance. 2 indexes needed, given in the order: $\omega_1 \approx \omega_2$

◊ 1-3 or 13: 1-3 Darling-Dennison resonance. 4 indexes needed, given in the order: $\omega_1 \approx \omega_2 + \omega_3 + \omega_4$

◊ 2-2 or 22: 2-2 Darling-Dennison resonance. 4 indexes needed, given in the order: $\omega_1 + \omega_2 \approx \omega_3 + \omega_4$

For Resonances=List=Modify, an action must be specified at the beginning of the line, before the resonance type and the normal modes. The available actions are ADD (add resonance) and DEL (remove resonance if previously identified).

Examples

Example 1: Frequency data calculated in the current job:

```
%Chk=example1
# B3LYP/6-311+G(d,p) Freq=(Anharmonic,ReadAnharm)

Anharmonic frequencies example 1

molecule specification

DataMod=SkipPT2=Modes               Freq=ReadAnharm additional input
RedDim=Inactive=1                   # modes to make inactive
PT2Model=GVPT2
Resonances=Add
Print=InDataX=Ext                   Write data to an external file
                                    blank line
6                                   Modes to make inactive: RedDim=Inactive
                                    blank line
4 5                                 Modes for which to discard derivatives: DataMod=SkipPT2=Modes
                                    blank line
1-2 2 3 3                           List of additional resonances: Resonances=Add
                                    blank line
```

Example 2: Frequency data read from input stream. It uses the checkpoint file from Example 1 to retrieve the molecular geometry and Hessian:

```
%OldChk=example1
%Chk=example2
# B3LYP/6-311+G(d,p) Freq=(ReadFC,Anharmonic,ReadAnharm) Geom=Check

Anharmonic frequencies example 2

0 1

PT2Model=GVPT2                       Freq=ReadAnharm additional input
DataSrc=(InQMWX,NMOrder=Desc)        Read harmonic/anharmonic data from input
```

```
DataAdd=Freq                         Replace harmonic frequencies with input values
RedDim=Inactive=1
                                     blank line
@input_data.dat                      Data file from previous job with Print=InDataX=Ext
                                     blank line
1198.351                             List of harmonic frequencies: DataAdd=Freq
1259.285
1530.636
1814.590
2884.164
2941.556
                                     blank line
6
                                     blank line
```

Example 3: The following example specifies alternate values for several parameters:

```
#  B3LYP/6-31G(d) Freq=(Anharm,ReadAnharm)

Anharmonic frequencies example 3

molecule specification

Tolerances=Coriolis=0.25                   Freq=ReadAnharm additional input
Resonances=(DFreq12=220.0,K12Min=1.1)
DataMod=SCHarm=0.98
                                           blank line
```

Example 4: Anharmonic VCD and ROA spectra calculation:

```
%Chk=project4
# Freq=(ROA,VCD,Anharm,ReadAnharm) CPHF=RdFreq ···

Anharmonic VCD and ROA spectrum

0 1
molecule specification

589.3nm                              incident light frequency

PT2Model=GVPT2                       ReadAnharm input section
Spectro=MaxQuanta=3
                                     blank line
```

5.32.8 ReadFCHT Input

The following keywords for specifying various aspects of Freq=FC, HT or FCHT calculations are included as additional input within the Gaussian input file. They control various aspects of these analyses. Note that these keywords are different from those supported in Gaussian 09.

Specifying Calculation Options and Parameters

Transition=*type*: Specify the type of electronic transition.

◊ Absorption: Absorption. This is the default.

◊ Emission: Emission.

Spectroscopy=*type(s)*: Spectroscopy to simulate. The default is Spectroscopy=OnePhoton. The following items are available (names may be abbreviated to just the capital letters below: e.g., RR for ResonanceRaman):

◊ OnePhoton: One-photon process. This is the default.

◊ CircularDichroism: Circular Dichroism: electronic circular dichroism (ECD) for absorption and circularly polarized luminescence (CPL) for emission.

◊ OnePhotonAbsorption: One-photon absorption. Implies Transition=Absorption.

◊ OnePhotonEmission: One-photon emission. Implies Transition=Emission.

◊ ElectronicCircularDichroism: Electronic circular dichroism. Implies Transition=Absorption.

◊ CircularlyPolarizedLuminescence: Circularly polarized luminescence. Implies Transition=Emission.

◊ ResonanceRaman: Vibrational Resonance Raman. Options are specified via a separate item (see below).

ResonanceRaman=*options*: The following RR-specific options are available:

◊ CombOnly: Compute only combination bands.

◊ NoComb: Skip computation of combination bands.

◊ aMean=*coeff*: Set the coefficient of the mean polarizability. The default is 45.

◊ Damping=*value*: Damping constant or lifetime of intermediate states (in cm^{-1}). The default is 500 cm^{-1}.

◊ dAnti=*coeff*: Set the coefficient of the antisymmetric anisotropy. The default is 5.

◊ gSymm=*coeff*: Set the coefficient of the symmetric anisotropy. The default is 7.

◊ Omega=*value*: Incident energy (in cm^{-1}). By default, it is calculated as the difference in energy between the vibrational fundamental states of the initial and intermediate states.

◊ OmegaList: Reads a list of incident energies (in cm^{-1}) in the input stream to build an energy profile.

◊ OmegaMax=*value*: Maximum energy (in cm^{-1}) for the incident energy profile.

◊ OmegaMin=*value*: Minimum energy (in cm^{-1}) for the incident energy profile.

◊ OmegaNum=*n*: Number of energies in the incident energy profile.

◊ OmegaStep=*value*: Energy interval (in cm^{-1}) between two energies for the incident energy profile.

◊ Scattering=*params*: Scattering geometry for the Resonance Raman simulation. Available items are (note that the asterisks below are part of the parameter names):

- ICP0: Compute RR intensity for Incident Circular Polarization (ICP) forward scattering.
- ICP90x: Compute RR intensity for ICP right-angle scattering (polarized).
- ICP90z: Compute RR intensity for ICP right-angle scattering (depolarized).
- ICP90*: Compute RR intensity for ICP right-angle scattering (magic angle).
- ICP180: Compute RR intensity for ICP backward scattering.
- SCP0: Compute RR intensity for Scattered Circular Polarization (SCP) forward scattering.
- SCP90x: Compute RR intensity for SCP right-angle scattering (polarized).
- SCP90z: Compute RR intensity for SCP right-angle scattering (depolarized).
- SCP90*: Compute RR intensity for SCP right-angle scattering (magic angle).
- SCP180: Compute RR intensity for SCP backward scattering.

- DCP180: Compute RR intensity for Double Circular Polarization (DCP) forward scattering.
- All: Compute the RR intensity for all supported scatterings.

Temperature: Specify temperature and related parameters:

◊ Value=*temp*: Temperature of simulation in K. The default is the value specified to Gaussian with the Temp keyword or via additional input; the Gaussian default is 298.15 K.

◊ MinPop=*ratio*: Minimum ratio between the Boltzmann populations of any vibrational initial state and the fundamental state for the former to be considered in the calculations. In other words, this item specifies the fraction of a vibrational state that must be populated in order for it to be treated as the starting point of a transition. The default value is 0.1 (10%).

TransProp=*definition*: Definition of the transition dipole moment(s). The first three items effectively select among Franck-Condon, Herzberg-Teller and FCHT analyses. However, the corresponding options to the Freq keyword are preferable. The following items are available:

◊ FC: The dipole moment is assumed constant during the electronic transition. This selection describes dipole-allowed transitions well. It is the default.

◊ FCHT: Computes zeroth- and first-order terms of the Taylor expansion of the transition dipole moment about the equilibrium geometry. It is needed to correctly treat weakly-allowed electronic transitions or CD spectra.

◊ HT: Implements linear variation of the dipole moment with respect to the normal mode. This item corresponds to the first-order term of the Taylor expansion of the transition dipole moment about the equilibrium geometry.

◊ DipA=*source*: Explicit definition of the transition dipole moment d_A. The following options for the data source are available:

- Auto: Gaussian will choose the definition depending on the simulation parameters. This is the default.
- Read: Read data from main input source (see “Data Sources” below).
- Input: Read data from input stream.

◊ DipB=*source*: Definition of the transition dipole moment d_B. This item accepts the same data source options as DipA.

◊ EDip=*source*: Definition of the transition electric dipole moment. This item accepts the same data source options as DipA.

◊ MDip=*source*: Definition of the transition magnetic dipole moment. This item accepts the same data source options as DipA.

◊ NoUse: Discard the transition dipole read in the data source and replace it with a unitary one. This is the default behavior for electronic transition with a change in multiplicity or charge. Keyword is not supported for HT and FCHT calculations.

Method=*method*: Selects the representation model for the electronic transition. The non-default options specify methods for approximating the excited state frequencies based on the ground state. These are not generally preferable to modeling the excited state explicitly. The following methods are available:

◊ AdiabaticHessian: Both PESs – ground state and excited state – are calculated at the harmonic level about their respective minimum. This is the default.

◊ AdiabaticShift: Both PESs are calculated at the harmonic level about their respective minimum, but the PES of the final state is assumed to be the same as the initial state. Only the equilibrium geometry of the final state is calculated.

◊ VerticalHessian: The PES of the final state is evaluated about the equilibrium geometry of the initial state.

◊ VerticalGradient: The PES of the final state is evaluated about the equilibrium geometry of the initial state, but the PES of the final state is assumed to be the same as the initial state. Only the energy gradient of the final state is calculated about the equilibrium geometry of the initial state. (This method is also known as the linear coupling model.)

For emission spectra, the form VerticalGradient=Abs causes Gaussian to compute the 0-0 transition energy in the same way as absorption spectra in order to get the correct 0-0 transition energy. It is needed when the emission spectrum is incorrectly computed as an absorption spectrum (equilibrium geometry of the ground state, frequencies of the ground state and forces of the excited state).

Prescreening=*params*: Sets the prescreening criteria for choosing the most intense transitions. Only available for the time-independent framework. Available parameters:

◊ MaxC1=*n*: Maximum quantum number reached in C_1 (C_1^{max}). The default is 20.

◊ MaxC2=*n*: Maximum quantum number reached by both modes involved in the combination bands in C_2 (C_2^{max}). The default is 13.

◊ MaxInt=*millions*: Maximum number of integrals to compute for each class above C_2 (N_I^{max}), in units of one million. The default value is 100 (100,000,000).

TimeIndependent: Use the time-independent framework. This is the default for one-photon spectroscopy. TI is a synonym for this item.

TimeDependent=*params*: Compute the band shape using the path-integral approach instead of the sum-over-states approach. This is the default for Resonance Raman spectroscopy. TD is a synonym for this item. Available parameters are:

◊ 2NStep=*m*: Use 2^m steps for the integration. *m* defaults to 18 for OPA, OPE, ECD and CPL and 12 for RR.

◊ 2NStepWin=*k*: Set the number of steps in which the correlation function X(t) is actually computed to as 2^k. The function is 0 outside this range. Note that *k* must be $\leq m$ from 2NStep above. By default, *k* is set equal to *m*.

◊ GauHWHM=*n*: Inhomogenous broadening, applied as a dephasing (in cm^{-1}). The default is 135 cm^{-1}.

◊ LorHWHM=*n*: Homogenous broadening, applied as a dephasing (in cm^{-1}). The default is 0 cm^{-1}.

◊ Time=*seconds*: Time interval Δt in seconds. The default value is $2^m \times 10^{-17}$, where *m* is from 2NStep above.

Termination=DeltaSP=*value*: Sets the termination criteria. Set the minimum difference between two consecutive classes of the final state to continue the calculation. The default is 0.0 (always continue).

Data Sources

The items in this section specify the locations and methods for obtaining various data used by the FCHT analysis.

Initial=*items*: Data source(s) and/or parameters for the initial state.

Final=*items*: Data source(s) and/or parameters for the final state.

Intermediate=*items*: Data sources(s) and/or parameters related to the intermediate state. Only valid for Resonance Raman.

The following parameters are available for the three preceding items:

◊ Source=*source*: Data source for the state:
  - Calc: Current calculation. This is the default for Absorption spectra.
  - Chk: Checkpoint file (the filename is given in the input stream). This is the default for Emission spectra.
  - LogFile: Gaussian output filename (the filename is given in the input stream).

◊ Freq=*params*: Source and handling of vibrational frequency data:
  - Read: Read frequencies from the main data source. This is the default.
  - Input: Read frequencies from the input stream.
  - Scale: Scale frequencies using a mode-specific extrapolated scaling factor based on the Duschinsky transformation. The reference state used is the other electronic state. For example, Final=Freq=Scale uses the input initial state frequencies to scale those of the final state. The required frequencies are taken from the input stream in their usual position (see below).

◊ MaxBands=*state*: Set the highest class state to consider. The default is 3 for Initial and 7 otherwise.

◊ MaxStates=*n*: Maximum number of initial vibrational states actually considered in the calculations. Only valid with Initial. Note that this value is not the number of configurations printed by Gaussian. The default is 50.

◊ ExcState=*state*: Excited electronic state actually treated in the Gaussian output file. Only used if the data source for the excited state is a Gaussian output file.

DataAdd=DeltaE=*value*: Difference in energy between the electronic states (in Hartrees). By default, it is calculated from the data sources used for the initial and final states.

DataMod=Duschinsky=*params*: Duschinsky matrix to use in the calculation, By default, the true Duschinsky matrix is used. Note that the definition of the Duschinsky matrix depends on the model used to describe the transition. Other options are:

◊ Identity: Use the identity matrix as the Duschinsky matrix.

◊ Diagonal: Swap the columns of the correct Duschinsky matrix to be as diagonal as possible and replace it by the identity matrix. If this cannot be done because the matrix is not diagonal enough, an error occurs.

Superpose=*params*: Specifty the superposition procedure.

◊ Reorient: Reorient molecules to maximize overlap between the two structures. This is the default.

◊ NoReorient: Do not try to superpose the structures.

◊ Rotation=*n*: Rotation algorithm used for the superposition:
  - 0: Alternate usage. First quaternions, then rotation angles. This is the default.
  - 1: Use the program default (be aware that this may change between Gaussian versions).
  - 2: Use quaternions only.
  - 3: Use rotation angles only.

◊ RotNIter=*xxxyyy*: Maximum number of iterations for each superposition algorithm, where *xxx* is the number of iterations for the algorithm based on quaternions, and *yyy* is the number of iterations for the angle-based algorithm. The default parameter is 030100 or 30 for quaternions and 100 for rotation angles.

Output Selection

Print=*params*: This item controls the information included in the output.

◊ Spectra=*params*: Spectra to be included (the non-default items are not valid in the time-dependent framework nor for Resonance Raman):

- Final: Print only the final spectrum. This is the default.
- All: Print the maximum number of class-specific spectra.
- ClassI: Print one spectrum for each class of the initial state.
- ClassF: Print one spectrum for each class of the final state.

◊ Matrix=*list*: Matrices to be displayed: J, K, A, B, C, D, E. The string is composed of the matrices of interest, e.g., JK. A, B, C, D and E are the Sharp and Rosenstock matrices, J is the Duschinsky matrix, and K is the shift vector.

◊ HuangRhys: Print Huang-Rhys factors [459].

◊ AssignThresh=*n*: Include data (assignment, energy, intensity) of the transitions, which contribute at least to 100*n*% to the total intensity of the spectrum obtained for each initial vibrational state. The default is 0.01 (1%).

◊ TDAutoCorr=*n*: Print *n* points of the time-dependent autocorrelation function. The default is 0 (output disabled).

◊ Tensors: Print the transition tensors instead of the invariants. Only available for Resonance Raman.

◊ Conver: Print convergence data. Only available for Resonance Raman.

◊ Color=*params*: Specify RGB standard for color output:

- sRGB: Use the sRGB standard (white reference: D65, gamma correction based on the IEC 61966-2-1 standard). This is the default.
- None: Do not print the color in an RGB format.
- CIE: Use the CIE RGB standard (white reference: D65, no gamma correction).
- HDTV: Use the HDTV RGB standard (white reference: D65, gamma correction based on the ITU-R BT.709-3 standard).

Spectrum=*params*: Control spectrum layout.

◊ Lower=*value*: Energy of the lower bound of the spectrum (in cm^{-1}). For absorption, the default is -1000 cm^{-1}. For emission, the default is -8000 cm^{-1}. The bound is defined with respect to the 0-0 transition.

◊ Upper=*value*: Energy of the upper bound of the spectrum (in cm^{-1}). For absorption, the default is +8000 cm^{-1}. For emission, the default is +1000 cm^{-1}. The bound is defined with respect to the 0-0 transition.

◊ AbsBound: Deactivate the default behavior of Gaussian to define the lower and upper bounds of the spectrum with respect to the 0-0 transition.

◊ Grain=*value*: Energy difference between two adjacent points for the discretization of the spectrum (in cm^{-1}). The default is 8cm^{-1}.

◊ Broadening=*params*: Discribution function used to compute the band-shape:

- Gaussian: Use normal distribution functions to simulate the inhomogeneous broadening. This is the default.
- Lorentzian: Use Cauchy distribution functions to simulate the homogeneous broadening.
- Stick: Do not simulate the band broadening. Print the bands as sticks.

◊ HWHM=*value*: Half-width at half-maximum of the distribution function used for the convolution (in cm^{-1}). The default is 135 cm^{-1}.

Reduced-Dimensionality Schemes

RedDim=*params*: Activates and sets a reduced-dimensionality scheme.

◊ Block: Use the correct Duschinsky matrix to find the projection of a list of modes given in the input stream on the other state to reduce consistently the dimension of the problem. By default, the state of reference is the lower one.

◊ ClearLowFreq=*n*: Remove all normal modes with a frequency below n cm^{-1} in absolute value. If this parameter is specified but n is omitted, it defaults to 50.

◊ ClearImFreq: Remove at most one normal mode per state with an imaginary frequency. If there is one mode with an imaginary frequency in each state, Gaussian checks that they are uncoupled to other modes and project themselves on each other in the electronic transition.

◊ BlockThresh=*value*: Threshold for the block(s) construction (between 0 and 1). A high value ensures that the selected modes are uncoupled from the remaining ones. The default is 0.9.

◊ BlockTol=*n*: Specifies the maximum ratio allowed between the final number of selected modes and the initial set: if more than $n \times \#modes$ chosen by the user are selected, Gaussian stops the calculations. The default is 1.6.

Freq=ReadFCHT Additional Input Ordering

The potential input sections for the various Freq=ReadFCHT additional input items should follow the keyword list section in the following order. Blank lines separate input sections, but each section and its terminating blank line should be included only when the corresponding item is specified.

Freq=ReadFCHT input keywords
    blank line
filename for the Chk or LogFile parameters to Input=Source
filename for the Chk or LogFile params. to Final=Source (OPA, OPE, ECD, CPL) or Intermediate=Source (RR)
    blank line
initial state frequency list: Initial=Freq=Input and/or Final=Freq=Scale
    blank line
final/intermediate state frequency list: the Freq=Input option to Final (OPA, OPE, ECD, CPL) or Intermediate (RR) and/or Initial=Freq=Scale
    blank line
transition dipole moment data for TransProp=DipA or TransProp=EDip
    blank line
transition dipole moment data for TransProp=DipB or TransProp=MDip
    blank line
list of modes for RedDim=Block
    blank line
incident energies for ResonanceRaman=OmegaList

blank line

Examples

The following input file illustrates the use of additional input for FC analyses.

Example 1. The following calculation performs a Franck-Condon frequency analysis for phenoxyl:

```
%Chk=phenoxyls0.chk
# Freq=(FC,ReadFC,ReadFCHT) Geom=Check ···

Phenoxyl Frank-Condon analysis

0 2

Final=Source=Chk                  ReadFCHT input: specify source for final state.
Print=(Spectra=All,Matrix=JK)     Output all spectra, the Duschinsky matrix and the shift vector.

phenoxyls1.chk                    Checkpoint file for the final state.
     terminal blank line
```

Example 2. The following job predicts the ECD spectrum (selected by the item in the additional input section). Note that the second checkpoint file for the final section is specified in the subsequent input section. It is read even though there is no explicit item in the additional input section since Final=Source=Chk is the default for emission spectra.

```
%Chk=initial
# Freq=(FCHT,ReadFC,ReadFCHT) Geom=AllCheck ···

Transition=Absorption          ReadFCHT input.
Spectroscopy=CD                Select spectrum to predict: ECD.

final.chk                      Checkpoint file for the final state.
     terminal blank line
```

Example 3. The following job performs the analysis at 500 K, using the time-dependent framework.

```
%Chk=temp500init
# Freq=(FC,ReadFC,ReadFCHT,SaveNM) Geom=AllCheck ···

Temperature=Value=500.0

temp500final.chk
     terminal blank line
```

Example 4. The following job computes the resonance Raman spectrum:

```
%Chk=S0_freq
# Freq=(FC,ReadFC,ReadFCHT) Geom=Check ···

RR spectrum
```

```
0 1

Spectroscopy=RR                    ReadFCHT input: select spectrum to predict.
TransProp=EDip=Input
RR=OmegaList
TD=(2NSTEP=12,2NSTEPWIN=12,Time=1.0d-12,GauHWHM=100)
Print=(Tensors,Matrix=JK)

S2_freq.chk                        Checkpoint file for final state.

1.000D0 1.000D0 0.00D0             Transition dipole for TransProp=EDip.

55000 55500 57000                  List of incident energies for RR=OmegaList.
     terminal blank line
```

Example 5. The final job step below computes the vibrational envelope of a photoionization spectrum:

```
%chk=neutral         Calculation on the neutral form (initial state).
# B3LYP/6-31+G(d,p) Freq=SaveNM

neutral form

0 1
molecule specification

--Link1--
%chk=cation          Calculation on the cation (final state).
# B3LYP/6-31+G(d,p) Freq=SaveNM

cation

1 2
molecule specification

--Link1--
%oldchk=neutral      Data for the neutral form.
%chk=fc
# B3LYP/6-31+G(d,p) Freq=(FC,ReadFCHT) Geom=Check

photoionization

0,1

Initial=Source=Calc Final=Source=Chk                    Retrieve final state from a checkpoint file.
Spectrum=(Lower=-500.,Upper=+5000.,Grain=1.,HWHM=50.)   Parameters for computed spectrum.
Prescreening=(MaxC1=30.,MaxC2=20.)                      Parameters to select transitions.
Print=Matrix=JK                                         Output Duschinsky matrix and shift vector.
```

```
cation.chk
```
Checkpoint filename for final state.

terminal blank line

5.33 G09Defaults

This keyword restore the calculation defaults from Gaussian 09. It is equivalent to:

```
Integral=(FineGrid,Acc2E=10) Constants=2006 SCRF=G09Defaults
```

5.33.1 Related Keywords

Integral, Constants, SCRF

5.34 Gen and GenECP

A set of standard basis sets is stored internally in Gaussian (see the *Basis Sets* page); these basis sets may be specified by including the appropriate keyword within the route section for the calculation. The Gen keyword allows a user-specified basis set to be used in a Gaussian calculation. It is used in the place of a basis set keyword or a density fitting basis set keyword. In this case, the basis set description must be provided as input (in a separate basis set input section).

Gen may be used in a completely analogous way to specify an alternate density fitting basis set (see the examples).

The GenECP variation may be used to read in both basis functions and ECPs; it is equivalent to Gen Pseudo=Read. It is designed for use in ONIOM calculations in which you want to use a general basis set with ECPs within one ONIOM layer.

Option

NZCore=*N*

Specifies that the basis set is *N*-zeta in the core. The default is 1 (minimal) except for internally stored multiple-zeta core basis sets.

The GFPrint keyword may be used to include the gaussian function table within the output file. The GFInput keyword may be used to have the table printed in a form that is suitable for input to Gen. The ExtraBasis keyword may be used to make additions to standard basis sets. Similarly, the ExtraDensityBasis keyword may be used to make additions to standard density fitting basis sets.

A brief general overview of basis functions is provided as the final subsection of this discussion.

5.34.1 Input

External basis sets are read into Gaussian by specifying Gen (for general basis) in the route section. The keywords 5D, 6D, 7F, and 10F are used to specify use of Cartesian or pure d and f (and higher) functions; the defaults are 5D and 7F. All d-shells in a calculation must have the same number of functions. Similarly, f- and higher shells must either be all Cartesian or all pure.

Defining a shell. External basis input is handled by the routine GenBas in Link 301. The basic unit of information that it reads from the basis set input section is the *shell definition block*. A shell definition block, together with the global specification of pure vs. Cartesian functions, contains all necessary information to define a shell of functions. It consists of a *shell descriptor* line, and one or more *primitive gaussian* lines:

IType NGauss Sc	*Shell descriptor line: shell type, # primitive gaussians, and scale factor.*
α_1 $d_{1\mu}$	*Primitive gaussian specification: exponent and contraction coefficient.*
α_2 $d_{2\mu}$	
...	
α_N $d_{N\mu}$	*There are a total of NGauss primitive gaussian lines.*

IType defines the shell type and shell constraint and may be S, P, D, SP, F, G, $\cdots$, for an s-shell, p-shell, d-shell, sp-shell, f-shell, g-shell, and so on. *NGauss* specifies the number of primitive gaussian shells (the degree of contraction) for the shell being defined. The shell scale factor is given by *Sc* (i.e., all primitive exponents are scaled by Sc^2).

The subsequent *NGauss* primitive gaussian lines define the exponents α_k and contraction coefficients, $d_{k\mu}$. Each line provides the exponent for one primitive, followed by its contraction coefficient (or s and p coefficients for an sp-shell).

A second format also exists to specify a shell as a least-squares gaussian expansion of a Slater orbital. This is requested by a shell descriptor line of the form STO, *IOrb, NGauss, Sc*. *IOrb* is one of 1S, 2S, 2P, 2SP, 3S, 3P, 3SP, 3D, 4SP, and specifies which expansion is requested. Note that 2SP requests the best least-squares fit simultaneously to S and P slater orbitals and is not equivalent to separately specifying the best S and the best P expansions. *NGauss* is the same as above. Gaussian expansions of Slater functions having from 1 to 6 primitives are available. *Sc* is the scale factor and hence the exponent of the slater function being expanded. No primitive gaussian lines are required after a shell descriptor line requesting an STO expansion.

Defining the basis for an atom or atom type. One customarily places at least one, and often several, shells on any given nuclear center ("atom"), via a *center definition block*. A center definition block consists of a *center identifier line* and one *shell definition block* for each shell desired on the center(s) specified. It is terminated by a line with either asterisks or plus signs in columns 1 through 4:

c_1 c_2 $\cdots$ 0	*Center identifier line: specifies applicability for these shells.*
IType NGauss Sc	*First shell definition block.*
α_2 $d_{2\mu}$	
...	
α_N $d_{N\mu}$	
...	*Additional shell definition blocks.*
IType NGauss Sc	*Final shell definition block.*
α_2 $d_{2\mu}$	
...	
α_N $d_{N\mu}$	
****	*Separator: terminates the center definition block.*

The center identifier line specifies a list of centers on which to place the basis functions in the center definition block, terminated by a 0. It can contain one or more integers, which are used to indicate the corresponding atom(s) in the molecule specification; more commonly, it contains a list of atomic symbols to refer to all atoms of a specific type. Center numbers and atomic symbols may be freely intermixed within a single center identifier line.

To help detect input mistakes, if a center definition block specifies an atom that is not present in the molecule, the run is aborted. If the center is preceded by a minus sign (e.g. -H), the basis set information is simply skipped if no atom of that type is present in the molecule specification (the terminal zero may also be omitted in this case). The latter syntax is intended for creating basis set include files that specify a standard basis set for many atoms; once built, it can be included in its entirety in the input stream when the basis set is desired, via the include () function (as described earlier in this chapter).

A center or atom type may be specified in more than one center definition block. For example, in the Gaussian 09 basis set directory – *$g09root/g09/basis* on UNIX systems – there is one file which specifies 6-31G as a general basis set (*631.gbs*) and another file containing d exponents which would be included as well to specify 6-31G* (*631s.gbs*). Every atom from H through Cl is specified in both files, and in practice both of them would be included (most often along with additional basis set specifications for those atoms in the molecule for which the 6-31G basis set is not available).

Basis set transformation option to the Integral keyword. Several options to the Integral keyword control whether/how generalized contraction basis sets are transformed to reduce the number of primitives. Int=BasisTransform=*N* says to transform generalized contraction basis sets to reduce the number of primitives, neglecting primitives with coefficients of 10^{-N} or less. This is the default, with *N*=4. Int=ExactBasisTransform says to transform generalized contraction basis sets to reduce the number of primitives, but using only transformations which are exact (do not change the computed energy). Finally, Int=NoBasisTransform says not to transform generalized contraction basis sets to reduce the number of primitives.

Drawing on Pre-Defined Basis Sets in Gen Input. Gaussian adds flexibility to general basis set input by allowing them to include pre-defined basis sets within them. Within a center definition block for an atom type (or types), an entire shell definition block may be replaced by a line containing the standard keyword for a pre-defined basis set. In this case, all of the functions within the specified basis set corresponding to the specified atom type(s) will be used for all such atoms within the molecule.

The SDD, SHF, SDF, MHF, MDF, MWB forms may be used to specify Stuttgart/Dresden basis sets/potentials within general basis set input. Note that the number of core electrons must be specified. The ECP potential name def2 or the synonym QZV can be used in GenECP input to request the potentials which are used with both the def2 and QZV basis sets.

5.34.2 Examples

Here is a portion of the Gen input corresponding to the 6-31+G(d) basis set:

```
H 0                                          Applies to all hydrogen atoms.
S    3 1.00
 0.1873113696D+02  0.3349460434D-01
 0.2825394365D+01  0.2347269535D+00
 0.6401216923D+00  0.8137573262D+00
S    1 1.00
 0.1612777588D+00  0.1000000000D+01
****
```

```
C 0                                  Applies to all carbons.
S    6 1.00                          6-31G functions.
 0.3047524880D+04  0.1834737130D-02
 0.4573695180D+03  0.1403732280D-01
 0.1039486850D+03  0.6884262220D-01
 0.2921015530D+02  0.2321844430D+00
 0.9286662960D+01  0.4679413480D+00
 0.3163926960D+01  0.3623119850D+00
SP   3 1.00
 0.7868272350D+01 -0.1193324200D+00  0.6899906660D-01
 0.1881288540D+01 -0.1608541520D+00  0.3164239610D+00
 0.5442492580D+00  0.1143456440D+01  0.7443082910D+00
SP   1 1.00
 0.1687144782D+00  0.1000000000D+01  0.1000000000D+01
D    1 1.00                          Polarization function.
 0.8000000000D+00  0.1000000000D+01
****
C 0                                  Applies to all carbons.
SP   1 1.00                          Diffuse function.
 0.4380000000D-01  0.1000000000D+01  0.1000000000D+01
****
```

The following Gen input uses the 6-31G(d,p) basis set for the carbon and hydrogen atoms and the 6-31G‡ basis set for the fluorine atoms in the molecule and places an extra function only on center number 1 (which happens to be the first carbon atom in the molecule specification for 1,1-difluoroethylene):

```
C H 0
6-31G(d,p)
****
F 0
6-31G(d',p')
****
1 0      Place a diffuse function on just one carbon atom.
SP   1 1.00
 0.4380000000D-01  0.1000000000D+01  0.1000000000D+01
****
```

The following job uses the Gaussian include file mechanism to specify the basis functions for chromium:

```
# Becke3LYP/Gen Opt Test

HF/6-31G(*) Opt of Cr(CO)6

molecule specification
```

```
C O 0
6-31G(d)
****
@/home/gwtrucks/basis/chrome.gbs/N
```

Note that *.gbs* is the conventional extension for basis set files (for *gaussian basis set*).

The following example uses general basis set input to specify both the basis set and the density fitting basis set.

```
# RBLYP/GEN/GEN 6D

HCl: reading in 6-31g* AO basis and DGA1 fitting set.
6D is specified because the default for general basis
input is 5D but the 6-31g* basis is defined to use 6D

0,1
cl
h,1,1.29

! here are the 6-31g* basis sets for Cl and H
cl 0
 S   6 1.00
      0.2518010000D+05  0.1832959848D-02
      0.3780350000D+04  0.1403419883D-01
      0.8604740000D+03  0.6909739426D-01
      0.2421450000D+03  0.2374519803D+00
      0.7733490000D+02  0.4830339599D+00
      0.2624700000D+02  0.3398559718D+00
 SP   6 1.00
      0.4917650000D+03 -0.2297391417D-02  0.3989400879D-02
      0.1169840000D+03 -0.3071371894D-01  0.3031770668D-01
      0.3741530000D+02 -0.1125280694D+00  0.1298800286D+00
      0.1378340000D+02  0.4501632776D-01  0.3279510723D+00
      0.5452150000D+01  0.5893533634D+00  0.4535271000D+00
      0.2225880000D+01  0.4652062868D+00  0.2521540556D+00
 SP   3 1.00
      0.3186490000D+01 -0.2518280280D+00 -0.1429931472D-01
      0.1144270000D+01  0.6158925141D-01  0.3235723331D+00
      0.4203770000D+00  0.1060184328D+01  0.7435077653D+00
 SP   1 1.00
      0.1426570000D+00  0.1000000000D+01  0.1000000000D+01
 D   1 1.00
      0.7500000000D+00 0.1000000000D+01
****
h 0
 S   3 1.00
```

```
      0.1873113696D+02  0.3349460434D-01
      0.2825394365D+01  0.2347269535D+00
      0.6401216923D+00  0.8137573261D+00
 S   1 1.00
      0.1612777588D+00  0.1000000000D+01
****

! here are the DGA1 fitting sets for Cl and H
cl 0
 S   1 1.00
      0.2048000000D+05  0.1000000000D+01
 S   1 1.00
      0.4096000000D+04  0.1000000000D+01
 S   1 1.00
      0.1024000000D+04  0.1000000000D+01
 S   1 1.00
      0.2560000000D+03  0.1000000000D+01
 S   1 1.00
      0.6400000000D+02  0.1000000000D+01
 SPD   1 1.00
      0.2000000000D+02  0.1000000000D+01  0.1000000000D+01  0.1000000000D+01
 SPD   1 1.00
      0.4000000000D+01  0.1000000000D+01  0.1000000000D+01  0.1000000000D+01
 SPD   1 1.00
      0.1000000000D+01  0.1000000000D+01  0.1000000000D+01  0.1000000000D+01
 SPD   1 1.00
      0.25000D+00  0.10000D+01  0.10000D+01  0.10000D+01 0.0D+01  0.10000D+01
 ****
h 0
 S   1 1.00
      0.4500000000D+02  0.1000000000D+01
 S   1 1.00
      0.7500000000D+01  0.1000000000D+01
 S   1 1.00
      0.1500000000D+01  0.1000000000D+01
 S   1 1.00
      0.3000000000D+00  0.1000000000D+01
 ****
```

If you wanted to specify the density fitting basis set with general basis set input, then you would use a route section like this one (substituting the appropriate basis set for your problem):

```
# RBLYP/6-31G(d,p)/Gen 6D
```

5.34.3 Related Keywords

ExtraBasis, ExtraDensityBasis, GFInput, GFPrint, Integral, Pseudo

5.34.4 Basis Function Overview

A single *basis function* is composed of one or more *primitive gaussian functions*. For example, an s-type basis function φ_μ(r) is:

$$\varphi_\mu(r) = \sum_{i=1}^{N} d_{i\mu} e^{-\alpha_{i\mu} f_\mu^2 r^2}$$

S-Type Basis Function

N is the number of primitive functions composing the basis function, and it is called the *degree-of-contraction* of the basis function. The coefficients $d_{i\mu}$ are called *contraction coefficients*. The quantities $\alpha_{i\mu}$ are the *exponents*, and f is the *scale factor* for the basis function. The maximum degree-of-contraction permitted in Gaussian is 100.

A *shell* is a set of basis functions φ_μ with shared exponents. Gaussian supports shells of arbitrary angular momentum: s, p, d, f, g, h, and so on. An s-shell contains a single s-type basis function. A p-shell contains the three basis functions p_X, p_Y, and p_Z. An sp-shell contains four basis functions with common gaussian exponents: one s-type function and the three p-functions p_X, p_Y, and p_Z.

A d-shell may be defined to contain either the six second-order functions:
$(d_{X^2}, d_{Y^2}, d_{Z^2}, d_{XY}, d_{XZ}, d_{YZ})$
or the five so-called “pure d” basis functions:
$(d_{z^2-r^2}, d_{x^2-r^2}, d_{xy}, d_{xz}, d_{yz})$

Likewise, an f-shell may contain either the 10 third-order gaussians or the 7 “pure f” functions. Higher-order shells function similarly. Note that the contraction coefficients in a shell must be the same for all functions of a given angular momentum, but that s and p contraction coefficients can be different in an sp-shell. A scale factor is also defined for each shell. It is used to scale all the exponents of primitives in the shell. The program has the ability to convert between the two types of functions [114].

Consider the series of basis sets STO-3G, 6-31G, and 6-311G(d) for the carbon atom. With the STO-3G basis, there are two shells on a carbon atom. One is an s-shell composed of 3 primitive gaussian functions (which are least-squares fit to a Slater 1s orbital). The other sp-shell is a least-squares fit of 3 gaussians to Slater 2s and 2p orbitals with the constraint that the s and p functions have equal exponents. These expansions are the same for all atoms. Only the scale factors for each shell differ from atom to atom. For carbon atoms, the 1s- and 2sp-shells have scale factors of 5.67 and 1.72, respectively. The 6-31G basis on a first row atom has three shells. One shell is a contraction of six primitive s-type gaussians. The second shell is a combination of three primitive sp-shells. The third shell consists of a single sp-function. These functions were optimized for the atom. Scale factors of 1.00, 1.00, and 1.04, respectively, for each shell for carbon were then determined by molecular calculations. As its name implies, the 6-311G(d) basis has 5 shells: an s-shell with 6 primitives, 3 sp-shells with 3, 1, and 1 primitives, and an uncontracted d-shell. All shells are unscaled (have unit scale factor).

5.35 GenChk

This keyword is found in the second and later automatically generated job steps for compound calculation like Opt Freq. It serves to ensure that all basis set, ECP, and fitting set information (as applicable) is read from the checkpoint file while retaining the keywords for internally stored components (regardless of whether they were modified in the route) so that they can be reported in the output and archive entries for these later calculations. This keyword serves no purpose within route sections created by users.

5.35.1 Example

The following output from the second job step of an Opt Freq calculation illustrates the GenChk keyword:

```
Link1:  Proceeding to internal job step number  2.
--------------------------------------------------------------
#N Geom=AllCheck Guess=Read SCRF=Check GenChk RHF/STO-3G Freq
--------------------------------------------------------------
```

5.36 Geom

The Geom keyword specifies the source of the molecule specification input, options related to coordinate definitions, and geometry-related output. By default, it is read from the input stream, as described previously. Geom may be used to specify an alternate input source. It also controls what geometry-related information is printed and use of internal consistency checks on the Z-matrix. The Geom keyword is not meaningful without at least one item selection option.

Gaussian 16 supports generalized internal coordinates (GIC), a facility which allows arbitrary redundant internal coordinates to be defined and used for optimization constraints and other purposes [460]. There are several GIC-related options to Geom, and the *GIC Info* subsection describes using GICs as well as their limitations in the present implementation.

5.36.1 Options

Geometry Retrieval Options

Checkpoint

Take the molecule specification (including variables) from the checkpoint file. Only the charge and multiplicity are read from the input stream. For example, Geom=Checkpoint may be used by a later job step to retrieve the geometry optimized during an earlier job step from the checkpoint file. Checkpoint may be combined with the ModRedundant option or with the AddGIC option if you want to retrieve and alter the molecule specification in a checkpoint file using internal coordinate-style modifications. Note that Geom=(Checkpoint,ModRedundant,GIC) is equivalent to Geom=(Checkpoint,AddGIC). The Geom=(Checkpoint,GIC) option (i.e., without ModRedundant) will not use the GICs from the checkpoint file, but instead, it will build a new set of GICs automatically based on the current molecular specification (e.g., Cartesian coordinates) from the checkpoint file. The Geom=(Checkpoint,ReadAllGIC) option will not use the GICs from the checkpoint file, but instead, it will read a new set of GICs from the input file and calculate their values based on the current molecular specification (e.g., Cartesian coordinates) from the checkpoint file. Note that Geom=(Checkpoint,GIC) is equivalent to Geom=Checkpoint Opt=GIC, and Geom=(Checkpoint,ReadAllGIC) is equivalent to Geom=Checkpoint Opt=ReadAllGIC.

AllCheck

Take the molecule specification (including variables), the charge and multiplicity, and the title section from the checkpoint file. Thus, only the route section and any input required by keywords within it need be specified when using this option. This option is not valid with Modify but may be combined with ModRedundant, GIC, or AddGIC in the same way as the Checkpoint option.

Step=*N*

This option retrieves the structure produced by the N^{th} step of a failed or partial geometry optimization (it is not valid for a successful optimization). Step=Original recovers the initial starting geometry. This option is used for restarting geometry optimization from intermediate points. It must be combined with one of Checkpoint, AllCheck, or Modify. Note that not all steps are always present in the checkpoint file; a Hessian updated message in the log file means that the corresponding step is available in the checkpoint file. NGeom=*N* retrieves the N^{th} geometry from an optimization checkpoint file using the same record of points used for display in GaussView, where *N*=1 corresponds to the input molecule specification. Geom=Step=*M* is automatically converted to Geom=NGeom=*M+1* if the previous optimization used redundant internal coordinates.

Modify

Note that this option refers to modification specifications for geometry optimizations using Z-matrix coordinates only. In general, it is deprecated in favor of ModRedundant. It should not be used with GICs. Use Geom=(Checkpoint,AddGIC) instead of Geom=Modify if you want to modify the generalized internal coordinates.

This option specifies that the geometry is to be taken from the checkpoint file and that modifications will be made to it. A total of two input sections will be read: the first contains the charge and multiplicity, and the second contains alterations to the retrieved geometry.

Modification specifications for geometry optimizations using Z-matrix coordinates have the following form:

variable [*new-value*] [**A**|**F**|**D**]

where *variable* is the name of a variable in the molecule specification, *new-value* is an optional new value to be assigned to it, and the final item is a one-letter code indicating whether the variable is to be active (i.e., optimized) or frozen; the code letter D requests numerical differentiation be performed with respect to that variable and activates the variable automatically. If the code letter is omitted, then the variable's status remains the same as it was in the original molecule specification.

Geometry Specification & Modification Options

ModRedundant

Except for any case when it is combined with the GIC option (see below), the ModRedundant option will add, delete, or modify redundant internal coordinate definitions (including scan and constraint information) before performing the calculation. This option requires a separate input section following the geometry specification. AddRedundant is synonymous with ModRedundant.

This option may be used for job types other than optimizations. It may also be combined with NGeom, Check or AllCheck to retrieve and modify an internal coordinate definition from a checkpoint file.

When used with Check, or NGeom, two input sections will be read: the first contains the charge and multiplicity, and the second contains alterations to the retrieved internal coordinate definition. When combined with the AllCheck option, only the internal coordinate definition modifications input is needed.

Lines in a ModRedundant input section use the following syntax:

[*Type*] *N*1 [*N*2 [*N*3 [*N*4]]] [A | F]

[*Type*] *N*1 [*N*2 [*N*3 [*N*4]]] B

[*Type*] *N*1 [*N*2 [*N*3 [*N*4]]] K | R

[*Type*] *N*1 [*N*2 [*N*3 [*N*4]]] D

[*Type*] *N*1 [*N*2 [*N*3 [*N*4]]] H *diag-elem*

[*Type*] *N*1 [*N*2 [*N*3 [*N*4]]] S *nsteps stepsize*

*N*1, *N*2, *N*3, and *N*4 are atom numbers or wildcards (discussed below). Atom numbering begins at 1, and any dummy atoms are not counted.

The atom numbers are followed by a one-character code letter indicating the coordinate modification to be performed; the action code is sometimes followed by additional required parameters as indicated above. If no action code is included, the default action is to add the specified coordinate. These are the available action codes:

A	Activate the coordinate for optimization if it has been frozen.
F	Freeze the coordinate in the optimization.
B	Add the coordinate and build all related coordinates.
K	Remove the coordinate and kill all related coordinates containing this coordinate.
R	Remove the coordinate from the definition list (but not the related coordinates).
D	Calculate numerical second derivatives for the row and column of the initial Hessian for this coordinate.
H	Change the diagonal element for this coordinate in the initial Hessian to *diag-elem*.
S	Perform a relaxed potential energy surface scan. Increment the coordinate by *stepsize* a total of *nsteps* times, performing an optimization from each resulting starting geometry.

An asterisk (*) in the place of an atom number indicates a wildcard. Here are some examples of wildcard use:

*	All atoms specified by Cartesian coordinates.
* *	All defined bonds.
3 *	All defined bonds with atom 3.
* * *	All defined valence angles.
* 4 *	All defined valence angles around atom 4.
* * * *	All defined dihedral angles.
* 3 4 *	All defined dihedral angles around the bond connecting atoms 3 and 4.

By default, the coordinate type is determined from the number of atoms specified: Cartesian coordinates for 1 atom, bond stretch for 2 atoms, valence angle for 3 atoms, and dihedral angle for 4 atoms. Optionally, *type* can be used to designate these and additional coordinate types:

X	Cartesian coordinates.
B	Bond length.
A	Valence angle.
D	Dihedral angle.

L	Linear bend specified by three atoms (if *N*4 is -1) or by four atoms, where the fourth atom is used to determine the 2 orthogonal directions of the linear bend.

See the examples under the Opt keyword for illustrations of the use of ModRedundant.

NewDefinition

Generate a new set of redundant internal coordinates, replacing any that were in the checkpoint file. This option should not be used with GICs; use Geom=(Checkpoint,GIC) instead.

SkipAll

Do not generate any internal coordinates automatically. In the case of redundant internal coordinates based on the old algorithm available in Gaussian 09, all of the required coordinates must be explicitly specified in the ModRedundant input section. In the case of GIC-type internal coordinates based on the new GIC algorithm, you should use Geom=ReadAllGIC instead of this option.

SkipAng

Generates bonds but omits angles and dihedrals.

SkipDihedral

Suppresses the generation of dihedrals.

SkipHBond

Skips generation of hydrogen-bond coordinates.

KeepConstants

KeepConstants retains and NoKeepConstants discards information about frozen variables. The default is to retain them in symbolic form for the Berny algorithm and to discard them for older optimization algorithms (which don't understand them anyway).

NewRedundant

Rebuilds the redundant internal coordinates from the current Cartesian coordinates. If used with Geom=Modify, the new modifications are appended to any earlier Opt=ModRedundant input before the coordinate system is updated. This option should not be used with GICs; use GIC or AddGIC instead (as appropriate).

Redundant

Build an automatic set of redundant internal coordinates such as bonds, angles, and dihedrals from the current Cartesian coordinates or Z-Matrix values, using the old algorithm available in Gaussian 09.

Specifying Connectivity

Connectivity

Specify explicit atom bonding data via an additional input section (blank line-terminated) following the geometry specification and any modification to it. This option requires one line of input per atom, ordered the same as in the molecule specification, using the following syntax:

*N*1 *Order*1 [*N*2 *Order*2 $\cdots$]

where the '*N*'s are atoms to which the current atom is bonded, and the '*Order*'s are the bond order of the corresponding bond. For example, this input specifies that the current atom is bonded to atoms 4 and 5, with bond orders of **1.0** and **2.0** respectively:

```
8 4 1.0 5 2.0
```

A bond order of **0.1** indicates a bond which should be used in generating internal coordinates but which should not affect atom types or connectivity for molecular mechanics.

This input section is terminated by a blank line.

ModConnectivity

Modify the connectivity of the atoms in the molecule specification (or retrieved from the checkpoint file). This option requires an additional input section (blank line-terminated) following the geometry specification and any modification to it. Connectivity modifications use the following syntax:

$M\ N1\ Order1\ [N2\ Order2 \cdots]$

where M is the atom number, the 'N's are atoms to which that atom is bonded, and the '$Order$'s are the bond order of the corresponding bond. A bond order of **-1.0** removes a bond. For example, this input specifies that atom 8 is bonded to atoms 4 and 5, with bond orders of **1.0** and **2.0** respectively, and removes any bond to atom 9:

```
8 4 1.0 5 2.0 9 -1
```

This input section is terminated by a blank line.

ZMConnectivity

Read connectivity using the atom numbering specified in the Z-matrix (including dummy atoms). Bond orders involving dummy atoms are discarded.

IHarmonic=*n*

Add harmonic constraints to the initial structure with force constant *n*/1000000 Hartree/Bohr^2. InitialHarmonic is a synonym for this option.

ChkHarmonic=*n*

Add harmonic constraints to the initial structure saved on the checkpoint file with force constant *n*/1000000 Hartree/Bohr^2. CHarmonic is a synonym for this option.

ReadHarmonic=*n*

Add harmonic constraints to an additional structure read in the input stream (in the input orientation), with force constant *n*/1000000 Hartree/Bohr^2. RHarmonic is a synonym for this option.

ReadOptimize

Read an input section modifying which atoms are to be optimized. The atom list is specified in a separate input section (terminated by a blank line). By default, the atom list contains all atoms in the molecule, unless any atoms are designated as frozen within the molecule specification, in which case the initial atom list excludes them. If the structure is being read in from the checkpoint file, then the list of atoms to be optimized matches that in the checkpoint file. ReadOpt and RdOpt are synonyms for this option. ReadFreeze and RdFreeze are deprecated synonyms.

The input section uses the following format:

atoms=*list* [notatoms=*list*]

where each *list* is a comma- or space-separated list of atom numbers, atom number ranges, and/or atom types. Keywords are applied in succession. Here are some examples:

atoms=3-6,17 notatoms=5	*Adds atoms 3,4,6,17 to atom list. Removes 5 if present.*
atoms=3 C 18-30 notatoms=H	*Adds all C & non-H among atoms 3, 18-30.*
atoms=C N notatoms=5	*Adds all C and N atoms except atom 5.*
atoms=1-5 notatoms=H atoms=8-10	*Adds atoms 8-10 and non-hydrogens among atoms 1-5,*

Bare integers without a keyword are interpreted as atom numbers:

1,3,5 7 *Adds atoms 1, 3, 5 and 7.*

For ONIOM optimizations only, block and notblock can be similarly used to include/not include rigid blocks defined in ONIOM molecule specifications. If there are contradictions between atoms specified as atoms and within blocks – e.g., an atom is included within a block but excluded by atom type – Gaussian 09 generates an error.

You can start from an empty atom list by placing noatoms as the first item in the input section. For example, the following input optimizes all non-hydrogen atoms within atoms 1-100 and freezes all other atoms in the molecule:

```
noatoms atoms=1-100 notatoms=H
```

Atoms can also be specified by ONIOM layer via the [not]layer keywords, which accept these values: real for the real system, model for the model system in a 2-layer ONIOM, middle for the middle layer in a 3-layer ONIOM, and small for the model layer of a 3-layer ONIOM. Atoms may be similarly included/excluded by residue with residue and notresidue, which accept lists of residue names. Both keyword pairs function as shorthand forms for atom lists.

Micro

Set up redundant internal coordinates for ONIOM(MO:MM) microiterations, even if this is not an optimization.

Output-Related Options

Distance

This option requests printing of the atomic distance matrix (which is the default for molecules with fewer than 50 atoms). NoDistance suppresses this output.

CAngle

This option requests printing of interatomic angles using distance cutoffs to determine "bonded atoms". The default is not to print (NoAngle). Angle requests printing of the interatomic angles for Opt=Z-matrix (using the Z-matrix to determine which atoms are bonded). Only one of CAngle, Angle, and NoAngle may be specified.

CDihedral

This option requests printing of dihedral angles using distance cutoffs to determine "connectivity". The default is not to print (NoDihedral). Dihedral specifies printing of dihedral angles for Opt=Z-matrix (using connectivity information from the Z-matrix to decide which atoms are bonded). Only one of CDihedral, Dihedral, and NoDihedral may be specified.

PrintInputOrient

This option includes the table giving the Cartesian coordinates in the input orientation within the output file.

Print

This option turns on additional printing by the model builder facility.

Generalized Internal Coordinate (GIC)-Related Options

GIC

This option builds an automatic set of GIC-type internal coordinates instead of the default. The option NoGIC builds the internal coordinates of [461] as in Gaussian 09, and it is the default. The GIC-type coordinates generated by the GIC option are essentially the same as those generated by default.

AddGIC

This option adds, deletes, or modifies the GIC-type internal coordinate definitions generated automatically or retrieved from the checkpoint file. This option requires a separate GIC input section following the geometry specification. The syntax of the GIC input section is described in *GIC Info* tab. Note that Geom=(ModRedundant,GIC) is equivalent to Geom=AddGIC.

DefaultGIC

Make GICs the default (for use in a *Default.Route* file).

DefaultNoGIC

Make Peng internal coordinates the default (for use in a *Default.Route*, but this is the default anyway).

GICOld

Build the default set of redundant internal coordinates of [461] as in Gaussian 09, and then convert the coordinates into GIC-type internal coordinate definitions.

ReadAllGIC

Do not build any redundant internal coordinates by default. Instead, read the input stream for user-provided GIC definitions and create the coordinates. This option requires a separate GIC input section following the geometry specification. The syntax of the GIC input section is described in *GIC Info*.

Geometry Checking Options

Crowd

Crowd activates and NoCrowd turns off a check that aborts the job if atoms are closer than 0.5 Å. By default, the check is performed for every read-in geometry. It is not performed by default for later points of geometry optimizations, numerical frequencies, etc., when the geometry has been generated during the job. NoTest skips the test entirely.

Independent

Independent activates and NoIndependent turns off a check on the linear independence of the variables specified in a Z-matrix. This is done by default only if a full optimization is requested using the Berny algorithm (Opt=Z-matrix).

Other Options

Huge

Changes various defaults for huge (>20K atom) systems. Currently, this sets Geom=NoTest and Symm=None.

5.36.2 Related Keywords

Guess=Read, Opt=ModRedundant

5.36.3 GIC Info

The options that do not mention GIC and can be used with the Geom keyword should work as described – except for NewDefinition and NewRedundant. The latter should not be combined with any GIC-related option. In the case of GICs, you should use Geom=(Checkpoint,GIC) instead of Geom=NewDefinition, Geom=GIC instead of Geom=NewRedundant, and Geom=(Checkpoint,addGIC) instead of Geom=(Modify,NewRedundant).

This section discusses specifying generalized internal coordinates (GICs) in Gaussian input files. GICs have many potential uses: defining additional coordinates whose values are reported during geometry optimizations, freezing various structural parameters during the optimization of a molecular system, specifying parameters over which to perform a scan, defining constraints for geometry optimizations based on structural parameters or complex relationships between them, requesting calculation of parts of the Hessian, and other purposes.

The GIC input section is separated from the earlier input by a blank line. It has one or more lines containing coordinate definitions, expressions or standalone options. Here is a simple GIC input section for water illustrating some of the possible features:

`R(1,2)`	*Define a bond length coordinate for atoms 1 and 2*
`Bond2=R[1,3]`	*Define another bond length coordinate named Bond2*
`HOH(freeze)=A(2,1,3)`	*Define an optimization constraint: a bond angle coordinate named HOH (∠2-1-3)*

For an optimization, these coordinates will result in the bond angle remaining fixed at its initial value and the two bond distances being optimized.

The basic form of a coordinate is the following:

label(options)=expression

All of the components are optional. In the preceding examples, all components were present only in the third line. The first line contained only a coordinate expression, while the second line also contained a label without options. Note that options may also be placed following the expression:

```
HOH=A(2,1,3) Freeze
```

Labels are user-assigned identifiers for the coordinate. They are not case sensitive. Labels many contain letters and number, but must begin with a letter. If no label is specified, a generic one will be assigned by the program (e.g., R1, R2, A1, etc.). A parenthesized, comma-separated list of options can be included following the label if desired. Note that square brackets or braces may be substituted for parentheses anywhere in a coordinate definition.

Structural Parameters

Coordinates are defined by expressions. The simplest expressions simply identify a specific structural parameter within the molecule, using the following constructs. Note that an asterisk may be used as a wildcard for any atom number (see the examples).

R(*i*,*j*)

Define a bond coordinate between atoms *i* and *j*. B, Bond and Stretch are synonyms for R.

A(*i*,*j*,*k*)

Define a non-linear angle coordinate involving atoms *i*, *j* and *k* where the angle vertex is at atom *j*. Angle and Bend are synonyms for A.

D(*i*,*j*,*k*,*l*)

Define a dihedral angle between the plane containing atoms *i*, *j* and *k* and the plane containing atoms *j*, *k* and *l*. Dihedral and Torsion are synonyms for D.

L(*i*,*j*,*k*,*l*,*M*)

Define the linear bend coordinate involving atoms *i*, *j* and *k* where the angle vertex is at atom *j*. Linear and LinearBend are synonyms for L.

A linear bend definition has two components, indicated by *M* values of -1 and -2 for the first and second components, respectively (no other values are permitted). A linear bend is specified by defining its two orthogonal directions. These can be indicated in two ways:

◊ For a nonlinear molecule with more than 3 atoms, a fourth atom which does not form a linear angle with *i*, *j* and *k* in any combination can be used. In this case, *l* can be set to its atom number. For example, the following may be used to specify a linear bend involving atoms 1, 2 and 3 using atom 6 to determine the two orthogonal directions:

```
L(1,2,3,6,-1)
L(1,2,3,6,-2)
```

If *l* is set to -4, then the fourth atom will be determined automatically based on the molecular geometry.

◊ The other method is to project the linear bend onto one of the coordinate system's axial planes: the values of -1, -2 and -3 for *l* specify the YZ, XZ and XY planes (respectively). The value 0 may also be used to request that the appropriate plane be determined automatically:

```
L(1,2,3,0,-1)
L(1,2,3,0,-2)
```

X(*i*)

Define the x Cartesian coordinate for atom *i*. Cartesian(*i*,-1) and Cartesian(*i*,X) are synonyms, and Cartesian may be abbreviated as Cart.

Y(*i*)

Define the y Cartesian coordinate for atom *i*. Cartesian(*i*,-2) and Cartesian(*i*,Y) are synonyms, and Cartesian may be abbreviated as Cart.

Z(*i*)

Define the z Cartesian coordinate for atom *i*. Cartesian(*i*,-3) and Cartesian(*i*,Z) are synonyms, and Cartesian may be abbreviated as Cart.

XCntr(*atom-list*)
YCntr(*atom-list*)
ZCntr(*atom-list*)

Define x, y or z Cartesian coordinate for the geometric center (centroid) of a molecular fragment that contains specified atoms. The atom list is a comma-separated list of atom numbers and/or ranges. For example, XCntr(1,12-15,27) defines the x coordinate of the fragment containing atoms 1, 12, 13, 14, 15 and 27. If the atom list is omitted, it defaults to the entire molecule.

DotDiff(*i*,*j*,*k*,*l*)

Define the dot product $(a \cdot b)$ of the two Cartesian coordinate difference vectors a and b for atoms i, j, k and l determined as $a = (X_i - X_j, Y_i - Y_j, Z_i - Z_j)$ and $b = (X_k - X_l, Y_k - Y_l, Z_k - Z_l)$.

Compound Expressions

Complex expressions may be constructed by combining multiple items using one or more mathematical operations. The argument(s) A and B can be the labels of a previously defined coordinate, a valid GIC expression or even constants (integer or floating-point). The operation names are not case sensitive. The following operations are available:

◊ Square root: SQRT(A).

◊ Power of *e*: EXP(A) for e^A.

◊ Trigonometric functions: SIN(A), COS(A), TAN(A).

◊ Inverse cosine: ARCCOS(A).

◊ Addition: A+B

◊ Subtraction: A-B

◊ Multiplication: A*B

◊ Division: A/B

◊ Exponentiation: A***n* for A^n (n is an integer). The form A^*n* is also accepted.

Here are some simple examples which define symmetrized OH bonds in water:

```
R12(inactive)=B(1,2)
R13(inactive)=B(1,3)
RSym  = (R12 + R13)/SQRT(2)
RASym = [Bond(1,2) - Bond(1,3)]/SQRT(2)
```

The first two coordinates are set as inactive since they are intermediates not intended to be used in the optimization. Line 3 illustrates an expression using previously defined labels, while line 4 shows the use of literal expressions with operators. Note that the argument to the square root function is the constant 2.

Options

A comma separated list of options can follow the coordinate label, enclosed in parentheses. Alternatively, options may follow the expression, separated from it and from one another by spaces. All options are case insensitive.

For the purposes of geometry optimizations, a coordinate can be designated as:

◊ *Active*: The coordinate is part of the list of internal coordinates used in the geometry optimzation. In contrast, *Inactive* coordinates are not included in the set used for the geometry optimization. By default, active coordinates are unfrozen: allowed to change value (see the next bullet).

◊ *Frozen*: A coordinate whose value is held constant during the course of a geometry optimization. The values of active, unfrozen coordinates change during a geometry optimization. The frozen or unfrozen status of inactive coordinates is irrelevant during an optimization.

In the descriptions that follow, coordinates that “already exist” refers to previously-defined coordinates with the same label or the same value expression. Such coordinates may have been defined earlier in the input stream or retrieved from the checkpoint file from a previous job.

Active

If the specified coordinate does not already exist, build a new coordinate defined by the given expression, and flag it as active and unfrozen. If the coordinate was previously defined, then flag it as active and unfrozen (whatever its previous status). It is the default. Activate, Add and Build are synonyms for Active. May be abbreviated to A when specified following the expression.

Frozen

Build a coordinate defined by the expression if it does not exist, and flag the coordinate as active for geometry optimizations and freeze it at the current value.

Freeze is a synonym for Frozen. May be abbreviated to F when specified following the expression.

Inactive

If the coordiante does not already exist, build a new coordinate defined by the expression and flag it inactive. If the coordinate with the given label or for the given expression has been already built and flagged as active (frozen or unfrozen), then remove it from the geometry optimization by flagging it as inactive. Remove is a synonym for Inactive. May be abbreviated to R when specified following the expression.

Kill

Remove the coordinate from the list of internal coordinates used in geometry optimization along with any dependent coordinates by flagging all of them as inactive. The dependent coordinates include any coordinate that depends on the same atoms as the given coordinate. For example, R(1,5) Kill will result in removing the coordinate R(1,5) – the internuclear distance between atoms 1 and 5 – as well as the valence angles, dihedral angles and any other coordinate that depends on the Cartesian coordinates of atoms 1 and 5 in combination with other atoms in the molecule. RemoveAll is a synonym for Kill. May be abbreviated to K when specified following the expression.

PrintOnly

Include the initial value of the coordinate in the starting geometry in the Gaussian output file, and then flag it as inactive.

Modify

A label must be included in the coordinate specification for this option. It replaces the old coordinate with the specified label with the new expression, and flags the newly modified coordinate as active and unfrozen.

Diff

Calculate numerical second derivatives for the row and column of the initial Hessian corresponding to this coordinate. May be abbreviated to D when specified following the expression.

FC=*x*

Change the diagonal element for the given coordinate in the initial Hessian to x, a floating-point number in atomic units. ForceConstant is a synonym for FC.

Value=*x*

Set the initial value for the given internal coordinate to x, a floating point value. The units for the value are those of the Gaussian program, as defined by the Units keyword (Angstroms or degrees by default). The current Cartesian coordinates will be adjusted to match this value as closely as possible. This option should be used cautiously and sparingly. It is far easier and more reliable to set the initial molecular structure as desired in a graphical environment like GaussView.

StepSize=*x*,NSteps=*n*

These options are used to specify a relaxed potential energy surface scan in which the coordinate is incremented by x a total of n times, and a constrained optimization is perfromed from each resulting starting geometry. x should be a positive floating-point number in atomic units, n should be an integer >1. When these options follow the expression, the comma separating them should be replaced by a space.

Min=*min*,Max=*max*

This option is used in combination with Active, Freeze or Inactive. It adds, freezes or makes inactive the coordinate when its value satisfies the condition $min \leq value \leq max$. min and max are floating-point numbers in the units defined by the Units (Angstroms or degrees by default). If Min or Max is omitted, the condition becomes $value \leq max$ or $min \geq min$ respectively. When these options follow the expression, the comma should be replaced by a space.

action* OnlyIf *condition

action* IfNot *condition

These options provide conditional coordinate operations. They can only be placed following the expression defining the current coordinate. *Action* is one of Active, Freeze or Inactive. The *condition* is a label or expression for another coordinate. The specified action will be performed for the current coordinate if the coordinate referred to in *condition* is active for OnlyIf or inactive for IfNot. Note that the conditional test applies only to the *action* specified preceding the option and not to other options that may be present in the coordinate specification.

Standalone Options

The following options are independent of coordinate definitions and apply globally. They should be specified alone on their input line.

FreezeAll

Freeze all internal coordinate previously added as active.

UnFreezeAll

Unfreeze all internal coordinates previously added as active frozen.

RemoveAll

Remove/inactivate all internal coordinate previously added as active (frozen or unfrozen).

Atom *i action*

Apply the specified *action* to the Cartesian coordinates of atom *i*. If *i* is an asterisk, then the action applies to all atoms. *Action* is one of Active, Freeze, UnFreeze, Remove (make inactive), RemoveAll and XYZOnly. These options are as defined above; XYZOnly says to remove any internal coordinates that depend on atom *i* but to add/retain the coordinates of that atom. The default *action* is Active.

Examples

The following example manipulates some automatically-generated coordinates, defines some new ones, and then uses wildcards to remove coordinates related to specific atoms:

`R(5,9) freeze`	*Freeze bond distance R(5,9).*
`R(8,9)`	*Add a new active coordinate R(8,9) with a default label.*
`Ang189 = A(1,8,9)`	*Add a new active coordinate A(1,8,9) labeled Ang189.*
`R10(remove)`	*Remove a coordinate labeled R10.*
`Dih6123(remove) = D(6,1,2,3)`	*If D(6,1,2,3) exists, then remove the coordinate.*
`Dis79(freeze) = R(7,9)`	*Freeze the coordinate R(7,9): if it is new, then label it Dis79; if it already exists, retain the old label.*
`G1 = (R16+R19)*0.529177`	*Add a new coordinate labeled G1.*
`Ang189a(modify)=cos(g2)*57.29577951`	*Change the definition of coordinate Ang189a.*
`R(11,*) remove`	*Remove distances between atom 11 and any other atom.*
`D(*,1,17,*) remove`	*Remove any dihedral built around the 1-17 bond.*

Note that if a specified coordinate already exists, then an entry adding it will result in an error (e.g., lines 1-3 above).

The following example first defines the centroids of two fragments. Then, it defines the interfragment distance as an optimization coordinate:

Define the center of Fragment 1, but don't include it in the optimization.

```
XC1(Inactive)=XCntr(1-10)
YC1(Inactive)=YCntr(1-10)
ZC1(Inactive)=ZCntr(1-10)
```

Define the center of Fragment 2, but don't include it in the optimization.

```
XC2(Inactive)=XCntr(11-21)
YC2(Inactive)=YCntr(11-21)
ZC2(Inactive)=ZCntr(11-21)
```

Define the distance F1-F2 and include it in the optimization. Its value will be reported in Å:

```
F1F2=sqrt[(XC1-XC2)^2+(YC1-YC2)^2+(ZC1-ZC2)^2]*0.529177
```

The following example requests a relaxed PES scan over the same coordinate:

```
F1F2(NSteps=10,StepSize=0.2)
```

The following example removes an angle coordinate generated by default if ≥179.9^o, substituting a linear bend:

```
 A(1,2,3) Remove Min=179.9              Remove angle coordinate if too large.
 L(1,2,3,0,-1) Add IfNot A(1,2,3)       Add linear bend only if the angle coordinate not active.
 L(1,2,3,0,-2) Add IfNot A(1,2,3)
```

The following example removes an angle coordinate if it is ≤ the specified value, setting the corresponding force constant is set to 0.2 au. The latter applies whenever it is needed: as the initial force constant and the force constant to use should be variable be reactivated. The second line specifies the force constant for a bond coordinate:

```
A(1,2,3) Remove Min=3.139847 ForceConstant=0.2
R(1,2) FC=0.5
```

The following example sets the force constants for various coordinates. It also inactivates bond angle coordinates ≥ 179.8^o:

```
R(1,*) FC=0.8
D(*,4,5,*) FC=0.4
A(*,1,*) FC=0.5
A(*,*,*) R Min=179.8
```

Limitations of GICs in the Current Implementation

In the current implementation, GICs can be successfully used for many purposes including optimization constraints and PES scans. However, there are potential problems with active composite coordinates including multiple dihedral angles. In general, coordinates comprised of combinations of bond distances and bond angles should behave well. Simple dihedral angles are also welll supported. Complex expressions involving multiple dihedral angles are acceptable for frozen coordinates and for PES scans. However, they should be avoided as active optimization coordinates.

In a non-GIC optimization, or one using GICs with only regular dihedrals, then the program is careful about the periodicity of these coordinates. For example, in deciding whether a step in the geometry is too big and needs to be scaled back, it recognizes that a change in value from 1 degree to 359 degrees is really a change of -2 degrees rather than 358 degrees. Similarly, in numerically differentiating the forces in order to update the Hessian, displacements between geometries in internal coordinates are needed, and the periodicity is accounted for. A problem can arise when a GIC is a combination of parts for which such periodicity is important, typically, combinations of multiple dihedral angles. For example, consider these GICs:

```
D1 = D(1,2,3,4)
D2 = D(5,6,7,8)
V1 = D1 + 2*D2
```

D1 and D2 are dihedral angles, but they are intermediates and are not used as variables in the optimization. Their periodicity is not currently recognized in the composite coordinate V1. Suppose they have values of 1 and 2 degrees at one geometry and 1 and 359 degress at the next. The change in the optimization variable V1 should be 0 + 2*(-3) = -6 degrees, but it is actually 0 + 2*(357) = 714 degrees, which looks like an enormous change. This will result in the optimization algorithm performing very poorly. V1 isn't a simple periodic function; it is necessary to apply periodicity to its component parts as it is computed, which is not done in the current GIC implementation.

GIC Units in Gaussian Output

The values of the GICs defined as pure distances and angles (including valence angles, linear bends and dihedral angles/torsions) are computed from the Cartesian coordinates in atomic units (Bohrs) and stored internally in Bohrs and radians. However, for the user's convenience, they are expressed as usual in Angstroms and degrees in the Gaussian output. In the case of a generic GIC (i.e., when the GIC is not a pure Cartesian coordinate, bond distance or angle), the GIC value is computed as a function of Cartesian coordinates and bond distances in Bohrs and angles in radians, combined with optional constants in user-defined units. Such generic GIC values (labeled as GIC) are computed, stored and output in these same units: i.e., if the GIC is a combination of bonds or a combination of valence angles, then the arbitrary units become Bohrs for the bonds and radians for the angles.

Use of ModRedundant Format Input

Modifications to the GICs can be read in using the ModRedundant format from the current internal coordinate algorithm. However, the old format is only available with the GICs that include only pure bond distances, bond angles or torsion angles. In addition, the old format and the new GIC format described above cannot be mixed together in the same input section.

5.37 GFInput

The GFInput (for Gaussian Function Input) output generation keyword causes the current basis set to be printed in a form suitable for use as general basis set input and can thus be used in adding to or modifying standard basis sets. By default, both the basis set and any fitting set are printed.

5.37.1 Options

JNormalization

Print the density basis using Coulomb normalization. This is the default for the fitting set.

AONormalization

Print the density fitting basis set using AO (atomic overlap) normalization. The basis set is always printed using AO normalization.

RawNormalization

Print the density fitting set basis unnormalized.

5.37.2 Related Keywords

Gen, GFPrint

5.38 GFPrint

This output generation keyword prints the current basis set and density fitting basis set in tabular form. The variant GFOldPrint keyword prints the basis set information in the Gaussian format.

5.38.1 Related Keywords

Gen, GFInput

5.39 Gn Methods

These method keywords request the following methods for computing very accurate energies:

◊ Gaussian-1 (G1) [462, 463]

◊ Gaussian-2 (G2) [464]

◊ Gaussian-3 (G3) [465]

◊ Gaussian-4 (G4) [466]

◊ G2MP2 requests the modified version of G2 known as G2(MP2), which uses MP2 instead of MP4 for the basis set extension corrections [467] and is nearly as accurate as the full G2 method at substantially reduced computational cost.

◊ G3MP2 requests the similarly modified G3(MP2) method [468].

◊ The G3 variants using B3LYP structures and frequencies [469] are requested with the G3B3 and G3MP2B3 keywords.

◊ G4 and G4MP2 request the fourth-generation methods [466, 470].

All of these methods are complex energy computations involving several pre-defined calculations on the specified molecular system. All of the distinct steps are performed automatically when one of these keywords is specified, and the final computed energy value is displayed in the output. No basis set keyword should be specified with these keywords.

Users should generally consider other high accuracy methods before selecting one of these. CBS-QB3 is equally accurate and significantly faster, while W1U is more accurate (but slower).

Either of the Opt=Maxcyc=*n*, QCISD=Maxcyc=*n*, or CCSD=Maxcyc=*n* keywords may be used in conjunction with any of the these keywords to specify the maximum number of optimization, QCISD, or CCSD cycles, respectively.

5.39.1 Options

SP

Do only a single-point energy evaluation using the specified compound model chemistry. No zero-point or thermal energies are included.

NoOpt

Perform the frequencies and single-point energy calculation for the specified model chemistry at the input geometry. Freq=TProjected is implied. This option is not meaningful or accepted for methods such as G1, which use different geometries for the frequencies and the single-point steps. StartFreq is a synonym for NoOpt.

ReadIsotopes

This option allows you to specify alternatives to the default temperature, pressure, frequency scale factor and/or isotopes – 298.15 K, 1 atmosphere, no scaling, and the most abundant isotopes (respectively). It is useful when you want to rerun an analysis using different parameters from the data in a checkpoint file. Be aware, however, that all of these can be specified in the route section (Temperature, Pressure and Scale keywords) and molecule specification (the Iso parameter), as in this example:

```
#T Method/6-31G(d) JobType Temperature=300.0 ···

...


0 1
C(Iso=13)
...
```

ReadIsotopes input has the following format:

temp pressure [scale]	*Values must be real numbers.*
isotope mass for atom 1	
isotope mass for atom 2	
...	
isotope mass for atom n	

Where *temp*, *pressure*, and *scale* are the desired temperature, pressure, and an optional scale factor for frequency data when used for thermochemical analysis (the default is unscaled). The remaining lines hold the isotope masses for the various atoms in the molecule, arranged in the same order as they appeared in the molecule specification section. If integers are used to specify the atomic masses, the program will automatically use the corresponding actual exact isotopic mass (e.g., 18 specifies ^{18}O, and Gaussian uses the value 17.99916).

Restart

Resume a partially-completed calculation from its checkpoint file. When used in combination with ReadIso, this option allows for the rapid computation of the energy using different thermochemistry parameters and/or isotope selections.

5.39.2 Examples

Calculation Summary Output. After all of the output for the component job steps, Gaussian prints a table of results for these methods. Here is the output from a G2 calculation:

```
Temperature=        298.150000   Pressure=             1.000000
E(ZPE)=                .020511   E(Thermal)=            .023346
E(QCISD(T))=        -76.276078   E(Empiric)=           -.024560
DE(Plus)=             -.010827   DE(2DF)=              -.037385
G1(0 K)=            -76.328339   G1 Energy=          -76.325503
G1 Enthalpy=        -76.324559   G1 Free Energy=     -76.303182
E(Delta-G2)=          -.008275   E(G2-Empiric)=         .004560
G2(0 K)=            -76.332054   G2 Energy=          -76.329219
G2 Enthalpy=        -76.328274   G2 Free Energy=     -76.306897
```

The temperature and pressure appear first, followed by the various components used to compute the G2 energy. The output concludes with the G2 energy at 0 K and at the specified temperature (the latter includes a full thermal correction rather than just the zero-point energy correction), and (in the final output line) the G2 theory predictions for the enthalpy and Gibbs free energy (both computed using the thermal-corrected G2 energy). (Note that the same quantities predicted at the G1 level are also printed in this summary section.)

The energy labels thus have the following meanings (G2 is used as an example):

G2 (0 K)	Zero-point-corrected electronic energy: $E_0 = E_{elec} + ZPE$
G2 Energy	Thermal-corrected energy: $E = E_0 + E_{trans} + E_{rot} + E_{vib}$
G2 Enthalpy	Enthalpy computed using the G2 predicted energy: $H = E + RT$
G2 Free Energy	Gibbs Free Energy computed using the G2 predicted energy: $G = H - TS$

Rerunning the Calculation at a Different Temperature. The following two-step job illustrates the method for running a second (very rapid) G2 calculation at a different temperature. This job computes the G2 energy at 298.15 K and then again at 300 K:

```
%Chk=formald
# G2 Test

G2 on formaldehyde

0 1
molecule specification

--Link1--
%Chk=formald
%NoSave
# G2(Restart,ReadIso) Geom=Check

300.0 1.0
```

isotope specifications

5.40 Guess

The Guess keyword controls the initial guess for the Hartree-Fock wavefunction. This keyword is not meaningful without an option. By default, a Harris guess is used.

5.40.1 Options

Algorithm-Related Options

Harris

Diagonalize the Harris functional [471] for the initial guess. This is the default for all HF and DFT calculations.

Huckel

Requests that a Huckel guess be generated. This is the default for CNDO, INDO, MNDO, and MINDO3. The Huckel guess should be considered for PM6 calculations involving many second-row atoms.

RdScale

Read in the scale factor on atomic hardnesses used in iterative extended Huckel. The default is 7.0 times the QEq value.

OldHuckel

Use the old Huckel guess (pre-Gaussian 03) instead of CNDO or the updated Huckel.

INDO

Use the Gaussian 98 default guess: INDO for first-row systems, CNDO for second-row, and Huckel for third-row and beyond.

AM1

Do an AM1 calculation for the initial guess (currently only works with sparse matrix code). Guess=(AM1, Always) causes later steps in a geometry optimization to generate a new guess at each point and compare the energies with the density from the old point and the new guess and take the best one.

Core

Requests that the core Hamiltonian be diagonalized to form the initial guess. This is the default for AM1, PM3, PM3MM, PM6, and PDDG.

TightConvergence

Change the defaults to 10^{-10} convergence on SCF and 10^{-8} for CIS, TD and CC. CPHF convergence is not changed.

Orbital-Related Options

Permute

Read in a permutation of orbitals in the initial guess. The numbers of the generated guess orbitals are given in the order in which they should be used in the SCF. Ranges (e.g. 7–12) can be used, and all orbitals not listed are put in after the listed orbitals in their original order. Separate permutation lists for α and β orbitals must be specified (each list separated by a blank line) for open shell systems.

Alter

Indicates that the orbitals selected for occupation in the Hartree-Fock wavefunction should not be those of lowest energy. Normally, the occupied orbitals are selected as those with lowest eigenvalues for the one-electron Hamiltonian used in the initial guess programs. The alteration sections consist of a set of transpositions indicating that one of these occupied orbitals is to be replaced by one of the other (virtual) orbitals. Each such transposition is on a separate line and has two integers N_1 and N_2 (free format, separated by spaces or a comma as usual) indicating that orbital N_1 is to be swapped with orbital N_2. The list of orbital transpositions is terminated by the blank line at the end of the input section.

For UHF calculations, two such orbital alteration sections are required, the first specifying transpositions of α orbitals and the second specifying transpositions of β orbitals. Both sections are always required. Thus, even if only α transpositions are needed, the β section is required even though it is empty (and vice-versa). The second blank line to indicate an empty β section must be included.

Mix

Requests that the HOMO and LUMO be mixed so as to destroy $\alpha - \beta$ and spatial symmetries. This is useful in producing UHF wavefunctions for singlet states. Mixing orbitals is only done by default when generating a complex initial guess. NoMix says to not mix orbitals.

DensityMix[=*N*]

Whether to mix occupied and virtual orbital contributions in forming the initial guess density. N defaults to -3 (use Huckel eigenvalues to decide which orbitals to mix).

CopyChk

Copy the MOs and density unedited, unprojected and unchecked from the checkpoint file for the guess. Useful when importing from matrix element files.

Biorthogonalize

For an unrestricted guess, biorthogonalize the α and β MOs to maximally pair electrons of opposite spin. Done automatically when a UHF wavefunction is read-in as a guess for ROHF. This option is also useful in the combination of Guess=(BiOrth,Read,Only,Save) to replace the canonical UHF orbitals with ones which have alpha and beta orbitals matched up as much as possible for viewing in GaussView or other visualization packages.

NaturalOrbitals

Include natural orbitals in the checkpoint file. This must be accomplished via a separate job step specifying this option as well as Only and Save. See the discussion of the Population keyword for details.

Procedural Options

Only

Guess=Only functions as a calculation type keyword and requests that the calculation terminate once the initial guess is computed and printed. Note that the amount of orbital information that is printed is controlled by the Pop keyword. Guess=Only may not be used with the MOPAC-based semi-empirical methods (INDO, CNDO, MNDO, MINDO3).

This option is useful in preliminary runs to check if configuration alteration is necessary. For example, Guess=Only may be specified with CASSCF in order to obtain information on the number of CI configurations in the CAS active space (as well as the initial orbitals).

Guess=(Only,Read) may also be used to produce population and other post-calculation analyses from the data in a checkpoint file. For example, these options alone will produce a population analysis using the wavefunction in the checkpoint file. Guess(Only,Read) Prop will cause electrostatic properties to be calculated using the wavefunction in the checkpoint file.

Always

Requests that a new initial guess be generated at each point of an optimization. By default, the SCF results from the last point are used for the guess at the next point.

Fragment=*N*

Generate a guess built from fragment guesses or SCF solutions. The guess is saved on the checkpoint file regardless of if it is used in the current calculation, allowing it to be used in subsequent ones. This option is usually combined with Guess=Only to generate a guess from fragment guess orbitals (otherwise a full SCF calculation is performed for each fragment). Assignment of atoms to fragments and fragment charges and multiplicities are specified as described in Molecule Specifications, except that a negative spin multiplicity means that the unpaired orbitals for the fragment are to become β spin orbitals in the combined set.

Local

Requests that orbitals be localized using the Boys method [184]. Occupied and virtual orbitals are localized separately, and the irreducible representations (after possible merging using LowSymm or NoSymm) are not mixed. Localized orbital analysis of a converged SCF wavefunction may then be done using a second job step, which includes Guess(Read,Local,Only) and Pop=Full in its route section.

Sparse

Perform a sparse SE calculation for the initial guess. This option may be useful for very, very large HF or DFT calculations using the sparse matrix facility.

Extra

Do an extra, new initial guess when reading orbitals from the RWF (i.e., during geometry optimizations). By default, this is done if the default Harris guess is allowed, no alteration of configuration was requested, and the optimization did not take a small step as flagged by variable 4 in `ILSW`. Use NoExtra to disable this feature.

Fock

Reuse Fock matrices rather than orbitals when reading from previous results from the read-write or checkpoint files. This is the default for periodic boundary conditions calculations if Guess=Alter is not specified. NoFock disables this behavior, and it is the default for non-PBC calculations.

Guess Reading/Saving Options

Read

Requests that the initial guess be read from the checkpoint file (Guess=Read is often specified along with Geom=Checkpoint). This option may be combined with Alter, in which case the orbitals are read from the checkpoint file, projected onto the current basis set, and then the specified alterations are made. Checkpoint is a synonym for Read. The TCheck option says to attempt to read a guess from the checkpoint file but to generate a new one if necessary.

Alpha

Use alpha orbitals for both alpha and beta guess during Guess=Read.

Translate

Translate requests that the coordinates of the atoms used to produce a guess, which is read in, be translated to the current atomic coordinates. This is the default. It may fail in unusual cases, such as when a wavefunction is used as a guess for a system with a different stoichiometry, in which case Guess=NoTranslate should be specified.

Cards

Specifies that after the initial guess is generated, some or all of the orbitals will be replaced with ones read from the input stream. This option can be used to read a complete initial guess from the input stream by replacing every orbital. The replacement orbitals are placed in the input section following the guess alteration commands, if any. For UHF, there are separate α and β replacement orbital input sections.

The replacement orbitals input section (the α replacement orbitals section for UHF) begins with a line specifying the Fortran format with which to read the replacement orbital input, enclosed in parentheses. For example: (`4E20.8`). The remainder of the section contains one or more instances of the following:

IVec	*Orbital to replace (0=end, -1=replace all orbitals in order).*
(A(I,IVec),I=1,N)	*New orbital in the format specified in the first line.*

The format for the line containing *IVec* is Fortran I5. The β orbital replacement section for UHF calculations differs only in that it omits the initial format specification line. See the examples section for sample replacement orbital input.

Input

Read a line from the input file containing the name of a checkpoint file. Instead of a filename, you can specify one of the following keywords:

◊ generate says to generate a guess as usual.

◊ read and chk say to retrieve data from the job's checkpoint file. They are thus equivalent to Guess=Read, and use the checkpoint file specified to %Chk or %OldChk as usual.

◊ none says to skip generating a guess at all.

◊ dummy generates a dummy guess that does not require any work.

See the examples section for information about using this option with ONIOM.

Restart

Start a new SCF calculation using the restart data saved on the checkpoint file from a job with a different geometry and/or basis set. (The option name is a bit of a misnomer.)

Save

Save the generated initial guess back into the checkpoint file at the conclusion of a Guess=Only run. This option is useful for saving localized orbitals.

Print

Print the initial guess.

Symmetry-Related Options

LowSymm

Requests that irreducible representations of the molecular point group be combined in the symmetry information used in the N^3 steps in the SCF to allow lowered symmetry of the wavefunction. This enables the orbitals (and possibly but not necessarily the total wavefunction) to have lower symmetry than the full molecular point group. This option is available only for GVB calculations, where it is often necessary for calculations on symmetric systems (see the discussion of the GVB keyword below for an example using this option).

The option expects a single line of input (in the format `16I2`) giving the *numbers* of the irreducible representations to combine, with the new groups separated by 0; the list itself must be terminated by a 9. The numbers correspond to the order in which the representations are listed by Link 301 in the output file (see the examples subsection below).

Since this input section is always exactly one line long, it is not terminated by a blank line. Note that irreducible representations are combined before orbital localization is done and that localized orbitals retain whatever symmetry is kept. Guess=NoSymm removes all orbital symmetry constraints without reading any input.

NoSymm

Requests that all orbital symmetry constraints be lifted. Synonymous with SCF=NoSymm and Symm=NoSCF.

ForceAbelianSymmetry

Force the initial guess orbitals to transform according to irreducible representations of the Abelian point group. NoForceAbelianSymmetry is the default.

Valid Option Combinations

Only reasonable combinations are valid. For example Guess=(Always,Alter) and Guess=(Read,Alter) work as expected (in the former case, alterations are read once and the same interchanges are applied at each geometry). Conversely, Guess=(Always,Read) is contradictory and will lead to unpredictable results. Refer to the input sections order table at the beginning of this chapter to determine the ordering of the input sections for combinations of options like Guess=(Cards,Alter).

5.40.2 Restrictions

Guess=Only may not be used with the MOPAC-based semi-empirical methods: INDO, CNDO, MNDO, and MINDO3.

5.40.3 Related Keywords

Geom, Pop

5.40.4 Examples

Transposing Two Orbitals with Guess=Alter

This example finds the UHF/STO-3G structure of the $^2A_1$ excited state of the amino radical. First, a Guess=Only calculation is run to determine whether any alter (reordering) instructions are needed to obtain the desired electronic state. The HF/STO-3G theoretical model is used by default:

```
# Guess=Only Test

Amino radical test of initial guess

0 2
n
h 1 nh
h 1 nh 2 hnh
```

```
nh 1.03
hnh 120.0
```

Here is the orbital symmetry summary output from the job, which comes immediately before the population analysis in the output:

```
Initial guess orbital symmetries:
 Alpha Orbitals:
       Occupied  (A1) (A1) (B2) (A1) (B1)
       Virtual   (A1) (B2)
 Beta  Orbitals:
       Occupied  (A1) (A1) (B2) (A1)
       Virtual   (B1) (A1) (B2)
 The electronic state of the initial guess is 2-B1.
 Initial guess <Sx>= 0.0000 <Sy>= 0.0000 <Sz>= 0.5000 <S**2>= 0.7500 S= 0.5000
```

Since a doublet state is involved, α and β orbitals are given separately. From the orbital symmetries, the electron configuration in the initial guess is $a_1^2 a_1^2 b_2^2 a_1^2 b_1^1$, yielding a ${}^2B_1$ wavefunction. This is indeed the ground state of NH_2. The expectation value of S^2 for the unrestricted initial guess is printed, in this case, that corresponding to a pure doublet: 0.75.

Since we want to model the ${}^2A_1$ excited state of the amino radical, we will need to alter this initial orbital configuration: a β electron must be moved from orbital 4 to orbital 5 (the electron configuration is then $a_1^2 a_1^2 b_2^2 b_1^2 a_1^1$).

Guess=Alter may be used to accomplish this. Here is the input for the geometry optimization:

```
# UHF/6-31G(d) Opt Guess=Alter

Amino radical: HF/6-31G(d)
Structure of 2-A1 state

0 2
 n
h 1 nh
h 1 nh 2 hnh
     Variables:
nh 1.03
hnh 120.0
         Blank line ends the molecule specification section.
         Blank line ends the alpha section (empty in this case).
4 5      Transpose orbitals 4 and 5.
         End of the beta alteration section.
```

Note that an extra blank line – line 14 – is necessary to indicate an empty α alteration section. The final two lines then constitute the β alteration section.

The initial guess program prints a list of orbitals that were interchanged as a result of the Alter option:

```
 Harris functional with IExCor=  205 diagonalized for initial guess.
```

```
...
No Alpha orbitals switched.
Pairs of Beta  orbitals switched:
   4   5
Initial guess orbital symmetries:
Alpha Orbitals:
      Occupied  (A1) (A1) (B2) (A1) (B1)
      Virtual   (A1) (B2) (B1) (A1) (B2) (A1) (B2) (A1) (A2) (A1)
                (B1) (A1) (B2) (A1)
Beta  Orbitals:
      Occupied  (A1) (A1) (B2) (B1)
      Virtual   (A1) (A1) (B2) (B1) (A1) (B2) (A1) (B2) (A1) (A2)
                (A1) (B1) (A1) (B2) (A1)
The electronic state of the initial guess is 2-A1.
Initial guess <Sx>= 0.0000 <Sy>= 0.0000 <Sz>= 0.5000 <S**2>= 0.7500 S= 0.5000
```

The SCF calculation follows, and the energy and S^2 eigenvalue for the UHF wavefunction are printed. The S^2 value which results if contamination of the wavefunction from the next possible spin multiplicity (quartets for doublets, quintets for triplets, etc.) is removed is also printed:

```
 SCF Done:  E(UHF) =  -55.4915172451     A.U. after   12 cycles
            Convg  =    0.2693D-08             -V/T =  2.0038
<Sx>= 0.0000 <Sy>= 0.0000 <Sz>= 0.5000 <S**2>= 0.7534 S= 0.5017
<L.S>= 0.000000000000E+00
Annihilation of the first spin contaminant:
S**2 before annihilation     0.7534,    after     0.7500
```

Although this calculation does in fact converge correctly to ${}^2A_1$ state, it sometimes happens that the order of orbital symmetries switches during the course of the SCF iterations. If the orbital symmetries of the final wavefunction are different from those in the initial guess (whether or not you are using Guess=Alter), we recommend using the direct minimization routine, specified with the SCF=QC or SCF=DM keywords, which usually holds symmetry from one iteration to the next.

Reordering Orbitals with Guess=Permute

This option is often is the easiest way to perform a complex modification of the initial guess, as in this example:

```
# CASSCF/6-31G(d,p) Opt Guess=Permute Pop=Reg

CAS job

0 1
```
molecule specification

```
1-60 65 63 64 66 68 67 61-62 69
```
Specify new orbital ordering.

Here, we have rearranged orbitals 61-68. Listing the final orbital (69) is not really necessary, but it helps to make the input easier to understand for humans.

Specifying Calculation-Specific Guesses for ONIOM with Guess=Input

The Guess=Input is useful for ONIOM jobs where the SCF is hard to converge for some of the sub-calculations at the initial geometry. Convergence can be accomplished using a procedure like the following:

Step 1. Use ONIOM=OnlyInput to print out input files for the individual calculations.

Step 2. Converge the SCF for the problem step separately (for example, using Stable=Opt).

Step 3. After verifying that the correct state has been found, run an ONIOM calculation with Guess=Input to retrieve the correct initial guesses for the individual states. In the ONIOM input file, you will then have a line for the guess for each calculation (i.e., 3 lines for a 2-layer MO:MO), where each line can be a checkpoint file name or read (for Guess=Read from the regular checkpoint file) or generate (generate a new initial guess), as appropriate. For example, if only the high level calculation on the model system was problematic, the input would be:

generate	*Low level calculation on the real system.*
	Blank line.
guess_for_high.chk	*High level calculation on the model system.*
	Blank line.
generate	*Low level calculation on the model system.*

Reading in Orbitals with Guess=Cards

Some or all of the orbitals may be replaced, after the initial guess is generated, using Guess=Cards. Here is some sample input for this option, which replaces orbitals 1 and 4 (note that the format for the third and following lines is specified in line 1):

```
(3E20.8)
    1
    0.5809834509E+00    0.4612416518E+00   -0.6437319952E-04
    0.1724432549E-02    0.1282235396E-14    0.5417658499E-13
    0.1639966912E-02   -0.9146282229E-15   -0.6407549694E-13
   -0.4538843604E-03    0.6038992958E-04   -0.1131035485E-03
    0.6038992969E-04   -0.1131035471E-03
    4
    0.7700779642E-13    0.1240395916E-12   -0.3110890228E-12
   -0.4479190461E-12   -0.1478805861E-13    0.5807753928E+00
    0.6441113412E-12   -0.3119296374E-14    0.1554735923E+00
   -0.1190754528E-11    0.2567325943E+00    0.1459733219E+00
   -0.2567325943E+00   -0.1459733219E+00
    0
```

An orbital number of zero ends the replacement orbital input.

Antiferromagnetic Coupling

Here is an example of a fragment guess job. The first step generates guesses for each one of the fragments and then combines them together into a guess for the SCF on the full molecule. The second step reads in that guess for a subsequent calculation. The charge and multiplicity line has the overall molecular charge and multiplicity first, followed by those for each fragment.

This example is Fe_2S_2 + 4 S-R ligands, where R is a phenyl group. It is a singlet with an overall charge of -2. The guess does each bare sulfur as $S^{(2-)}$ closed-shell (fragments 2 and 4), the two irons as $Fe^{(3+)}$ sextets antiferromagnetically coupled (fragments 1 and 3, with 1 being alpha-spin and 3 beta-spin), and each of the S-R fragments (5-8) as closed-shell singlet anions.

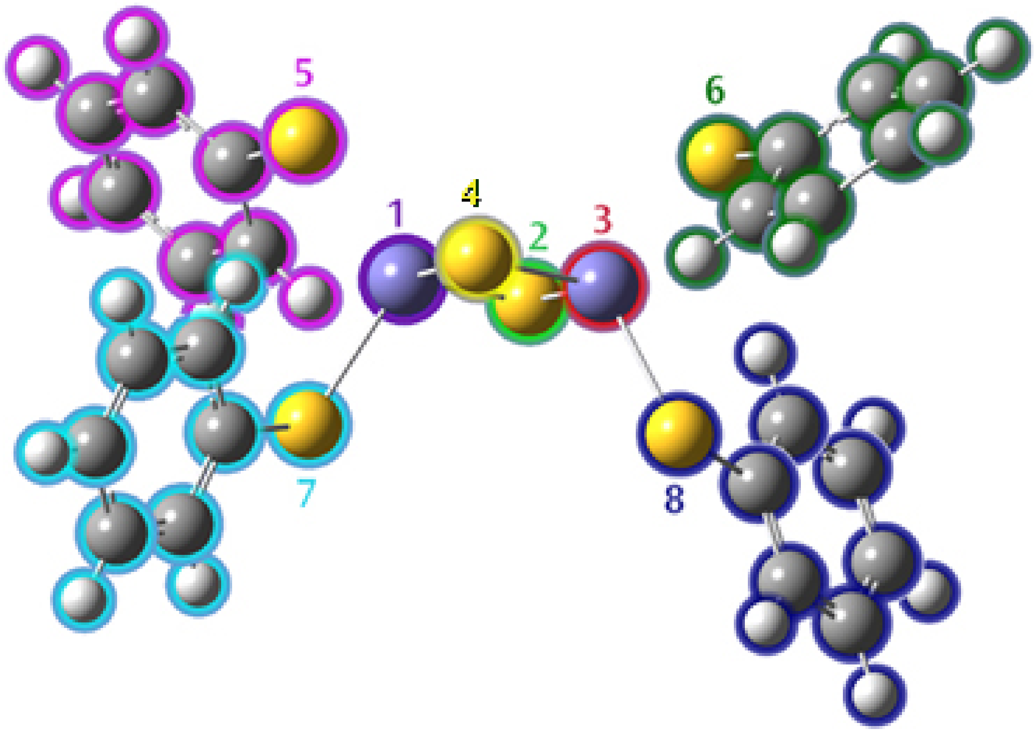


Figure 5.1: Fragment Assignments

```
%chk=FragGuess
%mem=64mw
#P UBP86/6-311G(d) Guess=(Fragment=8,Only) Pop=None

Fe2S2 cluster with phenylthiolates.
Step 1: Generate fragment guess

-2,1 3,6 -2,1 3,-6 -2,1  -1,1  -1,1  -1,1  -1,1
H(Fragment=7) 23.5010  2.2873  8.5744
S(Fragment=2) 14.8495  1.1490  7.0431
Fe(Fragment=3) 17.0430  1.0091  7.0068
S(Fragment=4) 17.4565 -1.1490  7.0431
S(Fragment=5) 14.3762 -2.1581  8.7983
C(Fragment=5) 12.5993 -2.1848  8.6878
H(Fragment=5) 12.3743 -3.6513 10.1678
C(Fragment=5) 10.4994 -3.1122  9.4309
H(Fragment=5)  9.9929 -3.7579 10.0022
C(Fragment=5)  9.8049 -2.2791  8.5639
H(Fragment=5)  8.8050 -2.2873  8.5744
C(Fragment=5) 10.4833 -1.4146  7.6615
H(Fragment=5)  9.9730 -0.8525  7.0106
```

```
S(Fragment=8) 14.3794 -1.8091  5.0446
C(Fragment=5) 11.9048 -1.3675  7.7057
H(Fragment=5) 12.4158 -0.7843  7.0743
C(Fragment=6) 17.2999  3.4265  4.6624
C(Fragment=6) 16.6376  4.1967  5.6090
H(Fragment=6) 16.5022  3.8494  6.5369
C(Fragment=6) 16.1530  5.4856  5.2463
H(Fragment=6) 15.6665  6.0472  5.9155
C(Fragment=6) 16.3468  5.9257  4.0431
H(Fragment=6) 16.0236  6.8408  3.8020
C(Fragment=6) 17.0091  5.1398  3.0522
H(Fragment=6) 17.1330  5.4944  2.1254
C(Fragment=6) 17.4775  3.8823  3.3884
H(Fragment=6) 17.9400  3.3149  2.7071
S(Fragment=7) 17.9298  2.1581  8.7983
C(Fragment=7) 19.7067  2.1848  8.6878
C(Fragment=7) 20.4174  3.0650  9.5194
H(Fragment=7) 19.9317  3.6513 10.1678
C(Fragment=7) 21.8066  3.1122  9.4309
H(Fragment=7) 22.3132  3.7579 10.0022
C(Fragment=7) 22.5011  2.2791  8.5639
C(Fragment=7) 21.8227  1.4146  7.6615
H(Fragment=7) 22.3330  0.8525  7.0106
C(Fragment=7) 20.4012  1.3675  7.7057
H(Fragment=7) 19.8902  0.7843  7.0743
C(Fragment=8) 15.0061 -3.4265  4.6624
Fe(Fragment=1) 15.2630 -1.0091  7.0068
C(Fragment=8) 15.6684 -4.1967  5.6090
H(Fragment=8) 15.8038 -3.8494  6.5369
C(Fragment=8) 16.1530 -5.4856  5.2463
H(Fragment=8) 16.6395 -6.0472  5.9155
C(Fragment=5) 11.8886 -3.0650  9.5194
C(Fragment=8) 15.9592 -5.9257  4.0431
H(Fragment=8) 16.2824 -6.8408  3.8020
C(Fragment=8) 15.2969 -5.1398  3.0522
H(Fragment=8) 15.1730 -5.4944  2.1254
C(Fragment=8) 14.8285 -3.8823  3.3884
H(Fragment=8) 14.3660 -3.3149  2.7071
S(Fragment=6) 17.9266  1.8091  5.0446

--Link1--
%chk=FragGuess
%mem=64mw
#P UBP86/6-311G*/Auto Guess=Read Geom=AllCheck ···
```

5.41 GVB

This method keyword requests a perfect-pairing General Valence Bond (GVB-PP) calculation. GVB requires one parameter: the number of perfect-pairing pairs to split; for example: GVB(4). This parameter may also be specified with the NPair option. The natural orbitals for the GVB pairs are taken from occupied and virtual orbitals of the initial guess determinant (described in the *Input* tab).

5.41.1 Input

Normally, most of the difficult input for a GVB-PP calculation involves specifying the initial guess (Link 401). This often includes alteration of orbitals to ensure the correct identification of high-spin, perfect-pairing, and closed-shell orbitals and possible reduction of SCF symmetry to account for the localized orbitals which usually represent the lowest energy solution for GVB-PP.

The GVB program reads the number of orbitals in each GVB pair (in format 40I2). The number of lines read is fixed (and normally 1), so no terminating blank line is needed. For a molecule having spin multiplicity S, N GVB pairs, and $n_1, \cdots, n_N$ orbitals in each pair, orbitals from the initial guess are used in the following manner by the GVB program:

- ◊ The S-1 highest occupied orbitals in the initial guess, which would have been singly occupied in an ROHF calculation, become high-spin orbitals.
- ◊ The next lower N occupied orbitals, which would have been doubly occupied in an ROHF calculation, become the first natural orbitals of the GVB pairs.
- ◊ Any remaining orbitals occupied in the guess stay closed-shell.
- ◊ The lowest $n_1 - 1$ virtual orbitals become natural orbitals 2 through n_1 of the first GVB pair, then the next $n_2 - 1$ orbitals are assigned to pair 2, and so on. The GVB-PP scheme does not allow an orbital to be shared by more than one GVB pair.
- ◊ Any remaining (virtual) orbitals from the initial guess become virtual orbitals in the GVB calculation.

Generally Guess=Alter is required to ensure that guess-occupied orbitals, which will be used as first natural orbitals, match up with the correct guess virtual orbitals that will become the corresponding higher natural orbitals. Often it is helpful to start off with Guess=(Local,Only), examine the orbitals to determine alteration requirements, then do Guess=(Local,Alter) and GVB(NPair=N,Freeze) to allow the higher natural orbitals to become more appropriate. Finally, the full calculation can be run with Guess=Read and all orbitals optimized in the GVB. If there is any confusion or concern with the orbitals breaking symmetry, the calculation should be done with Symm=NoSCF and initially with Guess=Local. In fact, this approach is generally recommended except for those very expert users.

If the number of orbitals in a pair is negative, the root of the CI to use for that pair and the pair's initial GVB coefficients are read in format (I2,5D15.8). This is useful if a $^1\Sigma$ or $^1\Delta$ state is being represented as a GVB pair of the form $x^2 \pm y^2$.

5.41.2 Options

NPair

Gives the number of perfect-pairing pairs. GVB(*N*) is equivalent to GVB(NPair=*N*). NPair=0 is acceptable and results in a closed-shell or spin-restricted SCF calculation.

InHam=*N*

Read in N Hamiltonians (Fock operators, sets of coupling coefficients). This option may be combined with perfect-pairing pairs. Each Hamiltonian is read using the following syntax (format in parentheses):

NO	*# of orbitals in current Hamiltonian (I5)*
Fj	*Occup. # (1.0=closed-shell) (D15.8)*
(AJ(I), I =1,NHam)	*J coefficients (5D15.8)*
(AK(I), I =1,NHam)	*K coefficients (5D15.8)*

Combining several orbitals with the same *AJ* and *AK* coefficients into one "shell" is not currently supported, so *NO* is always 1. The *ham506* utility can be used to generate averaged Hamiltonians for the common case of spherical averaging in atomic calculations. The Hamiltonian coefficients are described in Bobrowicz and Goddard [472]. A good introduction to the qualitative interpretation of GVB wavefunctions can be found in the review article by Goddard and Harding [473].

OSS

Do a two-electron, two-orthogonal-orbital open-shell singlet. This option may be combined with perfect-pairing pairs. OpenShellSinglet is a synonym for OSS.

Freeze

Freeze closed-shell and open-shell orbitals and first natural orbitals of GVB pairs, allowing only 2nd and higher orbitals to vary. This option is useful for starting off difficult wavefunctions.

5.41.3 Availability

Energies, analytic gradients, and numerical frequencies.

5.41.4 Examples

Here is a GVB(3/6) calculation performed on singlet methylene:

```
# GVB(3)/6-31G(d) Pop=Full
  Guess=(Local,LowSym,Alter)

GVB(3) on CH2

molecule specification

 1 4 0 2 3 9          Guess=LowSym input
2,3                   Guess=Alter input

 2 2 2                GVB input
```

Each of the 3 valence electron pairs is split into a GVB pair. A preliminary Guess=Only calculation was performed to determine the localized orbitals and which alterations would be required.

The perfect pairing GVB method includes the effects of *intra*-pair correlation but not those of *inter*-pair correlation. Consequently, GVB electrons pairs tend to be localized. In the case of singlet methylene, the carbon lone pair is localized even at the Hartree-Fock level. The canonical Hartree-Fock orbitals for the C-H bonds are delocalized into linear combinations (C-H_1 + C-H_2) and (C-H_1 − C-H_2) having A_1 and B_2 symmetry, respectively. In order to allow the localization in the guess to produce separate bond pairs, these two irreducible representations must be combined. Similarly, the GVB calculation itself must be told not to impose the full molecular symmetry on the orbitals, which would force them to be delocalized. Combining the A_1 and B_2 representations and combining the A_2 and B_1 representations causes the calculation to impose only C_s symmetry on the individual orbitals, allowing separate GVB pairs for each bond. Since the resulting pairs for each bond will be equivalent, the resulting overall wavefunction and density will still have C_{2v} symmetry.

The Guess=LowSymm keyword specifies that the irreducible representations of the molecular point group will be combined in the symmetry information used in a GVB calculation. It takes a single line of input consisting of giving the numbers of the irreducible representations to combine, where the numbers correspond to the order in which the representations are listed in the output file (they appear just after the standard orientation). For example, here is the output for a molecule with C_{2v} symmetry:

```
There are    4 symmetry adapted basis functions of A1  symmetry.
There are    0 symmetry adapted basis functions of A2  symmetry.
There are    1 symmetry adapted basis functions of B1  symmetry.
There are    2 symmetry adapted basis functions of B2  symmetry.
```

Thus for C_{2v} symmetry, the order is A1, A2, B1, B2, referred to in the Guess=LowSym input as 1 through 4, respectively. A zero separates groups of representations to be combined, and a nine ends the list. Thus, to combine A1 with B2 and A2 with B1, thereby lowering the SCF symmetry to C_s, the appropriate input line is:

```
1 4 0 2 3 9
```

Since this information always requires exactly one line, no blank line terminates this section.

The order of orbitals generated after localization by the initial guess in the first job step was C-1s C-H_1 C-H_2 C-2s for the occupied orbitals and C-2p C-H_1* C-H_2* for the lowest virtual orbitals. Hence if no orbitals are interchanged, the C-2s lone pair would be correctly paired with the unoccupied p-orbital, but then the next lower occupied, C-H_2, would be paired with the next higher virtual, C-H_1*. So either the two bond occupied orbitals or the two bond virtual orbitals must be exchanged to match up the orbitals properly.

Finally, the one line of input to the GVB code indicates that there are 2 natural orbitals in each of the 3 GVB pairs.

5.42 HF

This method keyword requests a Hartree-Fock calculation [474]. Unless explicitly specified, RHF is used for singlets and UHF for higher multiplicities. In the latter case, separate α and α orbitals will be computed [475, 476] ([477] for electron correlation methods starting from a UHF reference). RHF, ROHF or UHF can also be specified explicitly.

5.42.1 Availability

Energies, analytic gradients, and analytic frequencies for RHF and UHF and numerical frequencies for ROHF.

5.42.2 Examples

The Hartree-Fock energy appears in the output as follows:

```
SCF Done:  E(RHF) =  -74.9646569691     A.U. after    4 cycles
```

```
      Conv   =      0.48D-08           -V/T =  2.0038
```

For UHF jobs, the output also prints S^2 and related values:

```
SCF Done:  E(UHF) =  -38.7068863059     A.U. after   11 cycles
            Convg  =    0.7647D-08             -V/T =  2.0031
<Sx>= 0.0000 <Sy>= 0.0000 <Sz>= 1.0000 <S**2>= 2.0142 S= 1.0047
<L.S>= 0.000000000000E+00
Annihilation of the first spin contaminant:
S**2 before annihilation     2.0142,   after     2.0001
```

The second and third lines give the SCF convergence limit and the expectation value of S^2.

5.43 Huckel

This method keyword requests an extended Huckel calculation [478–482]. ExtendedHuckel is a synonym for this keyword. No basis set keyword should be specified.

5.43.1 Options

Hoffmann

Requests an Extended Huckel calculation using the default parameter set from the Huckel group.

Muller

Requests an Extended Huckel calculation using parameters collected by Edgar Muller.

Guess

Requests an Extended Huckel calculation using the modified parameters used for Guess=Huckel [483–485].

5.43.2 Availability

Energies, "analytic" gradients, and numerical frequencies.

5.43.3 Examples

The energy appears in the output file as follows (followed by the x, y, and z components of the dipole moment):

```
Huckel eigenvalues -- -1.245  -0.637  -0.558  -0.544  -0.043  0.352
Energy= -5.968836513622 NIter=   0.
Dipole moment=       0.000000       0.000000       0.000000
```

The energy is as defined by this semi-empirical model. Note that energy differences computed from the values in semi-empirical calculations are in Hartrees and may be compared directly with energy differences computed from jobs using other methods.

5.43.4 Related Keywords

Guess=Huckel

5.44 INDO

Requests a semi-empirical calculation using the INDO Hamiltonian [486]. No basis set keyword should be specified.

5.44.1 Availability

Energies, “analytic” gradients, and numerical frequencies.

5.44.2 Examples

The INDO energy appears in the output file as follows:

```
SCF Done:  E(UINDO) =  -8.08016620373     A.U. after   11 cycles
```

The energy is as defined by the INDO model.

5.45 Integral

The Integral keyword modifies the method of computation and use of two-electron integrals and their derivatives.

5.45.1 Options

Integration Grid Selection Option

Grid=*grid-name*

Specifies the integration grid to be used for numerical integrations. Note that it is very important to use the same grid for all calculations where you intend to compare energies e.g., computing energy differences, heats of formation, and so on.

“Pruned” grids are grids that have been optimized to use the minimal number of points required to achieve a given level of accuracy. Pruned grids are used by default when available, currently defined for H through Kr. For example FineGrid is a pruned (75,302 grid), having 75 radial shells and 302 angular points per shell, resulting in about 7000 points per atom. UltraFine requests a pruned (99,590) grid. It is recommended for molecules containing lots of tetrahedral centers and for computing very low frequency modes of systems. This grid is also useful for optimizations of larger molecules with many soft modes such as methyl rotations, making such optimizations more reliable. SuperFineGrid is a more accurate grid than UltraFine; SuperFineGrid is a pruned 175,974 for first-row atoms and 250,974 for atoms in the second and later rows.

Other special values for this parameter are CoarseGrid, which requests a pruned version of the (35,110) grid, and SG1Grid, a pruned version of (50,194). Note, however, that they are not recommended for production calculations [487]. Pass0Grid requests the obsolete pruned (35,110) grid once intended for pass 0 of a tight SCF calculation.

The default grid is UltraFine, and the default grid for the CPHF is SG1. If SG1 is selected as the integration grid, the Coarse grid is used for the CPHF. When a specific grid is specified to the Int=Grid=*grid-name* option, then that grid is also used for the CPHF. Finally, be aware that SG1 is used in the CPHF as the default integration grid for a few DFT jobs including Polar=OptRot, Freq=Anharmonic, and Freq=NNROA.

The parameter to this option is either a grid name keyword or a specific grid specification. If a keyword is chosen, then the option name itself may be omitted (i.e., Integral(Grid=SuperFine) and Integral(SuperFine) are equivalent).

Specific grids may be selected by giving an integer value *N* as the argument to Grid. N may have one of these forms:

◊ A large positive integer of the form *mmmnnn*, which requests a grid with *mmm* radial shells around each atom, and *nnn* angular points in each shell. The total number of integration points per atom is thus *mmm***nnn*. For example, to specify the (99,302) grid, use Int(Grid=99302). The valid numbers of angular points are 38, 50 [488], 72 [489], 86, 110 [488], 146, 194, 302 [490], 434 [491], 590, 770, and 974 [492]. If a larger number of angular points is desired, a spherical product grid can be used.

◊ A large negative integer of the form -*mmmnnn*, which requests *mmm* radial shells around each atom, and a spherical product grid having *nnn* θ points and 2**nnn* φ points in each shell. The total number of integration points per atom is therefore 2**mmm**nnn^2. This form is used to specify the (96,32,64) grid commonly cited in benchmark calculations: Int(Grid=-96032).

Note that any value for *nnn* is permitted; although, small values are silly (values of $nnn < 15$ produce grids of similar size and inferior performance to the special angular grids requested by the second format above). Large values are expensive. For example, a value of 200100 would use 2*200*100*100 or 4 million points per atom!

Algorithm Selection Options

SSWeights

Use the weighting scheme of Scuseria and Stratmann [493] for the numerical integration for DFT calculations. This is the default.

FMMNAtoms=*N*

Set the threshold size for turning on FMM by default to *N*. The default is 60 atoms. Molecules with symmetry have higher crossover points, and the threshold is increased accordingly, to 120 atoms for the C2 and Cs point groups and 240 atoms for higher symmetry.

Symm

NoSymmetry disables and Symm enables the use of symmetry in the evaluation and storage of integrals (Symm is the default). Synonymous with the keywords Symm=[No]Int, which is the recommended usage.

FoFCou

Use routine FoFCou even when it would not otherwise be used. NoFoFCou forbid uses of FoFCou.

LTrace

Trace Linda transactions. Primarily for debugging.

SplitDBFSP

Split density S=P shells into separate S and P shells. NoSplitDBFSP is the default.

ECPAcc=*N*

Set ECP accuracy parameter to *N*.

Acc2E=*N*

Set 2-electron integral accuracy to 10^{-N}. The default is 10^{-12}.

DigestCartesian

Transform integrals from Cartesian to Pure form before digesting them (contracting with density matrices during direct calculations). The default is to decide the based on parameters such as how many density matrices are being processed.

UnconAOBasis

Uncontract all the primitives in the AO basis. UncontractAOBasis is a synonym for this option.

UnconDBF

Uncontract all the primitives in the density fitting basis. UncontractDensityBasis is a synonym for this option.

NoXCTest

Skip tests of numerical accuracy of XC quadrature.

ReadB

Read common /B/ from disk after the initial geometry, even if a standard basis was set up.

General Basis Set-Related Options

Raff, Raf1, Raf2, Raf3

Whether to write out regular or the Raffenetti integrals. Raff and Raff1 integrals write the Raffenetti1 integrals. Raff2 and Raff3 will write the 1 and 2/1,2 and 3 Raffenetti integral combinations (respectively). Primarily useful with External or Output=MatrixElement. NoRaff suppresses writing the Raffenetti integrals and writes out the regular integrals.

BasisTransform=*N*

Transform generalized contraction basis sets to reduce the number of primitives, neglecting primitives with coefficients of 10^{-N} or less. This is the default, with N=4.

ExactBasisTransform

Transform generalized contraction basis sets to reduce the number of primitives, but using only transformations which are exact. Only exact duplicate primitives are removed, and there will be no change in the energy value.

NoBasisTransform

Do not transform generalized contraction basis sets to reduce the number of primitives.

5.45.2 Relativistic Calculations

DKH

Requests a Douglas-Kroll-Hess 2^{nd} order scalar relativistic calculation [494–497] (see [498, 499] for an overview). This method uses a Gaussian nuclear model [500]. DKH2 and DouglasKrollHess are synonyms.

DKH0

Requests a Douglas-Kroll-Hess 0^{th} order scalar relativistic calculation.

DKHSO

Requests a Douglas-Kroll-Hess 4^{th} order relativistic calculation including spin-orbit terms (if doing GHF/GKS).

RESC

Requests a RESC scalar relativistic calculation.

NoDKH and NonRelativistic request a non-relativistic core Hamiltonian, which is the default.

5.45.3 Related Keyword

SCF

5.46 IOp

The IOp keyword allows the user to set internal options (variables in system common /IOp/) to specific values. The syntax is:

IOp($overlay_a/option_a = value_a$ [,$overlay_b/option_b = value_b, \cdots$])

which sets option number $option_a$ to the value $value_a$ for every occurrence of overlay number $overlay_a$ (and so on for other entries in the list). For example, the following sets the value of option 12 in overlay 1 to 5 and that of option 44 in overlay 3 to 0: IOp(1/12=5,3/44=0)

IOp values explicitly set in the route section are not passed on to the second and subsequent automatically-generated job steps; this applies to keyword combinations like Opt Freq and to inherently multi-step methods such as G2 and the CBS methods. For example, if you want to specify an alternate grid for a DFT optimization+frequency job, you must use an option to the Int=Grid keyword rather than an explicit IOp value.

The execution of each overlay of Gaussian 16 is controlled by numbered options. Each option may be assigned an integer value, with 0 being the default. The value of an option is held unchanged throughout execution of all of the links in one overlay. Thus the significance of a particular option applies to all the component links in one pass through the overlay. Since setting internal options can have arbitrary effects on the calculation, archiving is disabled by use of this keyword.

The full list of Gaussian 16 options is given in the *Gaussian 16 IOps Reference*.

5.47 IRC

This calculation type keyword requests that a reaction path be followed by integrating the intrinsic reaction coordinate [501, 502]. The initial geometry (given in the molecule specification section) is that of the transition state, and the path can be followed in one or both directions from that point. The *forward* direction is defined as the direction the transition vector is pointing when the largest component of the transition vector ("phase") is positive; it can be defined explicitly using the Phase option. By default, both reaction path directions are followed.

IRC calculations *require* initial force constants to proceed. You must provide these to the calculation in some way. The usual method is to save the checkpoint file from the preceding frequency calculation (used to verify that the optimized geometry to be used in the IRC calculation is in fact a transition state), and then specify the RCFC option in the route section. Another possibility is to compute them at the beginning of the IRC calculation (CalcFC). Note that one of RCFC and CalcFC must be specified (CalcAll is also available but is not typically necessary with the HPC algorithm).

In Gaussian 16, most calculations use the HPC algorithm [502–504] by default (introduced in Gaussian 09). It is much more efficient than the one used in earlier program versions. ONIOM(MO:MM) calculations use the Euler predictor-corrector integration algorithm. This same integrator is also used by default in calculations using methods with gradients but without analytic second derivatives; such calculations should include the GradientOnly option. Available algorithms are discussed in *Availability*.

The default is to report only the energies and reaction coordinate at each point on the path; if geometrical parameters along the path are desired, these should be defined as redundant internal coordinates via Geom=ModRedundant or as input to the IRC code via IRC(Report=Read) (see *Options* for the latter's input format).

You can specify alternative isotopes for IRC jobs using the ReadIsotopes option (described in *Options*).

5.47.1 Options

Path Specification Options

Phase=(*N1*, *N2* [,*N3* [,*N4*]])

Defines the phase for the transition vector such that forward motion along the transition vector corresponds to an increase in the specified internal coordinate, designated by up to four atom numbers. If two atom numbers are given, the coordinate is a bond stretch between the two atoms; three atom numbers specify an angle bend; and four atoms define a dihedral angle.

Forward

Follow the path only in the forward direction.

Reverse

Follow the path only in the reverse direction.

Downhill

Proceed downhill from the input geometry.

MaxPoints=*N*

Number of points along the reaction path to examine (in each direction if both are being considered). The default is 10.

StepSize=*N*

Step size along the reaction path, in units of 0.01 Bohr. If N<0, then the step size is taken in units of 0.01 amu$^{1/2}$-Bohr. The default is 10.

ReadIsotopes

This option allows you to specify alternatives to the default temperature, pressure, frequency scale factor and/or isotopes – 298.15 K, 1 atmosphere, no scaling, and the most abundant isotopes (respectively). It is useful when you want to rerun an analysis using different parameters from the data in a checkpoint file. Be aware, however, that all of these can be specified in the route section (Temperature, Pressure and Scale keywords) and molecule specification (the Iso parameter), as in this example:

```
#T Method/6-31G(d) JobType Temperature=300.0 ···

...

0 1
C(Iso=13)
...
```

ReadIsotopes input has the following format:

temp pressure [scale]	*Values must be real numbers.*
isotope mass for atom 1	

isotope mass for atom 2

...

isotope mass for atom n

Where *temp*, *pressure*, and *scale* are the desired temperature, pressure, and an optional scale factor for frequency data when used for thermochemical analysis (the default is unscaled). The remaining lines hold the isotope masses for the various atoms in the molecule, arranged in the same order as they appeared in the molecule specification section. If integers are used to specify the atomic masses, the program will automatically use the corresponding actual exact isotopic mass (e.g., 18 specifies ^{18}O, and Gaussian uses the value 17.99916).

Algorithm Selection Options

HPC

Use the Hessian-based Predictor-Corrector integrator [502–504]: a very accurate algorithm that uses the Hessian-based local quadratic approximation as the predictor component and a modified Bulrisch-Stoer integrator for the corrector portion. This corrector integrator is done using a distance-weighted interpolant surface [505] fitted to energies, gradients, and Hessians at the beginning and ending points of the predictor step. This is the default for most calculations. Note that it is not practical for extremely large molecular systems.

EulerPC

Use the first-order Euler integration for the predictor step along with the HPC corrector step. This is the default for IRC=GradientOnly calculations. This is also the default algorithm for IRC calculations using an ONIOM(MO:MM) method. It is a practical choice for such calculations on large molecules.

LQA

Use the local quadratic approximation [406, 409] for the predictor step.

DVV

Use the damped velocity verlet integrator [506].

Euler

Use only the first-order Euler integration predictor step for the IRC. This option is not recommended for production use.

ReCalc=*N*

Compute the Hessian analytically every *N* predictor steps or every |*N*| corrector steps if *N*<0. Analytic second derivatives can be requested intermittently during IRCs using IRC=(CalcFC,RecalcFC=(Predictor=*N*, Corrector=*M*)), which computes second derivatives at the initial point and then at every N^{th} predictor step and every M^{th} corrector step. You must still specify RCFC or CalcFC to provide the initial Hessian. Update is a synonym for ReCalc. Requires a method which has analytic second derivatives.

GradientOnly

Use an algorithm that does not require second derivatives. Note that you must specify this option explicitly for such methods (they are not automatically detected). Can be combined with EulerPC (the default), HPC, Euler, or DVV.

Coordinate System Selection Options

MassWeighted

Follow the path in mass-weighted Cartesian coordinates. MW is a synonym for MassWeighted. This is the default.

Cartesian

Follow the path in Cartesian coordinates without mass-weighting.

Options For Generating Initial Force Constants

RCFC

Specifies that the computed force constants in Cartesian coordinates from a frequency calculation are to be read from the checkpoint file. ReadCartesianFC is a synonym for RCFC.

CalcFC

Specifies that the force constants be computed at the first point.

Procedure-Related Options

Restart

Restarts an IRC calculation that did not complete, or restarts an IRC calculation which did complete, but for which additional points along the path are desired.

Report[=*item*]

Controls which geometric parameters are reported by an IRC. By default, no geometrical parameters are reported. Report without a parameter includes all of the generated internal coordinates. The possible values for *item* are:

Read	Read a list of internal coordinates to report. These are specified as 1-4 atom numbers *only* (ModRedundant-style coordinate specifications are not supported).
Bonds	Reports bonds from the internal coordinates (if present).
Angles	Reports angles from the internal coordinates (if present).
Dihedrals	Reports dihedrals from the internal coordinates (if present).
Cartesians	Reports all Cartesian coordinates.

ReCorrect[=*when*]

Controls testing-and-recomputing for the correction step of HPC and EulerPC IRCs. ReCorrect (without a parameter) and ReCorrect=Yes say to repeat the corrector step whenever the correction is greater than the threshold (which can be decreased with the Tight and VTight options). The parameter can take on the following values:

Never	Do not repeat correction steps (i.e., suppress the threshold test).
Always	Always recompute the corrector at least once regardless of the size of the initial correction.
Test	Test the quality of the corrector step and report the results, but do not take an additional corrector step. The computed IRC path will be the same as with ReCorrect=Never.

The default is Yes for EulerPC and HPC, and Never for other integrators.

MaxCycle=*N*

Sets the maximum number of steps to *N*. The default is 20.

Tight

This option tightens the cutoffs on forces and step size that are used to determine convergence. For DFT calculations, Int=UltraFine should be specified as well.

VeryTight

Extremely tight optimization convergence criteria. VTight is a synonym for VeryTight. For DFT calculations, Int=UltraFine should be specified as well.

Options For Compatibility with Gaussian 03

The GS2 option requests the IRC algorithm used in Gaussian 03 within the new IRC implementation. Use the keyword Use=L115 in order to run the code that was the default in Gaussian 03 (recommended for reproducing old results only).

GS2

Use the IRC algorithm that was the default in Gaussian 03 and earlier [507, 508]. The geometry is optimized at each point along the reaction path such that the segment of the reaction path between any two adjacent points is described by an arc of a circle and so that the gradients at the end points of the arc are tangent to the path. By default, a GS2 IRC calculation steps 6 points in mass-weighted internal coordinates in the forward direction and 6 points in the reverse direction, in steps of 0.1 amu$^{1/2}$Bohr along the path.

CalcAll

Specifies that the force constants be computed at every point.

5.47.2 Availability

The default algorithms are available for HF, all DFT methods, CIS, TD, MP2, MP3, MP4(SDQ), CID, CISD, CCD, CCSD, QCISD, BD, CASSCF, and all semi-empirical methods.

5.47.3 Related Keywords

Opt, Scan, IRCMax

5.47.4 Examples

When the IRC has completed, the program prints a table summarizing the results:

```
Reaction path calculation complete.

Energies reported relative to the TS energy of        -91.564851
---------------------------------------------------------------------
   Summary of reaction path following
---------------------------------------------------------------------
                       Energy   Rx Coord
   1                  -0.00880  -0.54062
   2                  -0.00567  -0.43250
   3                  -0.00320  -0.32438
```

```
    4                    -0.00142  -0.21626
    5                    -0.00035  -0.10815
    6                     0.00000   0.00000      transition state
    7                    -0.00034   0.10815
    8                    -0.00131   0.21627
    9                    -0.00285   0.32439
   10                    -0.00487   0.43252
   11                    -0.00725   0.54065
 ---------------------------------------------------------------------
```

The initial geometry (transition structure) appears in the middle of the table (in this case, as point 6). It can be identified quickly by looking for reaction coordinate and energy values of 0.00000.

5.48 IRCMax

Performs an IRCMax calculation using the methods of Petersson and coworkers [509–517]. Taking a transition structure as its input, this calculation type finds the maximum energy along a specified reaction path, using the GS2 algorithm [507, 508] (see IRC=GS2 for details).

IRCMax requires two model chemistries as its options, separated by a colon: IRCMax(*model2*:*model1*): e.g., IRCMax(B3LYP/6-31G(d,p):HF/6-31G(d,p))

5.48.1 Options

ZC-VTST Options

Zero

Include the zero-point energy in the IRCMax computation.

Path Selection Options

Forward

Follow the path only in the forward direction.

Reverse

Follow the path only in the reverse direction.

ReadVector

Read in the vector to follow. The format is Z-matrix (FFF(I), I=1,NVAR), read as (8F10.6).

MaxPoints=*N*

Number of points along the reaction path to examine (in each direction if both are being considered). The default is 6.

StepSize=*N*

Step size along the reaction path, in units of 0.01 amu$^{1/2}$-Bohr. The default is 10.

MaxCyc=*N*

Sets the maximum number of steps in each geometry optimization. The default is 20.

Coordinate System Selection Options

MassWeighted

Follow the path in mass-weighted internal (Z-matrix) coordinates (which is equivalent to following the path in mass-weighted Cartesian coordinates). MW is a synonym for MassWeighted. This is the default.

Internal

Follow the path in internal (Z-matrix) coordinates without mass-weighting.

Cartesian

Follow the path in Cartesian coordinates without mass-weighting.

Convergence-Related Option

VeryTight

Tightens the convergence criteria used in the optimization at each point along the path. This option is necessary if a very small step size along the path is requested.

Options For Generating Initial Force Constants

CalcFC

Specifies that the force constants be computed at the first point.

CalcAll

Specifies that the force constants be computed at every point. The projected vibrational frequencies for motion are calculated for each optimized point on the path [453]. Note that these projected vibrational frequencies are valid only for reaction paths computed in mass-weighted internal coordinates.

Specifying Alternate Isotopes

ReadIsotopes

This option allows you to specify alternatives to the default temperature, pressure, frequency scale factor and/or isotopes – 298.15 K, 1 atmosphere, no scaling, and the most abundant isotopes (respectively). It is useful when you want to rerun an analysis using different parameters from the data in a checkpoint file. Be aware, however, that all of these can be specified in the route section (Temperature, Pressure and Scale keywords) and molecule specification (the Iso parameter), as in this example:

```
#T Method/6-31G(d) JobType Temperature=300.0 ···

...

0 1
C(Iso=13)
...
```

ReadIsotopes input has the following format:

temp pressure [scale]	*Values must be real numbers.*
isotope mass for atom 1	
isotope mass for atom 2	
...	
isotope mass for atom n	

Where *temp*, *pressure*, and *scale* are the desired temperature, pressure, and an optional scale factor for frequency data when used for thermochemical analysis (the default is unscaled). The remaining lines hold the isotope masses for the various atoms in the molecule, arranged in the same order as they appeared in the molecule specification section. If integers are used to specify the atomic masses, the program will automatically use the corresponding actual exact isotopic mass (e.g., 18 specifies ^{18}O, and Gaussian uses the value 17.99916).

Restart Option

Restart

Restarts an IRCMax calculation which did not complete, or restarts an IRCMax calculation which did complete, but for which additional points along the path are desired.

5.48.2 Availability

Analytic gradients are required for the IRC portion of the calculation (*model1* above). Any non-compound energy method and basis set may be used for *model2*.

5.48.3 Related Keywords

IRC, Opt, Freq

5.48.4 Examples

The following calculation will find the point on the HF/6-31G(d,p) reaction path where the B3LYP/6-31G(d,p) energy is at its maximum:

```
# IRCMax(B3LYP/6-31G(d,p):HF/6-31G(d,p))
```

The Zero option will produce the data required for zero curvature variational transition state theory (ZC-VTST) [509–512, 514, 517]. Consider the following route:

```
# IRCMax(MP2/6-311G(d):HF/6-31G(d),Zero,Stepsize=15,CalcAll)
```

This job will start from the HF/6-31G(d) TS and search along the HF/6-31G(d) IRC with a step size of 0.15 amu$^{1/2}$Bohr until the maximum of the MP2/6-311G(d) energy (including the HF/6-31G(d) ZPE) is bracketed. The position along the HF/6-31G(d) IRC for this MP2/6-311G(d) TS will then be optimized. The output includes all quantities required for the calculation of reaction rates using the ZC-VTST version of absolute rate theory: TS moments of inertia, all real vibrational frequencies (HF/6-31G(d)), the imaginary frequency for tunneling (fit to MP2/6-311G(d) + ZPE), and the total MP2/6-311G(d) + ZPE energy of the TS. Note if CalcAll is not used then all these quantities (ZPE, frequencies, etc.) are only computed at the HF/6-31G(d) level and the same quantities are used for all points in the IRCMax path.

5.49 LSDA

This method keyword requests a Local Spin Density Approximation calculation, using the Slater exchange functional and the VWN correlation functional for the DFT calculation. It is equivalent to SVWN. Note that LSDA is not uniquely defined in the literature. In fact, many differing but related methods are referred to using this term. Other programs offering an LSDA method may use somewhat different functionals. For example, some implement the functional specified by the SVWN5 keyword, while others use a correlation functional of Perdew. While Gaussian offers this keyword for convenience, it is probably better practice to specify the exact functional desired; see *DFT Methods* for full details on specifying and using Density Functional Methods in Gaussian.

5.50 MaxDisk

The MaxDisk keyword specifies the amount of disk storage available for scratch data, in 8-byte words (default). This value may also be followed by KB, MB, GB, TB, KW, MW, GW or TW (without intervening spaces) to specify units of kilo-, mega-, giga- or tera-bytes or words. Normally, this is set for a site in the *Default.Route* file.

Not all calculations can dynamically control their disk usage, so the effects of this keyword vary:

- ◊ SCF energy, gradient, and frequency calculations use a fixed amount of disk. This is quite small (only cubic in the size of the system) and is not usually a limitation.
- ◊ MP2 energies and gradients obey MaxDisk, which must be at least $2ON^2$.
- ◊ Analytic MP2 frequencies attempt to obey MaxDisk, but have minimum disk requirements.
- ◊ CI-Singles energies and gradients in the MO basis require about $4O^2N^2$ words of disk for a limited set of transformed integrals. Additional scratch space is required during the transformation and this is limited as specified by MaxDisk. This disk requirement can be eliminated entirely by performing a direct CI-Singles calculation by using CIS=Direct.
- ◊ CID, CISD, CCD, BD, and QCISD energies also have a fixed storage requirement proportional to O^2N^2, with a large factor, but obey MaxDisk in avoiding larger storage requirements.
- ◊ CCSD, CCSD(T), QCISD(T), and BD(T) energies have fixed disk requirements proportional to ON^3 which cannot be limited by MaxDisk.
- ◊ CID, CISD, CCD, QCISD densities, and BD and CCSD densities and gradients have fixed disk requirements of about $N^4/2$ for closed-shell and $3N^4/4$ for open-shell.
- ◊ EOM-CCSD initially sets its space requirements to its minimum needs. If this amount is more than MaxDisk, it obeys the latter at the expense of I/O and computation time.

5.51 MINDO3

The MINDO3 keyword requests a semi-empirical calculation using the MINDO3 Hamiltonian [518, 519] method. No basis set keyword should be specified.

5.51.1 Availability

Energies, "analytic" gradients, and numerical frequencies. Restricted open-shell (RO) wavefunctions are limited to optimizations using the Fletcher-Powell and pseudo-Newton-Raphson methods (the FP and EnOnly options to Opt, respectively).

5.51.2 Examples

The MINDO3 energy appears in the output file as follows:

```
 SCF Done:  E(UMINDO3) =  -8.08016620373     A.U. after   11 cycles
```

The energy is as defined by the MINDO3 model.

5.52 MNDO

The MNDO keyword requests a semi-empirical calculation using the MNDO Hamiltonian [519–527] method. No basis set keyword should be specified.

5.52.1 Availability

Energies, "analytic" gradients, and numerical frequencies. Restricted open shell (RO) wavefunctions are limited to optimizations using the Fletcher-Powell and pseudo-Newton-Raphson methods (FP and EnOnly, respectively).

5.52.2 Examples

The MNDO energy appears in the output file as follows:

```
SCF Done:  E(UMNDO) =  -8.08016620373     A.U. after   11 cycles
```

The energy is as defined by the MNDO model.

5.53 MM (Molecular Mechanics) Methods

There are three Molecular Mechanics methods available in Gaussian. They were implemented for use in ONIOM calculations, but they are also available as independent methods. No basis set keyword should be specified with these keywords.

The following force fields are available:

◊ Amber: The Amber force field as described in [Cornell95]. The actual parameters (*parm96.dat*) have been updated slightly since the publication of this paper. We use this current version from the Amber web site (`ambermd.org`).

◊ Dreiding: The Dreiding force field as described in [Mayo90].

◊ UFF: The UFF force field as described in [Rappe92].

5.53.1 Availability

Analytic energies, gradients, and frequencies.

5.53.2 Atom Types & Charges

Molecular Mechanics Molecule Specifications

Molecular Mechanics calculations use atom types to determine the functions and parameters which make up the force field. For a single element, such as carbon, there can be many different MM atom types. Which one to use depends on aspects such as the hybridization, chemical environment, and so on.

Gaussian will attempt to assign atom types automatically for UFF and Dreiding calculations. However, Amber calculations require that all atom types be explicitly specified within the molecule specification section, as in these examples:

`C-CT`	*Specifies an SP3 aliphatic carbon atom, denoted by the Amber CT keyword.*
`C-CT-0.32`	*Specifies an SP3 aliphatic carbon atom with a partial charge of +0.32.*
`O-O-0.5`	*Specifies a carbonyl group oxygen atom (type O) with a partial charge of -0.5.*

The atom type keyword is specified following the element symbol and a separating hyphen. Consult the Amber paper [528] for definitions of atom types and their associated keywords.

As the second and third lines in the example illustrate, partial charges may also be specified, as the third component of the atom description, preceded again by a hyphen separator. Partial atomic (point) charges are used to compute the electrostatic interactions. Gaussian can assign these charges automatically using the QEq formalism [529]. Request this by specifying the QEq option to the MM keyword: e.g., Dreiding=QEq. Note that these charges depend on the geometry and are computed at the beginning of the calculation; however, they are not updated during the course of an optimization or other process that changes the geometry.

Atom types and charges may also be provided for UFF and Dreiding calculations if desired. Be aware that the automatic assignment of UFF and Dreiding atom types is only reliable for well-defined systems, and it is often safer to specify them explicitly.

When using MM methods, and especially when you are modifying force field terms or defining new ones (as discussed below), it is often necessary to specify the connectivity between atoms explicitly, using Geom=Connectivity. Note that GaussView includes this input section automatically when it constructs input files.

Charge Assignment-Related Options

Unless set in the molecule specification input, no charges are assigned to atoms by default when using any molecular mechanics force field. Options are available to estimate charges at the initial point using the QEq algorithm [529], under control of the following options, for any of the mechanics keywords:

QEq

Assign charges to all atoms using the QEq method [529]. OldQEq requests an older version which was default in G09 Rev. C [530].

UnTyped

Assign QEq charges only to those atoms for which the user did not specify a particular type in the input.

UnCharged

Assign QEq charges only for atoms which have MM charge zero (i.e., all atoms which were untyped or which were given a type but not a charge in the input), constraining the other atoms to retain their specified charges.

5.53.3 Options

Parameter Precedence Options

Terminology: Gaussian contains built-in parameter sets for the built-in force fields listed above; these are referred to as *hard-wired* parameters. *Soft* parameters are ones specified by the user in the input stream for the current job (or a previous job when reading parameters from the checkpoint file). By default, when no relevant option is given, the hard-wired parameters are the *only ones* used. This topic is discussed in detail in Specifications tab.

HardFirst

Read additional parameters from the input stream, with hard-wired parameters having priority over the read-in, soft ones. Hence, read-in parameters are used only if there is no corresponding hard-wired value. Note that wildcard matches within the hard-wired parameter set take precedence over soft parameters, even when the latter contains an exact match for the same item. Use SoftFirst if you want to override hard-wired parameter matches.

SoftFirst

Read additional parameters from the input stream, with soft (read-in) parameters having priority over the hard-wired values.

SoftOnly

Read parameters from the input stream and use only them, ignoring hard-wired parameters.

Handling Multiple Parameter Specification Matches

Since parameters can be specified using wildcards (see the Specifying Force Fields tab), it is possible for more than one parameter specification to match a given structure. The default is to abort if there are any ambiguities in the force field. The following options specify other ways of dealing with multiple matches.

FirstEquiv

If there are equivalent matches for a required parameter, use the first one found.

LastEquiv

If there are equivalent matches for a required parameter, use the last one found.

Reading MM Parameters from the Checkpoint File

By default, when the geometry is read-in from the checkpoint file with Geom=Check or Geom=AllCheck, any non-standard (soft) parameters present in the checkpoint file are also retrieved. These read-in parameters are used with higher priority than corresponding hard-wired parameters, unless HardFirst is also specified. The following options can be used to modify this default behavior.

ChkParameters

Read only the MM parameters from the checkpoint file. Any non-standard (soft) parameters present in the checkpoint file are used with higher priority than corresponding hard-wired parameters, unless HardFirst is also specified. Not valid with Geom=Check or Geom=AllCheck.

NewParameters

Ignore any parameters in the checkpoint file when reading the geometry.

Modify

Read additional parameter specifications and modifications after combining any soft parameters from the checkpoint file and the hard-wired parameters (as controlled by the precedence options). This option works analogously to Geom=Modify.

Printing MM Parameters

Print

By default, only the contributions to the energy are printed (i.e., from bond stretches, bends, electrostatics, etc.) when #P is specified. The Print option will also print the energy contributions and the force field parameters at the first geometry, and the energy contributions at later ones (since the parameters don't change). It will print two sets of parameters for ONIOM(MO:MM) since different parameters are assigned for the model and real systems (i.e., since H may replace a C link atom).

Note that the parameters are printed for all atoms in the system, but not in a form suitable for input. The internally stored parameters are already provided in the form of input files in the main Gaussian directory (*amber.prm*, *uff.prm*, etc.), along with files for some force fields which are not stored internally (*amber98.prm* and *oplsaa.prm*). Printing for all atoms in the current job is still useful because complex rules are involved in deciding what parameters are actually applied to a given bond, dihedral, etc.

Switching Functions Options

VRange=*N*

Set the van der Waals cutoff to *N* Å. This is ROff if soft cutoffs are used, which is the default. *N*<0 for hard cutoffs.

CRange=*N*

Set the Coulomb cutoff to N Å. Negative for hard cutoffs. The potential is $(Q_iQ_j(1-R^2/N^2)^2)/R$

CutRad=*M*

Use soft cutoffs with radius M (in Å), with either van der Waals or Coulomb if VRange or CRange was set. Thus with N, M the soft cutoffs have ROn=*N*-*M*, ROff=*N*.

Switch=*value*

Switch=YK means use the York-Karplus function, which is the default. Switch=CHARMM means use CHARMM-style switching for van der Waals. Switch=CHARMMSq means to use CHARMM-style expression but with R^2 instead of R for van der Waals switching. Switch=*N* means use switching function number *N*.

5.53.4 MM Potential Functions

Gaussian offers a tremendous amount of flexibility to users that want to modify or otherwise customize MM force fields. This section discusses the specifics of doing so, and it begins with some required background information necessary for understanding the function and parameter input.

A basic MM potential function is usually of the form:

$$E^{total} = \underset{\text{stretch terms}}{\sum_{bonds} K_r (r - r_{eq})^2} + \underset{\text{bend terms}}{\sum_{angles} K_\theta (\theta - \theta_{eq})^2} + \underset{\text{torsional terms}}{\sum_{dihedrals} \frac{V_n}{2} [1 + \cos(n\phi - \gamma)]} + \underset{\text{non-bonded interactions}}{\sum_{i<j} \left(\frac{A_{ij}}{r_{ij}^{12}} - \frac{B_{ij}}{r_{ij}^{6}} + \frac{q_i q_j}{\varepsilon r_{ij}} \right)}$$

Molecular Mechanics Potential Function

As noted, the terms of the expansion are the stretch terms (bonds), bend terms (bond angles), torsional terms (dihedral angles), and non-bonded interactions. The particular forms of the individual functions in this equation are from the Amber force field, which uses simple harmonic functions for stretches and bends, a cosine function for torsions, and the standard functions for electrostatic and van der Waals interactions. Other force fields use different functions, and some include additional types of terms not included by Amber, such as dipole interactions or stretch-bend couplings.

In order to evaluate the MM potential function, Gaussian needs to know which structures – stretches, bends, and torsion – are present in the system, as well as the functions and parameters to be used for them. The structures can be identified from the molecular connectivity. By default, Gaussian determines which atoms are bonded and the corresponding bond orders (single, double, and so on). It does a good job when the input geometry is reasonable and the bond orders (single, double, etc.) are all well defined. However, if the calculation is started with a more approximate geometry, or if there are bonds whose orders are not easy to identify, it is safer to specify the connectivity list explicitly in the input stream, using Geom=Connectivity.

Viewing the MM Functions

A useful way to begin consideration of this topic is to examine the output from a Gaussian MM job. Here is the input file we used for an Amber calculation on methane:

```
#P Amber=Print Geom=Connectivity

Methane
```

```
0 1
C-CT--0.4     -0.85  0.42  0.00
H-HC-0.1      -0.50 -0.57  0.00
H-HC-0.1      -0.50  0.93  0.87
H-HC-0.1      -0.50  0.93 -0.87
H-HC-0.1      -1.92  0.42  0.00
```

Note that the atoms types are CT for the carbon atom and HC for the hydrogen atoms. We have also assigned partial charges to each atom.

Here is the portion of the output produced by Amber=Print in the job's route section:

MM functions for single atoms.

```
 Atomic parameters:
 Center  VDW
      1 1.9080  .1094000
      2 1.4870  .0157000
      3 1.4870  .0157000
      4 1.4870  .0157000
      5 1.4870  .0157000
```

MM functions of 2 of more atoms.

Non-bonded interactions master function.

```
 Molecular mechanics terms:
 NBDir  3 1      0      0
```

Stretch terms.

```
 HrmStr1 1-2  340.00 1.0900
 HrmStr1 1-3  340.00 1.0900
 HrmStr1 1-4  340.00 1.0900
 HrmStr1 1-5  340.00 1.0900
```

Bend terms.

```
 HrmBnd1 2-1-3  35.00 109.50
 HrmBnd1 2-1-4  35.00 109.50
 HrmBnd1 2-1-5  35.00 109.50
 HrmBnd1 3-1-4  35.00 109.50
 HrmBnd1 3-1-5  35.00 109.50
 HrmBnd1 4-1-5  35.00 109.50
```

Non-bonded interactions of close neighbors.

```
 NBPair 2-1  3 1      0      0 -1.000 -1.000
 NBPair 3-1  3 1      0      0 -1.000 -1.000
 NBPair 3-2  3 1      0      0 -1.000 -1.000
 NBPair 4-1  3 1      0      0 -1.000 -1.000
 NBPair 4-2  3 1      0      0 -1.000 -1.000
 NBPair 4-3  3 1      0      0 -1.000 -1.000
 NBPair 5-1  3 1      0      0 -1.000 -1.000
 NBPair 5-2  3 1      0      0 -1.000 -1.000
 NBPair 5-3  3 1      0      0 -1.000 -1.000
 NBPair 5-4  3 1      0      0 -1.000 -1.000
```

Atomic Parameters

Some MM functions depend only on the atom type of the atom in question. In our example, Amber includes the van der Waals interaction for each atom, and the resulting values are listed in the preceding output under Atomic parameters. The center number corresponds to the atom position within the molecule specification. The DREIDING and UFF force fields contain only terms of this type.

Force Field Terms lists terms within the total potential that describe the interactions of multiple atoms. For example, there are stretch terms involving each bonded pair of atoms (1-2, 1-3 and 1-4), computed via the function called HrmStr1, using a force constant value of 340 kcal/(mol-Å^2) and an equilibrium bond length of 1.09 Å. These parameter values are determined from the atom types of centers 1 and 2: CT and HC (respectively).

The particular function and parameters are chosen as follows: from the connectivity, the program knows that a bond exists between centers 1 and 2. The Amber force field calls for a stretch term between all bonded pairs of atoms, and the various functions and parameters are stored in internal tables within the program. In the simplest case, Gaussian uses the two atom types to look up functions and parameters to use. In this case, the entry

```
HrmStr1 CT HC 340 1.09
```

is used since it corresponds to the two atom types in the C-H bond in methane. The force constant and equilibrium bond length specified in the entry are then used within the computation. When we look up this function in the reference section below, we see that the actual term will be computed as $ForceC * (R - R_{eq})^2$ where $ForceC$ is the force constant, R_{eq} is the equilibrium bond length and R is the bond length; in this case, this becomes 340.0*(R-1.09)^2.

The list of terms includes stretch terms for all bonded pairs of atoms and bend terms for all corresponding bond angles.

Non-Bonded Interaction Terms

The remaining terms in the list are those corresponding to non-bonded interactions: contributions to the potential arising from interactions between atoms which are not bonded. Before considering them specifically, we need to look at how they are computed.

In general, the list of non-bonded terms is formed from all possible pairs of atoms, and thus the number of terms scales quadratically with the number of atoms. However, interactions between atoms that are close together, based on the number of bonds that separate them, are usually scaled down. Typically, one- and two-bond separated interactions are scaled to zero (since these interactions are described by stretch and bend terms). Similarly, three-bond separated interactions are scaled with a factor between zero and one, depending on the force field. It is clear that for larger systems, the list of non-bonded terms can become very long. In most MM programs, cutoffs are used the keep the list manageable. In Gaussian, we use a less common approach to efficiently deal with the non-bonded interactions.

The total non-bonded energy in the system can be written as follows:

$$
\begin{aligned}
E^{NB} &= \sum_{i}^{N}\sum_{j}^{j<i}\left[s_{ij}^{vdW}E_{ij}^{vdW} + s_{ij}^{Q}E_{ij}^{Q}\right] \\
&= \underbrace{\sum_{i}^{N}\sum_{j}^{j<i}\left[E_{ij}^{vdW} + E_{ij}^{Q}\right]}_{\text{all pairs}} + \underbrace{\sum_{i}^{N}\sum_{j}^{j<i}\left[\left(1 - s_{ij}^{vdW}\right)E_{ij}^{vdW} + \left(1 - s_{ij}^{Q}\right)E_{ij}^{Q}\right]}_{\text{"overcounted" pairs}}
\end{aligned}
$$

Total Non-Bonded Energy Expression

Where E^{vdW} and E^{Q} are the van der Waals and electrostatic interactions between the two centers (respectively), and the corresponding s values are the associated scaling factors.

The first equation is expressed in the usual form for the non-bonded interactions, and the second one is an equivalent expression where all possible non-bonded interactions (regardless of distance) are evaluated without scaling (first term) followed by subtraction of the overcounted pairs (second term).

This reformulation has significant computational advantages. The first term can be evaluated in an efficient manner, using linear scaling and other techniques. In addition, most of the pairs in the second term give a zero interaction (provided the system is not too small), because the scaling factor s_{ij} is 1 for most pairs. The program simply keeps a list of those pairs that give non-zero interactions to compute the second term. The result is an efficient algorithm that does not use cutoffs, and requires only a list whose length scales linearly with the size of the system.

The functions NBDir and NBTerm are used when forming these two non-bonded interaction terms. As the previous methane job output illustrates, a single NBDir function describes the complete set of interactions between all pairs of atoms, i.e., the first term of the rewritten equation. The various NBTerm entries comprise the second term in the equation. The final two parameters to these functions are the scale factors for van der Waals and Coulomb interactions (respectively). In the methane example, both scale factors are 1 for all terms since no atoms are separated by more than two bonds in methane, and the final non-bonded interaction is accordingly 0.

5.53.5 Specifying Force Fields

Adding and Replacing Parameters

When hard-wired parameters are not available, either from incomplete tables or the introduction of new atom types, they need to be specified in the input stream. Determining the appropriate parameters to use can be challenging. They can often be found in the literature; otherwise they need to be derived from experimental or accurate computational data (this topic is beyond the scope of our discussion here).

To specify new parameters, use the Amber=HardFirst keyword, and then add them to the input file in a separate section. In this case, we define *HrmBnd1* functions for the missing atom type sequences. The 2-methylpropene job then looks like:

```
# Amber=HardFirst

2-methylpropene

0 1
 C-CM       -2.53    0.19    0.00
```

molecule specification continues···

```
HrmBnd1 CT CM CT 70.0 120.0
HrmBnd1 HA CM HA 40.0 120.0
```

The parameter values used in this example are a rough guess based on existing Amber parameters. The HardFirst option indicates that the usual Amber hard-wired parameters tables are checked first for functions – referred to as the hard parameters – and that functions defined in the input file are to be used only when no match is present. With the SoftFirst option, one can replace parameters from the hard-wired set. Note that the ordering specified by HardFirst and SoftFirst only matters when one or more functions is defined in both parameter lists. Finally, the SoftOnly option indicates that the complete force field definition will be provided in the input file, ignoring the hard-wired set completely.

Specifying Missing Parameters

Consider the following molecule: 2-methylpropene. We've set up an Amber calculation for this system, specifying the SP2 hybridized carbons as atom type *CM*, and the SP3 hybridized carbons as *CT*.

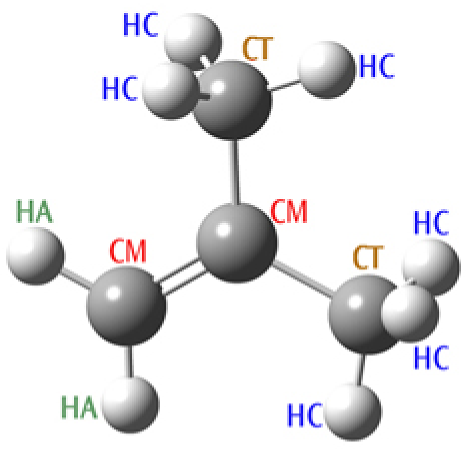


Figure 5.2: 2-MethylPropene

```
# Amber

2-methylpropene

0 1
 C-CM        -2.53    0.19    0.00
 H-HA        -2.00   -0.73    0.00
 H-HA        -3.60    0.19    0.00
 C-CM        -1.86    1.37    0.00
 C-CT        -2.63    2.70    0.00
 H-HC        -1.93    3.51    0.07
 H-HC        -3.24    2.76   -0.88
 H-HC        -3.24    2.76    0.85
 C-CT        -0.32    1.37    0.00
 H-HC         0.03    1.87   -0.83
 H-HC         0.03    1.87    0.81
 H-HC         0.03    0.36   -0.00
```

When we run this job, the program terminates with the following error message:

```
 Angle bend undefined between atoms    2    1    3
 Angle bend undefined between atoms    5    4    9
 MM function not complete
```

Atoms 2-1-3 correspond to types *HA-CM-HA*, and 5-4-9 correspond to *CT-CM-CT*. Even though the atom types and the sequence are chemically reasonable, there are no entries for these bending terms in the hard-wired tables. The reason is that the Amber force field has been developed specifically for biochemical systems, in which these particular sequences of atom types do not occur. Hence, the parameters have not been defined by Amber. Note that Gaussian only checks if functions are found for all the stretches and bends; torsional and other terms are simply left out of the total potential function if no entries exist in the tables.

Using External Parameter Files

It is possible to include MM function entries within an external file by replacing the entries within the input file with the usual Gaussian include file mechanism: e.g., @oplsaa.prm. Parameter files for various force fields are included within the main Gaussian directory (i.e., *$g16root/g16*). If you want to specify the entire force field via an external file, use an input file like the following:

```
# UFF=SoftOnly

Read-in force field example
```

molecule specification

```
@$g16root/g16/oplsaa.prm
```

Note that the choice of a molecular mechanics keyword is arbitrary as the entire force field depends on the read-in functions and parameters and not on any internal ones (in other words, specifying Amber would yield the same result as UFF).

Wildcards within Function Specifications

Wildcards can be used in function specifications given in the input file. For example, this Amber bend function specifies that all the bends with a central carbon atom of type *CM* will have the same parameter values:

```
HrmBnd1 * CM * 70.0 120.0
```

One could also include a generic entry using wildcards as well as other more specific entries:

```
HrmBnd1 *  CM *  70.0 120.0
HrmBnd1 HA CM HA 40.0 120.0
```

By default, the program uses the most specific entry that applies to a specific structure (here, a bond angle centered on a *CM* carbon). Relative specificity is determined by the number of wildcards within an entry. If there happen to be different entries that are equally specific, the program terminates with a multiple matching entries error message. In such cases, you can direct the program to use either the first or last match with the FirstEquiv or LastEquiv options, respectively. Be aware, however, that the multiple matching entries message typically indicates some inconsistency in the soft parameter set which needs to be investigated and resolved.

Note that wildcards/entry specificity do not affect whether an entry from the hard-wired or soft set is used. Thus, when SoftFirst is specified, an entry in the input file with three wildcards will be used even if there is an entry containing the exact atom type sequence in question within the hard-wired set.

Substructures: Using Bond Orders and Other Structural Features to Select Parameters

Substructures provide a mechanism for using additional structural information like bond orders and bond angle values to determine which parameters are used. Consider butadiene (illustrated below). The carbon atoms are all of type *CM*, but two bonds are formally double, and one is single. By default, Amber uses the same force constant and equilibrium bond length for the stretching term for each defined atom type combination. However, substructures can be used to assign different values for the different types of bonds.

Substructure numbers are appended to the function name, separated by a hyphen: e.g., *HrmStr-1*, *HrmStr-2*, and so on. In this case, the numbers refer to the bond order (see substructure definitions below). Sometimes, multiple substructure suffixes will be used (e.g., *AmbTrs-1-2*).

The butadiene calculations input file is shown below the molecular structure:

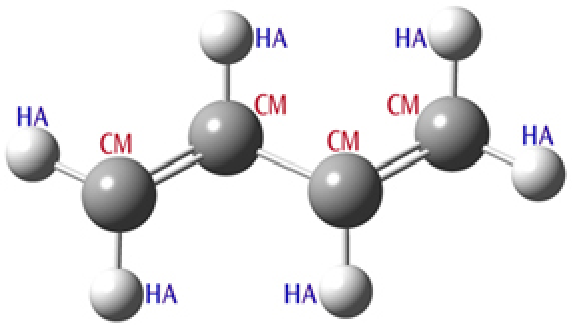


Figure 5.3: Butadiene Molecule

```
# Amber=SoftFirst Geom=Connectivity

Butadiene

0 1
 C-CM    -2.49  -0.07   0.00
 H-HA    -1.96  -1.00   0.00
 H-HA    -3.56  -0.07   0.00
 C-CM    -1.82   1.09   0.00
 H-HA    -2.35   2.02   0.00
 C-CM    -0.28   1.09   0.00
 H-HA     0.24   0.16   0.00
 C-CM     0.39   2.26  -0.00
 H-HA    -0.13   3.19   0.00
 H-HA     1.46   2.26  -0.00

 1 2 1.0 3 1.0 4 2.0
 2
 3
 4 5 1.0 6 1.0
 5
 6 7 1.0 8 2.0
 7
 8 9 1.0 10 1.0
 9
 10
```

```
HrmBnd1 HA CM HA 40.0 120.0    Parameters values for illustration purposes only!
HrmBnd1 CM CM CM 70.0 120.0
HrmStr1-1 CM CM 350.0 1.51     Substructures: bond order-specific stretch parameters.
HrmStr1-2 CM CM 500.0 1.37
```

This example specifies a larger force constant value and a smaller equilibrium bond length for double bonds for use with the *HrmStr1* function.

Note that the values used for the parameters in this example are only for illustration purposes only. In fact, for a production calculation, we would also need to define torsional parameters that are specific for single or double central bonds, using the same substructures mechanism.

Besides the bond order, other examples of substructures are whether the term lies in a cyclic structure, how many hydrogen atoms are connected to it, and the like. Note that multiple substructure suffixes can be used with many functions: e.g., *HrmStr1-1-0-4*.

Substructure slots have relatively consistent meanings in the contexts of different functions. The first substructure suffix defines the bond order or angle range, the second specifies the number of atoms bonded to the current atom, and the third describes the ring environment (if any). A 0 value in any substructure slot acts are a wildcard/placeholder and means that the particular substructure is to be ignored for that entry. Unneeded terminal substructures can be similarly set to 0 or be simply omitted.

The following substructure applies to functions that depend only on the atom type:

Third substructure: ring context (the first and second substructures must be 0):

-0-0-1	None of the following ring contexts apply
-0-0-3	Atom is in a 3-membered ring
-0-0-4	Atom is in a 4-membered ring
-0-0-5	Atom is in a 5-membered ring

The following substructures apply to functions related to bond stretches:

First substructure: bond order

-0	Ignore this substructure (substructure “wildcard”)
-1	Single bond: $0.00 \leq$ bond order < 1.50
-2	Double bond: $1.50 \leq$ bond order < 2.50
-3	Triple bond: bond order ≥ 2.50

Third substructure: ring context (if used, the second substructure must be 0):

-x-0-1	None of the following ring contexts apply
-x-0-3	Bond is in a 3-membered ring
-x-0-4	Bond is in a 4-membered ring
-x-0-5	Bond is in a 5-membered ring

The following substructures apply to functions for bond angles (angle values are in degrees):

First substructure: angle value range

-1	$0^o \leq \theta \leq 45^o$
-2	$45^o < \theta \leq 135^o$

-3	$135^o < \theta \leq 180^o$
Second substructure: numbers of bonded atoms	
-*x*-0	Ignore this substructure (substructure “wildcard”)
-*x*-*n*	*n* is the number of atoms bonded to the central atom
Third substructure: ring context	
-*x*-*y*-0	Ignore this substructure (substructure “wildcard”)
-*x*-*y*-1	None of the following ring contexts apply
-*x*-*y*-3	Bend is in a 3-membered ring
-*x*-*y*-4	Bend is in a 4-membered ring
-*x*-*y*-5	Bend is in a 5-membered ring
Fourth substructure: number of bonded hydrogens	
x-*y*-*z*-*n*	There are *n*+1 hydrogen atoms bonded to the central atom (other than ones comprising the bond angle itself)

The following substructures apply to functions involving dihedral angles.

First substructure: bond order	
-0	Skip this substructure (substructure “wildcard”)
-1	Single central bond: $0.00 \leq$ bond order < 1.50
-2	Double central bond: $1.50 \leq$ bond order < 2.50
-3	Triple central bond: bond order ≥ 2.50
Second substructure: bond type of the central bond	
-x-0	Skip this substructure (substructure “wildcard”)
-x-1	Resonance central bond ($1.30 \leq$ bond order ≤ 1.70)
-x-2	Amide central bond (priority over resonance)
-x-3	Other central bond type
Third substructure: ring context	
-*x*-*y*-1	None of the following ring contexts apply
-*x*-*y*-4	Dihedral is in a 4-membered ring
-*x*-*y*-5	Dihedral is in a 5-membered ring

For example, *HrmStr1-2-0-4* specifies that the *HrmStr1* term applies only to a stretch that involves a double bond that lies in a four-membered ring (compare to *HrmStr1-2* which applies to all double bonds). A substructure value of zero is similar to a wildcard: *HrnStr1-0-0-4* specifies that the term applies to a stretch of any bond order, but it must be in a four-membered ring.

Note that substructures are not always needed. Some functions explicitly include bond orders and similar structural information within their definitions. For example, the *HrmStr3* stretch term from the UFF force field includes the bond order as its third parameter. This parameter can be passed to this function during a UFF calculation for each bond by specifying -1.0 as the value, using a specification like the following in the input file (and the appropriate option to the UFF keyword):

```
HrmStr3 * * -1.0 0.1332
```

Defining New Atom Types

Gaussian understands the standard MM atom types from the Dreiding, UFF, and Amber force fields. However, when a system is studied that these force fields are not parametrized for, none of the available atom types may be suitable. In that case, one can define a new atom type simply by using a new atom type label within the molecule specification. Naturally, since any new atom types will not exist in the hard-wired tables, you will need to also define all required functions using them within the input file.

The Non-Bonded Interaction Master Function

As we've seen, specifying non-bonded interaction terms can be quite lengthy, requiring many *NBTerm* terms. When non-bonded interactions are specified in the input (using one of the options to the MM keyword), they can be specified using the *NonBon* function. This function uses a single entry to define the non-bonded interaction, and it is expanded automatically by the program into the appropriate *NBDir* and *NBTerm* entries.

This function has the following general form:

`NonBon` *V-Type C-Type V-Cutoff C-Cutoff VScale1 VScale2 VScale3 CScale1 CScale2 CScale3*

V-Type and *C-Type* are integers indicating the kind of van der Waals and Coulomb interactions to be used (respectively). The remaining terms are cutoff values and three scale factors for each of the two interaction types; the latter are scale factors for atoms separated by 1, 2 and 3 or more bonds (respectively). When any scale factor is <0, a scale factor of 1.0/|*scale-factor*| is used.

As an example, here is the function used for the standard Amber force field:

```
NonBon 3 1 0 0 0.0 0.0 0.5 0.0 0.0 -1.2
```

This function specifies the Amber arithmetic van der Waals interaction, 1/R Coulomb interaction, and no cut-offs. Van der Waals interactions are scaled by 0.5 for atoms more than 2 bonds distant and are removed for nearer pairs. Coulomb interactions are scaled by 1.0/1.2 for atoms more than 2 bonds distant and are also removed for nearer pairs.

Global Parameters

In some cases, a parameter needed for the potential function is not related to a specific atom type at all. One example is the dielectric constant. For such cases, we use global parameters: defined values which may be used as parameters within potential functions. Any defined global parameters are listed in the MM force field definition section of the output (requested with the Print option). Here is an example definition of the dielectric constant:

```
Dielc  78.39
```

Global parameters also can be specified in vector or matrix format. The following extract from the MM2 force field definition illustrates the use of a matrix in functions which specify the shift of the hydrogen (i.e., interatomic distance scaling) when calculating van der Waals interactions:

`VDWShf1 * 1 0.0`	*By default, use index value 1 with VDWShf2*
`VDWShf1 MM5 2 -1.0`	*Use index value 2 for atom type MM5*
`VDWShf1 MM36 3 -1.0`	*Use index value 3 for atom type MM36*
`VDWShf2 1 2 0.915`	*Scale factor: matrix element (1,2)*
`VDWShf2 1 3 0.915`	*Scale factor: matrix element (1,3)*

We'll consider these two functions in reverse order. *VDWShf2* sets up a lower triangular matrix of scale factors for the interatomic distance. The elements (1,2) and (1,3) are both set to 0.915; the other elements – (1,1), (2,2), (2,3) and (3,3) – are left as the default value, meaning that the distance is unscaled in the van der Waals interaction computation. Note that this formulation is independent of atom type.

The *VDWShf1* function defines which matrix elements to use for various atom types. The first entry specifies the default index value as 1, and the remaining two entries specify index values 2 and 3 for atom types *MM5* and *MM36*. The third parameter to this function specifies the second center involved in the interaction. In the first entry, the 0.0 value functions as a wildcard, while in the other two entries, the value of -1 tells the program to enter the appropriate center automatically.

These functions result in only bonds involving an atom of type *MM5* (H) or *MM36* (D) and another of a different type being shifted. When these functions are evaluated, interactions of type *MM5-X* use element (1,2) and ones of type *MM36-X* use (1,3). Interactions involving other types use (1,1), while *MM5-MM5*, *MM5-MM36* and *MM36-MM36* use (2,2), (2,3) and (3,3), respectively – all of which use unscaled interatomic distances.

Using this mechanism, parameter specification is much clearer and compact. Without it, an entry for every *pair* of atoms would have been needed to achieve the same goal.

Step-Down Tables

Wildcards provide a mechanism for specifying generic parameters/defaults that do not explicitly depend on each of the atom types involved in a particular term. Step-down tables have the same purpose, but are more sophisticated. Consider the following example, which is taken from the UFF force field. In these entries, the *UseTable* and *StepDown* entries are not potential functions and do not contain parameters, but instead describe the step-down mechanism that is used to process entries that do refer to potential functions and contain parameters.

In the following example, the first set of *UFFBnd3* lines define potential functions, while the *UFFBnd2* lines and the second set of *UFFBnd3* lines use the step-down mechanism:

```
UseTable UFFBnd2 #3        Specify step-down table used for specified functions
UseTable UFFBnd3 #3
StepDown UFFBnd2 0 1 0     Matching rules for specified functions
StepDown UFFBnd2 0 2 0
StepDown UFFBnd3 0 1 0
StepDown UFFBnd3 0 2 0
Table #3 C_R  Trig         Define alternate atoms types
Table #3 N_R  Trig
Table #3 H_   Lin
Table #3 Li   Lin
UFFBnd3    0 C_3    0 109.471   -1.0 -1.0 0.1332   Functions defined with
UFFBnd3    0 N_3    0 106.700   -1.0 -1.0 0.1332   explicit atom types
UFFBnd3    0 N_2    0 111.200   -1.0 -1.0 0.1332
UFFBnd3    0 O_3    0 104.510   -1.0 -1.0 0.1332
UFFBnd2-0-2    0 Lin    0 2       -1.0 -1.0 0.1332   Functions defined with step-
UFFBnd2-0-3    0 Trig   0 3       -1.0 -1.0 0.1332   down types & substructures
UFFBnd3-0-3    0 Lin    0 180.0 -1.0 -1.0 0.1332
UFFBnd3-0-4    0 Lin    0 180.0 -1.0 -1.0 0.1332
UFFBnd3-0-5    0 Lin    0 180.0 -1.0 -1.0 0.1332
```

```
UFFBnd3-0-6    0 Lin    0 180.0 -1.0 -1.0 0.1332
UFFBnd3-0-2    0 Trig   0 120.0 -1.0 -1.0 0.1332
UFFBnd3-0-4    0 Trig   0 120.0 -1.0 -1.0 0.1332
UFFBnd3-0-5    0 Trig   0 120.0 -1.0 -1.0 0.1332
UFFBnd3-0-6    0 Trig   0 120.0 -1.0 -1.0 0.1332
```

*H_* and *N_R* are UFF atom types in the UFF force field. If there is a bond angle involving such atoms in the molecule – *H_-N_R-H_* – the program needs to determine the appropriate function and parameters to use. First, it determines the appropriate step-down tables for the available bend function. In this example, table #3 is used for both available bend functions: *UFFBnd2* and *UFFBnd3*.

The *Table* entries define alternate types for specified atom types. Typically, they map many specific atom types to a more generic pseudo-type (e.g., *Lin* and *Trig* in the example). These alternate types allow wide-ranging default values to be specified for function parameters, effectively enabling undefined parameters to be automatically estimated.

The *StepDown* entries specify matching rules for selecting specific function entries. In this example, both bend functions use the same two entries: 0 1 0 and 0 2 0. They specify that first the program should look for an entry specifying the original atom type as the center atom (zero values function as wild cards). If no such entry is present, then an entry specifying the first alternate atom type as the central atom should be used.

In the example, the program would look for bend functions specifying *N_R* as the central atom, but none is defined. Next, it would look for functions with *Trig* as the type of the central atom. There are four such functions: *UFFBnd2-0-3* and *UFFBnd3-0* with second substructures 2, 4 and 5. Which one is used will depend on the number of atoms bonded to the central nitrogen atom.

Note that in practice, step-down tables can specify multiple alternates for the various atom types and also include many more types mapped to each listed alternate.

When a step-down table is in use, then any wildcards within other function specifications are ignored. In other words, of you choose to use a step-down table, then wildcards cannot be used anywhere else.

5.53.6 Force Field Terms

Preliminary Notes

◊ *Units*: Unless otherwise indicated, distances are in Angstroms, angles are in degrees, energies are in kcal/mol and charges are in atomic units.

◊ In equations, R refers to distances and θ refers to angles. Numerical and variable subscripts refer to centers, and "eq" refers to equilibrium: e.g., R_{12} and R_{ij} are the bond distances between atoms 1 and 2 and atoms i and j (respectively), R_i is the bond radii for atom i, R_{eq} and θ_{eq} are the equilibrium bond length and bond angle, and R_{eq12} is the equilibrium bond length between atoms 1 and 2 (typically used in an angle term).

◊ Wildcards may be used in any function definition. They are indicated by a 0 or an asterisk.

◊ Function equivalencies to those found in standard force fields are indicated in parentheses following the description.

VDW : van der Waals parameters, used for NBDir and NBTerm (See MMFF94 below for MMFF94-type van der Waals parameters)

`VDW` *Atom-type Radius Well-depth*

VDW94 : MMFF94 type van der Waals parameters (used for NBDir and NBTerm)

`VDW94` *Atom-type Atomic-pol NE Scale1 Scale2 DFlag*

Atomic-pol	Atomic polarizability (Angstrom^3).
NE	Slater-Kirkwood effective number of valence electrons (dimensionless).
Scale1	Scale factor (Angstrom$^{1/4}$).
Scale2	Scale factor (dimensionless).
DFlag	1.0 for donor type atom, 2.0 for acceptor type, otherwise 0.0.

Buf94 : MMFF94 electrostatic buffering

`Buf94` *Atom-type Value*

NonBon : Non-bonded interaction master function. This function will be expanded into pairs and a direct function (NBDir and NBTerm) before evaluation of the MM energy.

`NonBon` *V-Type C-Type, V-Cutoff C-Cutoff VScale1 VScale2 VScale3 CScale1 CScale2 CScale3*

V-Type is the van der Waals type:

0	No van der Waals
1	Arithmetic (as for Dreiding)
2	Geometric (as for UFF)
3	Arithmetic (as for Amber)
4	MMFF94-type van der Waals
5	MM2-type van der Waals
6	OPLS-type van der Waals

C-Type is the Coulomb type:

0	No Coulomb
1	1/R
2	1/R^2
3	1/R buffered (MMFF94)
10	Form dipole-dipole interactions without cutoffs

V-Cutoff and *C-Cutoff* are the van der Waals and Coulomb cutoffs (respectively):

0	No cutoff
>0	Hard cutoff
<0	Soft cutoff

Vscale1-3 are van der Waals scale factors for 1- to 3-bond separated pairs. *CScale1-3* are Coulomb scale factors for 1- to 3-bond separated pairs. If any scale factor < 0.0, then 1.0/|*scale-factor*| scaling is used (e.g., a scale factor specified as -1.2 results in scaling of 1.0/1.2).

NBDir : Coulomb and van der Waals direct (evaluated for all atom pairs)

`NBDir` *V-Type C-Type V-Cutoff C-Cutoff*

V-Type, *C-Type*, *V-Cutoff*, and *C-Cutoff* as above.

NBTerm : Coulomb and van der Waals single term cutoffs

`NBTerm` *Atom-type1 Atom-type2 V-Type C-Type V-Cutoff C-Cutoff V-Scale C-Scale*

V-Type, *C-Type*, *V-Cutoff*, *C-Cutoff*, *V-Scale*, and *C-Scale* as above.

AtRad : Atomic single bond radius

`AtRad` *Atom-type Radius*

EffChg : Effective charge (UFF)

`EffChg` *Atom-type Charge*

EleNeg : GMP Electronegativity (UFF)

`EleNeg` *Atom-type Value*

Table : Step down table

`Table` *Original-atom-type Stepping-down-type(s)*

HrmStr1 : Harmonic stretch I (Amber 1): $ForceC * (R - R_{eq})^2$

`HrmStr1` *Atom-type1 Atom-type2 ForceC* R_{eq}

ForceC	Force constant
R_{eq}	Equilibrium bond length

HrmStr2 : Harmonic stretch II (Dreiding 4a): $ForceC * [R - (R_i + R_j - Delta)]^2$

`HrmStr2` *Atom-type1 Atom-type2 ForceC Delta*

ForceC	Force constant
Delta	Delta

R_i and R_j are atomic bond radii specified with AtRad.

HrmStr3 : Harmonic stretch III (UFF 1a): $k * (R - R_{ij})^2$

Equilibrium bond length $R_{ij} = (1 - PropC * lnBO) * (R_i + R_j) - R_{en}$

Force constant $k = 664.12 * Z_i * Z_j / (R_{ij}^3)$

Electronegativity correction $R_{en} = R_i * R_j * [SQRT(X_i) - SQRT(X_j)]^2 / (X_i * R_i + X_j * R_j)$

`HrmStr3` *Atom-type1 Atom-type2 BO PropC*

BO	Bond order (if <0, it is determined on-the-fly)
PropC	Proportionality constant

R_i and R_j are atomic bond radii defined with AtRad. X_i and X_j are GMP electronegativity values defined with EleNeg. Z_i and Z_j are the effective atomic charges defined with EffChg.

MrsStr1 : Morse stretch I (Amber): $DLim * (e^{-a(R-R_{eq})} - 1)^2$ where a = SQRT(*ForceC*/*DLim*)

`MrsStr1` *Atom-type1 Atom-type2 ForceC* R_{eq} *DLim*

ForceC	Force constant
R_{eq}	Equilibrium bond length
DLim	Dissociation limit

MrsStr2 : Morse stretch II (Dreiding 5a): $DLim * (e^{-a(R_i+R_j-Delta)} - 1)^2$ where a = SQRT(*ForceC*/*DLim*)

`MrsStr2` *Atom-type1 Atom-type2 ForceC Delta DLim*

ForceC	Force constant
Delta	Delta

DLim	Dissociation limit

R_i and R_j are atomic bond radii defined with AtRad.

MrsStr3 : Morse stretch III (UFF 1b): $BO * DLim * (e^{-a(R-R_{ij})} - 1)^2$ where a = SQRT(k/[*BO***DLim*])

Equilibrium bond length $R_{ij} = (1 - PropC * lnBO) * (R_i + R_j) - R_{en}$

Force constant $k = 664.12 * Z_i * Z_j / (R_{ij}^3)$

Electronegativity correction $R_{en} = R_i * R_j * [SQRT(X_i) - SQRT(X_j)]^2 / (X_i * R_i + X_j * R_j)$

`MrsStr3` *Atom-type1 Atom-type2 BO PropC DLim*

BO	Bond order (if <0, it is determined on-the-fly)
PropC	Proportionality constant
DLim	Dissociation limit

R_i and R_j are atomic bond radii defined with AtRad. X_i and X_j are GMP electronegativity values defined with EleNeg. Z_i and Z_j are the effective atomic charges defined with EffChg.

QStr1 : Quartic stretch I (MMFF94 2): $(R_{eq}/2) * (R - ForceC)^2 * [1 + CStr * (R - ForceC + (7/12) * CStr^2 * (R - ForceC)^2]$

`QStr1` *Atom-type1 Atom-type2 ForceC* R_{eq} *CStr*

ForceC	Force constant (md-Angstrom^{-1})
Req	Equilibrium bond length
CStr	Cubic stretch constant (Angstrom^{-1})

UFFVOx : Atomic torsional barrier for the oxygen column (UFF 16)

`UFFVOx` *Atom-type Barrier*

UFFsp3 : Atomic SP3 torsional barrier (UFF 16)

`UFFVsp3` *Atom-type Barrier*

UFFVsp2 : Atomic SP2 torsional barrier (UFF 17)

`UFFVsp2` *Atom-type Barrier*

HrmBnd1 : Harmonic bend (Amber 1): $ForceC * (\theta - \theta_{eq})^2$

`HrmBnd1` *Atom-type1 Atom-type2 Atom-type3 ForceC* θ_{eq}

ForceC	Force constant (in kcal/(mol*rad^2))
θ_{eq}	Equilibrium angle

HrmBdn2 : Harmonic Bend (Dreiding 10a): $[ForceC/\sin(\theta_{eq}^2)] * (\cos(\theta) - \cos(\theta_{eq}))^2$

`HrmBnd2` *Atom-type1 Atom-type2 Atom-type3 ForceC* θ_{eq}

ForceC	Force constant
θ_{eq}	Equilibrium angle

LinBnd1 : Dreiding Linear Bend (Dreiding 10c): $ForceC * (1 + \cos(\theta))$

`LinBnd1` *Atom-type1 Atom-type2 Atom-type3 ForceC*

ForceC	Force constant

UFFBnd3 : UFF 3-term bend (UFF 11): $k*(C0+C1*\cos(\theta))+C2*\cos(2\theta)$ where

$C2 = 1/(4*\sin(\theta_{eq}^2))$,

$C1 = -4*C2*\cos(\theta_{eq})$ and

$C0 = C2*(2*\cos(\theta_{eq}^2)+1)$

Force constant: $k = 664.12*Z_i*Z_k*(3*R_{ij}*R_{jk}*(1-\cos(\theta_{eq}^2))-\cos(\theta_{eq})*R_{ik}^2)/R_{ik}^5$

`UFFBnd3` *Atom-type1 Atom-type2 Atom-type3* θ_{eq} BO_{12} BO_{23} *PropC*

θ_{eq}	Equilibrium angle
BO12	Bond order for *Atom-type1* - *Atom-type2* (when <0, it is determined on-the-fly)
BO23	Bond order for *Atom-type2* - *Atom-type3* (when <0, it is determined on-the-fly)
PropC	Proportionality constant

R_i, R_j and R_k are atomic bond radii defined with AtRad. X_i, X_j and X_k are GMP electronegativity defined with EleNeg. Z_i, Z_j and Z_k are effective atomic charges defined with EffChg.

UFFBnd2 : UFF 2-term bend (UFF 10): $[k/(Per^2)]*[1-\cos(Per*\theta)]$

Force constant: $k = 664.12*Z_i*Z_k*(3*R_{ij}*R_{jk}*(1-\cos(Per^2))-\cos(Per)*R_{ik}^2)/R_{ik}^5$

`UFFBnd2` *Atom-type1 Atom-type2 Atom-type3 Per* BO_{12} BO_{23} *PropC*

Per	Periodicity: 2 for linear, 3 for trigonal, 4 for square-planar.
BO_{12}	Bond order for *Atom-type1* - *Atom-type2* (when <0, it is determined on-the-fly)
BO_{23}	Bond order for *Atom-type2* - *Atom-type3* (when <0, it is determined on-the-fly)
PropC	Proportionality constant

R_i, R_j and R_k are atomic bond radii defined with AtRad. X_i, X_j and X_k are GMP electronegativity defined with EleNeg. Z_i, Z_j and Z_k are effective atomic charges defined with EffChg.

ZeroBnd : Zero bend term: used in rare cases where a bend is zero. This term is needed for the program not to protest about undefined angles.

`ZeroBnd` *Atom-type1 Atom-type2 Atom-type3*

CubBnd : Cubic bend (MMFF94 3): $(ForceC/2)*(1+CBend*(\theta-\theta_{eq}))*(\theta-\theta_{eq})^2$

`CubBnd` *Atom-type1 Atom-type2 Atom-type3 ForceC* θ_{eq} *CBend*

ForceC	Force constant (in md*Angstrom/rad^2)
θ_{eq}	Equilibrium angle
CBend	Cubic Bend constant (in deg^{-1})

LinBnd2 : MMFF94 Linear Bend (MMFF94 4): $ForceC*(1+\cos(\theta))$

`LinBnd2` *Atom-type1 Atom-type2 Atom-type3 ForceC*

ForceC	Force constant (md)

AmbTrs : Amber torsion (Amber 1): $\sum_{i=1,4}(Mag_i * [1+\cos(i*\theta - PO_i)])/NPaths$

`AmbTrs` *Atom-type1 A-type2 A-type3 A-type4* PO_1 PO_2 PO_3 PO_4 Mag_1 Mag_2 Mag_3 Mag_4 *NPaths*

PO_1–PO_4	Phase offsets
Mag_1–Mag_4	V/2 magnitudes
NPaths	Number of paths. When zero or less, determined on-the-fly.

DreiTrs : Dreiding torsion (Dreiding 13): $V * [1-\cos(Period*(\theta - PO))]/(2*NPaths)$

`DreiTrs` *Atom-type1 Atom-type2 Atom-type3 Atom-type4 V PO Period NPaths*

V	Barrier height
PO	Phase offset
Period	Periodicity
NPaths	Number of paths. When zero or less, determined on-the-fly.

UFFTorC : UFF torsion with constant barrier height (UFF 15): $[V/2]*[1-\cos(Period*PO)*\cos(V*\theta)]/NPaths$

`UFFTorC` *Atom-type1 Atom-type2 Atom-type3 Atom-type4 Period PO V NPaths*

Period	Periodicity
PO	Phase offset
V	Barrier height
NPaths	Number of paths. When zero or less, determined on-the-fly.

UFFTorB : UFF torsion with bond order based barrier height (UFF 17): $[V/2]*[1-\cos(Period*PO)*\cos(Period*\theta)]/NPaths$ where $V = 5*SQRT(U_j*U_k)*[1+4.18*Log(BO_{12})]$

`UFFTorB` *Atom-type1 Atom-type2 Atom-type3 Atom-type4 Period PO* BO_{12} *NPaths*

Period	Periodicity
PO	Phase offset
BO_{12}	Bond order for *Atom-type1* – *Atom-type2* (when <0, it is determined on-the-fly)
NPaths	Number of paths. When zero or less, determined on-the-fly.

U_j and U_k are atomic constants defined with UFFVsp2.

UFFTor1 : UFF torsion with atom type-based barrier height (UFF 16): $[V/2]*[1-\cos(Period*PO)*\cos(Period*\theta)]/NPaths$ where $V = SQRT(V_j*V_k)$

`UFFTor1` *Atom-type1 Atom-type2 Atom-type3 Atom-type4 Period PO NPaths*

Period	Periodicity
PO	Phase offset
NPaths	Number of paths. When zero or less, determined on-the-fly.

V_j and V_k are atomic constants defined with UFFVsp3.

UFFTor2 : UFF torsion with atom type based barrier height (UFF 16) (differs from UFFTor1 in the atomic parameter that is used): $[V/2]*[1-\cos(Period*PO)*\cos(Period*\theta)]/NPAths$ where $V = SQRT(V_j * V_k)$

`UFFTor2` *Atom-type1 Atom-type2 Atom-type3 Atom-type4 Period PO NPaths*

Period	Periodicity
PO	Phase offset
NPaths	Number of paths. When zero or less, determined on-the-fly.

V_j and V_k are atomic constants defined with UFFVOx.

VDWDreiTRS : Dreiding special torsion for compatibility with Gaussian 98 code. During processing, it is replaced with DreiTRS, with the following parameters:

- ◊ If there are three atoms bonded to the third center and the fourth center is H, it is removed.
- ◊ If there are three atoms bonded to the third center, and at least one of them is H, but the fourth center is not H, then these values are used: *V*=4.0, *PO*=0.0, *Period*=3.0, and *NPaths*=-1.0.
- ◊ Otherwise, these values are used: *V*=1.0, *PO*=0.0, *Period*=6.0, and *NPaths*=-1.0.

`OldTor` *Atom-type1 Atom-type2 Atom-type3 Atom-type4*

ImpTrs : Improper torsion (Amber 1): $Mag*[1+\cos(Period*(\theta-PO))]$

`ImpTrs` *Atom-type1 Atom-type2 Atom-type3 Atom-type4 Mag PO Period*

Mag	V/2 Magnitude
PO	Phase offset
Period	Periodicity

Wilson : Three term Wilson angle (Dreiding 28c, UFF 19): $ForceC*(C1+C2*\cos(\theta)+C3*\cos(2\theta))$ averaged over all three Wilson angles θ

`Wilson` *Atom-type1 Atom-type2 Atom-type3 Atom-type4 ForceC C1 C2 C3*

ForceC	Force constant
C1, *C2*, *C3*	Coefficients

HrmWil1 : Harmonic Wilson angle I (MMFF94 6): $(ForceC/2)*(\theta^2)$ summed over all 3 Wilson angles θ

`HrmWil1` *Atom-type1 Atom-type2 Atom-type3 Atom-type4 ForceC*

ForceC	Force constant

MM2Wil : MM2 Wilson sixth order bend (MM2): $\sum_{i=1,2,3}(ForceC_i/2)*(\theta_i^2)*[1+6Bend*(\theta_i^4)]$

`MM2Wil` *Atom-type1 Atom-type2 Atom-type3 Atom-type4 ForceC1 ForceC2 ForceC3 6Bend*

ForceC1 – ForceC3	Force constants
6Bend	Sixth order bend constant (in $\deg^{-4}$)

StrBnd : Stretch-bend (MMFF94 5): $(ForceC1*(R_{12}-R_{eq12})+ForceC2*(R_{23}-R_{eq23}))*(\theta-\theta_{eq})$

`StrBnd` *Atom-type1 Atom-type2 Atom-type3 ForceC1 ForceC2* R_{eq12} R_{eq23} θ_{eq}

ForceC1, *ForceC2*	Force constants

R_{eq12}, R_{eq23}	Equilibrium bond lengths (if <0, retrieved from the appropriate stretch terms)
θ_{eq}	Equilibrium angle

CubStr1 : Cubic Stretch 1 (MM2): $(ForceC/2)*(1-CStr*(R-R_{eq}))*(R-R_{eq})^2$

`CubStr1` *Atom-type1 Atom-type2 ForceC* R_{eq} *CStr*

ForceC	Force constant
R_{eq}	Equilibrium bond length
CStr	Cubic stretch constant (Angstrom^{-1})

SixBnd1 : Sixth order bend (MM2): $(ForceC/2)*(\theta-\theta_{eq})^2(1+6Bend*(\theta-\theta_{eq})^4)$

`SixBnd1` *Atom-type1 Atom-type2 Atom-type3 ForceC* θ_{eq} *6Bend*

ForceC	Force constant
θ_{eq}	Equilibrium angle
6Bend	Sixth order bend constant (in deg^{-4})

MM2Tors : MM2 Torsional (MM2): $En1/2(1+\cos\theta)+En2/2(1-\cos 2\theta)+En3/2(1+\cos 3\theta)$

`MM2Tors` *Atom-type1 Atom-type2 Atom-type3 Atom-type4 En1 En2 En3*

En1 – En3	Energies

MM2VDW : Parameters for the MM2 van der Waals function (MM2)

`MM2VDW` *Par1 Par1 Par3 Par4 Par5*

Par1 – Par5	Parameters

VDWX : Matrix with van der Waals parameters (MM2). These values do not depend on atom types, and can be used as alternatives for the atomic parameters, as needed in the MM2 force field (see discussion above).

`VDWX` *Index1 Index2 Radius Well-depth IsDef*

IsDef	Indicates whether the pair (1.0) has or (0.0) has not been defined.

VDWI : Atomic indices, used to refer to the matrix in VDWXM (MM2)

`VDWI` *Atom-type Index*

VDWShf2 : Matrix with parameters for atom shifting in the van der Waals calculation (MM2)

`VDWShf2` *Index1 Index2 Shift*

VDWShf1 : Atomic indices into the matrix in VDWShf2 (MM2)

`VDWShf1` *Atom-type Index BondTo*

BondTo	Center bonded to (if -1.0, determined by program).

SixBndI : Sixth order bend (MM2): $(ForceC/2)*(\theta-\theta_{eq})^2(1+6Bend*(\theta-\theta_{eq})^4)$

`SixBndI` *Atom-type1 Atom-type2 Atom-type3 ForceC* θ_{eq} *6Bend Flag*

ForceC	Force constant
θ_{eq}	Equilibrium angle
6Bend	Sixth order bend constant (in deg^{-4})
Flag	If ≥ 0.0, suppress projection onto plane even when the central atom is trigonal.

This bend is projected onto the plane in case of a trigonal center. If the center is not trigonal, the regular bend is calculated.

Dipole : Bond dipole

`Dipole` *Atom-type1 Atom-type2 DMom DPos*

DMom	Dipole moment (in Debye)
DPos	Position along the Atom1-Atom2 bond of the dipole

DielC : Dielectric constant. This allows the dielectric constant to be specified via the parameter file; the default is 1.

`DielC` *DielConst*

DielConst	Dielectric constant

QStr2 : Quartic stretch 2 (MM2/Tinker): $(ForceC/2)*(1+CStr*(R-R_{eq})+QStr*(R-R_{eq})^2)(R-R_{eq})^2$

`QStr2` *Atom-type1 Atom-type2 ForceC* R_{eq} *CStr QStr*

ForceC	Force constant
R_{eq}	Equilibrium bond length
CStr	Cubic stretch constant (Angstrom^{-1})
QStr	Quartic stretch constant (Angstrom^{-2})

CubStr2 : Cubic stretch 2 (for compatibility with old MMVB): $(ForceC/2)*(1-CStr*(R-R_{eq})*(R-R_{eq})^2$

`CubStr2` *Atom-type1 Atom-type2 ForceC* R_{eq} *CStr*

ForceC	Force constant
R_{eq}	Equilibrium bond length
CStr	Cubic stretch constant (Angstrom^{-1}; set to 0 when R > 15 Å)

NBonds : Formal number of bonds, based on atom type for this center

`NBonds` *Atom-type NumBnd*

NumBnd	Formal number of bonds

StrUnit : Units for the force constant in stretching functions

`StrUnit` *Flag*

Flag	0 means units of kcal/(mol*Angstrom^2); 1 means units of md/Angstrom

BndUnit : Units for the force constant in bending functions

`BndUnit` *Flag*

Flag	0 means units of kcal/(mol*rad^2); 1 means units of md*Angstrom/rad^2

TorUnit : Units for the barrier in torsion functions

`TorUnit` *Flag*

Flag	0 means units of kcal/mol; 1 means units of md*Angstrom

OOPUnit : Units for the force constant in out-of-plane bending functions

`OOPUnit` *Flag*

Flag	0 means units of kcal/(mol*rad^2); 1 means units of md*Angstrom/rad^2

SBUnit : Units for the force constant in stretch-bend functions

`SBUnit` *Flag*

Flag	0 means units of kcal/(mol*Angstrom*rad); 1 means units of md/rad

StrFact : Factor for the stretching functions

`StrFact` *Value*

BndFact : Factor for the bending functions

`BndFact` *Value*

TorFact : Factor for the torsion functions

`TorFact` *Value*

OOPFact : Factor for the out-of-plane functions

`OOPFact` *Value*

SBFact : Factor for the stretch-bend functions

`SBFact` *Value*

5.54 MP Methods

The MP*n* method keywords request a Hartree-Fock calculation (by default, RHF for singlets, UHF for higher multiplicities) followed by a Møller-Plesset correlation energy correction [531]:

◊ MP2: The Møller-Plesset expansion is truncated at second-order [532–536].

◊ MP3: Third-order MP theory correction [198, 537].

◊ MP4: Fourth-order MP theory correction [538], which defaults to full MP4 with single, double, triple and quadruple substitutions [538, 539] (MP4(SDTQ)).

◊ MP4(DQ): Include only the space of double and quadruple substitutions in the MP expansion.

◊ MP4(SDQ): Include only single, double and quadruple substitutions.

◊ MP5: Fifth-order MP theory correction [134]. The MP5 code has been written for the open-shell case only, and so specifying MP5 defaults to a UMP5 calculation. This method requires O^3V^3 disk storage and scales as O^4V^4 in cpu time.

Analytic gradients are available for MP2 [227, 532, 533, 540], MP3 and MP4(SDQ) [541, 542], and analytic frequencies are available for MP2 [536]. ROMP2, ROMP3 and ROMP4 energies are also available [543–545].

Double-Hybrid Methods

Gaussian 16 also includes some double hybrid methods that combine exact HF exchange with an MP2-like correlation to a DFT calculation. These methods have the same computational cost as MP2 (rather than that of DFT). Gaussian 16 includes:

◊ Grimme's B2PLYP [546] and mPW2PLYP [547] methods; the empirical dispersion corrected variations are specified by appending a D to the keyword name: e.g., B2PLYPD for B2PLYP with empirical dispersion [548].

◊ B2PLYPD3 requests the B2PLYP method combined with Grimme's D3BJ dispersion [324, 332].

◊ DSDPBEP86 [549], a dispersion-corrected double hybrid functional.

◊ The PBE0DH [550] and PBEQIDH [551] double-hybrid functionals.

5.54.1 Options

Frozen Core Options

FC

All frozen core options are available with this keyword; a frozen core calculation is the default. See the discussion of the FC options for full information.

Algorithm Selection Options for MP2 and Double Hybrid Methods

The appropriate algorithm for MP2 will be selected automatically based on the settings of %Mem and MaxDisk. Thus, the following options are almost never needed.

FullDirect

Forces the fully direct algorithm, which requires no external storage beyond that for the SCF. Requires a minimum of $2OVN$ words of main memory (O=number of occupied orbitals, V=number of virtual orbitals, N=number of basis functions). This is seldom a good choice, except for machines with very large main memory and limited disk.

TWInCore

Whether to store amplitudes and products in memory during higher-order post-SCF calculations. The default is to store these if possible, but to run off disk if memory is insufficient. TWInCore causes the program to terminate if these can not be held in memory, while NoTWInCore prohibits in-memory storage.

SemiDirect

Forces the semi-direct algorithm.

Direct

Requests some sort of direct algorithm. The choice between in-core, fully direct and semidirect is made by the program based on memory and disk limits and the dimensions of the problem.

InCore

Forces the in-memory algorithm. This is very fast when it can be used, but requires $N^4/4$ words of memory. It is normally used in conjunction with SCF=InCore. NoInCore prevents the use of the in-core algorithm.

5.54.2 Availability

MP2, B2PLYP methods, mPW2PLYP methods: Energies, analytic gradients, and analytic frequencies.

MP3, MP4(DQ) and MP4(SDQ): Energies, analytic gradients, and numerical frequencies.

MP4(SDTQ) and MP5: Analytic energies, numerical gradients, and numerical frequencies.

RO may be combined with MP2, MP3 and MP4 for energies only.

5.54.3 Related Keywords

HF, SCF, Transformation, MaxDisk

5.54.4 Examples

The MP2 energy appears in the output as follows, labeled as `EUMP2`:

```
E2=         -.3906492545D-01 EUMP2=        -.75003727493390D+02
```

Here is the output from an MP4(SDTQ) calculation:

```
Time for triples=         .04 seconds.
MP4(T)=     -.55601167D-04
E3=         -.10847902D-01         EUMP3=       -.75014575395D+02
E4(DQ)=     -.32068082D-02         UMP4(DQ)=    -.75017782203D+02
E4(SDQ)=    -.33238377D-02         UMP4(SDQ)=   -.75017899233D+02
E4(SDTQ)=   -.33794389D-02         UMP4(SDTQ)=  -.75017954834D+02
```

The energy labeled EUMP3 is the MP3 energy, and the various MP4-level corrections appear after it, with the MP4(SDTQ) value coming in the final line.

The B2PLYP energy appears as follows in the output:

```
E2(B2PLYP) =    -0.3262340664D-01 E(B2PLYP) =    -0.39113226645200D+02
```

5.55 Name

This keyword specifies the username that is stored in the archive entry for the calculation. It takes the desired username as its parameter (e.g., Name=RChavez). On UNIX systems, the default for the username is the operating system-level login name of the user who runs the job.

5.55.1 Related Keywords

Test

5.56 NMR

This properties keyword predicts NMR shielding tensors and magnetic susceptibilities using the Hartree-Fock method, all DFT methods and the MP2 method [552–555].

NMR shielding tensors may be computed with the Continuous Set of Gauge Transformations (CSGT) method [555–557] and the Gauge-Independent Atomic Orbital (GIAO) method [215, 555, 558–560]. Magnetic susceptibilities may also be computed with both GIAOs [561] and CGST. Gaussian also supports the IGAIM method [556, 557] (a slight variation on the CSGT method) and the Single Origin method, for both shielding tensor and magnetic susceptibilities.

Structures used for NMR calculations should have been optimized at a good level of theory. Note that CSGT calculations require large basis sets to achieve accurate results.

Spin-spin coupling constants may also be computed during an NMR job [562–566], via the SpinSpin option.

5.56.1 Options

SpinSpin

Compute spin-spin coupling constants in addition to the usual NMR properties. Be aware that this calculation type has a computational cost of about twice that of computing vibrational frequencies alone. It is available only for Hartree-Fock and DFT methods.

Mixed

Requests a two-step spin-spin coupling calculation [566]. This option causes two job steps to be run. In the first, the basis set specified by the user is modified to be appropriate for the Fermi Contact term, by uncontracting the basis and adding tight polarization functions for the core. In the second step, the other three terms in the spin-spin coupling are calculated with the unmodified basis set specified in the route section. The final results reported at the end of the second job step include the Fermi Contact contribution from the first step. This significantly improves the accuracy of spin-spin coupling constants, especially when done with typical valence-oriented basis sets such as 6-311G+(d,p), aug-CC-pVDZ or aug-CC-pVTZ. This approach is also faster than computing all four terms using a modified basis set incorporating tight polarization functions.

ReadAtoms

Calculate spin-spin coupling constants only for selected atoms. The atom list is specified in a separate input section (terminated by a blank line). The list is initially empty.

The input section uses the following format:

atoms=*list* [**notatoms=***list*]

where each *list* is a comma or space-separated list of atom numbers, atom number ranges and/or atom types. Keywords are applied in succession. Here are some examples:

`atoms=3-6,17 notatoms=5`	*Adds atoms 3, 4, 6 and 17 to the atom list. Removes 5 if present.*
`atoms=3 C 18-30 notatoms=H`	*Adds all C atoms and all non-H among atoms 3, 18-30.*
`atoms=C N notatoms=5`	*Adds all C and N atoms except atom 5.*
`atoms=1-5 notatoms=H atoms=8-10`	*Adds atoms 8-10 and non-hydrogens among atoms 1-5.*

Bare integers without a keyword are interpreted as atom numbers:

`1,3,5 7`	*Adds atoms 1, 3, 5 and 7.*

CSGT

Compute NMR properties using the CSGT method only. The data file for the ACID program can be generated with NMR=CSGT IOp(10/93=1).

GIAO

Compute NMR properties using the GIAO method only. This is the default.

IGAIM

Use atomic centers as gauge origins.

SingleOrigin

Use a single gauge origin. This method is provided for comparison purposes but is not generally recommended.

All

Compute properties with all three of the SingleOrigin, IGAIM, and CSGT methods.

PrintEigenvectors

Display the eigenvectors of the shielding tensor for each atom.

FCOnly

Compute only the Fermi contact spin-spin terms.

ReadFC

Read the Fermi contact spin-spin terms from the checkpoint file and then compute the other spin-spin coupling terms.

Susceptibility

Compute the magnetic susceptibility as well as the shielding.

5.56.2 Availability

SCF, DFT and MP2 methods. NMR may be combined with SCRF. NMR and Freq can now both be on the same route for HF and DFT.

5.56.3 Examples

Here is an example of the default output from NMR:

```
Magnetic properties (GIAO method)

Magnetic shielding (ppm):
  1  C    Isotropic =    57.7345   Anisotropy =   194.4092
   XX=    48.4143   YX=      .0000   ZX=      .0000
   XY=      .0000   YY=   -62.5514   ZY=      .0000
   XZ=      .0000   YZ=      .0000   ZZ=   187.3406
  2  H    Isotropic =    23.9397   Anisotropy =     5.2745
   XX=    27.3287   YX=      .0000   ZX=      .0000
   XY=      .0000   YY=    24.0670   ZY=      .0000
   XZ=      .0000   YZ=      .0000   ZZ=    20.4233
```

For this molecular system, the values for all of the atoms of a given type are equal, so we have truncated the output after the first two atoms.

The additional output from spin-spin coupling computations appears as follows:

```
 Total nuclear spin-spin coupling K (Hz):
                 1              2
      1  0.000000D+00
      2  0.147308D+02  0.000000D+00
 Total nuclear spin-spin coupling J (Hz):
                 1              2
      1  0.000000D+00
      2  0.432614D+03  0.000000D+00
```

The various components of the coupling constants precede this section in the output file. It displays the matrix of isotropic spin-spin coupling between pairs of atoms in lower triangular form. The K matrix gives the values which are isotope-independent, and the J matrix gives the values taking the job's specific isotopes into account (whether explicitly specified or the default isotopes).

5.57 ONIOM

This keyword requests a two- or three-layer ONIOM [567]. See [568–571] for recent developments, and [572–577] for earlier work. In this procedure, the molecular system being studied is divided into two or three layers which are treated with different model chemistries. The results are then automatically combined into the final predicted results. The layers are conventionally known as the Low, Medium and High layers. By default, atoms are placed into the High layer (from a certain point of view, any conventional calculation can be viewed as a one-layer ONIOM). Layer assignments are specified as part of the molecule specification (see below). GaussView provides many graphical tools that simplify setting up ONIOM calculations.

For MO:MM and MO:MO:MM jobs, the ONIOM optimization procedure uses microiterations [578] and an optional quadratic coupled algorithm is also available [568]. The latter (requested with Opt=QuadMacro) takes into account the coupling between atoms in the model system and the atoms only in the MM layer in order to produce more accurate steps than the regular microiterations algorithm [579].

MO:MM and MO:MO:MM calculations can take advantage of electronic embedding (ONIOM=EmbedCharge option). Electronic embedding incorporates the partial charges of the MM region into the quantum mechanical Hamiltonian. This technique provides a better description of the electrostatic interaction between the QM and MM regions (as it is treated at the QM level) and allows the QM wavefunction to be polarized.

There are several relevant considerations for MO:MM and MO:MO:MM optimizations and IRCs (note that none of this is relevant to ONIOM without MM):

◊ The default optimization algorithm uses the RFO algorithm for steps in internal coordinates that include just the model system, along with linear microiterations to optimize the atoms that are only in the real system (i.e., only treated with MM). For minima, the default algorithm is usually the best choice.

◊ For problem convergence cases, the main alternative is Opt=QuadMac, which does a quadratic step in the coordinates of all the atoms. Such an optimization can use either an updated approximate Hessian for the internal coordinates or an analytically computed Hessian (see the next bullet). It computes products of the low-level (MM) Hessian with vectors as needed.

◊ If there are still convergence problems, then try Opt=(QuadMac,CalcFC) or Opt=(QuadMac,CalcAll).

◊ For transition structures, the QuadMac option helps to ensure that you move in the direction of a proper transition structure, so Opt=(QuadMac,TS,CalcFC) is usually a good choice.

◊ For both minima and transition structures, it is usually more efficient to first optimize using mechanical embedding and then perform a second optimization with electronic embedding starting from the resulting structure (rather than trying to use electronic embedding from the beginning).

◊ For IRCs, first run a frequency calculation to confirm that you have a proper transition structure. You can then begin the IRC using that job's force constants via IRC=RCFC. For very large systems, include the Freq=SaveNM option so that IRC=RCFC does not have to recompute normal modes. The default IRC algorithm for MO:MM, IRC=EulerPC, is best for large systems. However, IRC=HPC may be preferable for small to medium-sized cases.

See [580] for an example study employing ONIOM.

ONIOM calculations can also use an external program for the calculations for one or more levels of theory. See the External keyword for details.

5.57.1 Required Input

The two or three desired model chemistries are specified as the options to the ONIOM keyword, in the order High, Medium, Low (the final one may obviously be omitted). The distinct models are separated by colons. For example, this route section specifies a three-layer ONIOM calculation, using UFF for the Low layer, PM6 for the Medium layer, and HF for the High layer:

```
# ONIOM(HF/6-31G(d):PM6:UFF)
```

Atom layer assignment for ONIOM calculations is done as part of the molecule specification, via additional parameters on each line according to the following syntax:

atom [freeze-code] coordinate-spec layer [link-atom [bonded-to [scale-fac1 [scale-fac2 [scale-fac3]]]]]

where *atom* and *coordinate-spec* represent the normal molecule specification input for the atom. The *freeze-code* indicates if/how the atom is to be frozen during a geometry optimization. If the value is omitted or 0, then the atom's coordinates will be optimized. If the value is -1, then the atom will be frozen. If any other negative integer is specified, then the atom is treated as part of a rigid fragment during the optimization; all atoms with the same negative value move together as a rigid block. Blocks can be frozen or activated with Opt=ReadOpt.

Layer is a keyword indicating the layer assignment for the atom, one of High, Medium, and Low. The other optional parameters specify how the atoms located at a layer boundary are to be treated. You use *link-atom* to specify the atom with which to replace the current atom (it can include atom type and partial charge and other parameters). Link atoms are necessary when covalent bonding exists between atoms in different layers in order to saturate the (otherwise) dangling bonds.

Note: All link atoms must be specified by the user. Gaussian 16 does not define them automatically or provide any defaults. GaussView does this automatically, but users still need to consider this general concern and follow the rules in [570].

The *bonded-to* parameter specifies which atom the current atom is to be bonded to during the higher-level calculation portion. If it is omitted, Gaussian will attempt to identify it automatically.

In general, Gaussian 16 determines bond distances between atoms and their link atoms by scaling the original bond distance (i.e., in the real system), using scaling factors which the program determines automatically. However, you can also specify these scale factors explicitly. For a two-layer calculation, the scale factors specify the link atom bond distance in the model system when calculated at the low and high levels (respectively). For a three-layer ONIOM, up to three scale factors may be specified (in the order low, medium, high).

All of these scale factors correspond to the g-factor parameter as defined in reference [567], extended to allow separate values for each ONIOM calculation level.

For a two-layer ONIOM, if only one parameter is specified, then both scale factors will use that value. For a three-layer ONIOM, if only one parameter is specified, then all three scale factors will use that value; if only two parameters are specified, then the third scale factor will use the second value.

If a scale parameter is explicitly set to 0.0, then the program will determine the corresponding scale factor in the normal way. Thus, if you want to change only the second scale factor (model system calculated at the medium level), then you must explicitly set the first scale factor to 0.0. In this case, for a three-layer ONIOM, the third scale factor will have the same value as the second parameter unless it is explicitly assigned a non-zero value (i.e., in this second context, 0.0 has the same meaning as an omitted value).

5.57.2 Per-layer Charge and Spin Multiplicity

Multiple charge and spin multiplicity pairs may also be specified for ONIOM calculations. For two-layer ONIOM jobs, the format for this input line is:

$chrg_{real-low}\ spin_{real-low}\ [chrg_{model-high}\ spin_{model-high}\ [chrg_{model-low}\ spin_{model-low}\ [chrg_{real-high}\ spin_{real-high}]]]$

where the subscript indicates the calculation for which the values will be used. The fourth pair applies only to ONIOM=SValue calculations. When only a single value pair is specified, all levels will use those values. If two pairs of values are included, then the third pair defaults to the same values as in the second pair. If the final pair is omitted for an S-value job, it defaults to the values for the real system at the low level.

Values and defaults for three-layer ONIOM calculations follow an analogous pattern (in the subscripts below, the first character is one of: *Real*, *Int*=Intermediate system, and *Mod*=Model system, and the second character is one of: *H*, *M* and *L* for the High, Medium and Low levels):

$^{c}RealL\ ^{s}RealL\ [^{c}IntM\ ^{s}IntM\ [^{c}IntL\ ^{s}IntL\ [^{c}ModH\ ^{s}ModH\ [^{c}ModM\ ^{s}ModM\ [^{c}ModL\ ^{s}ModL]]]]]$

For 3-layer ONIOM=SValue calculations, up to three additional pairs may be specified:

$\cdots\ ^{c}IntH\ ^{s}IntH\ [^{c}RealM\ ^{s}RealM\ [^{c}RealH\ ^{s}RealH]]$

Defaults for missing charge/spin multiplicity pairs are taken from the next highest calculation level and/or system size. Thus, when only a subset of the six or nine pairs are specified, the charge and spin multiplicity items default according to the following scheme, where the number in each cell indicates which pair of values applies for that calculation in the corresponding circumstances:

Charge & Spin Defaults

	# Pairs Specified					**(SValue only)**			
Calculation	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**
Real-Low	1	1	1	1	1	1	1	1	1
Int-Med	1	2	2	2	2	2	2	2	2
Int-Low	1	2	3	3	3	3	3	3	3
Model-High	1	2	2	4	4	4	4	4	4
Model-Med	1	2	2	4	5	5	5	5	5
Model-Low	1	2	2	4	5	6	6	6	6
Int-High	1	2	2	2	2	2	7	7	7
Real-Med	1	1	1	1	1	1	1	8	8
Real-High	1	1	1	1	1	1	1	8	9

5.57.3 Options

EmbedCharge

Use MM charges from the real system in the QM calculations on the model system(s). NoEmbedCharge is the default.

MKUFF

MKUFF uses the Merz-Kollman-Singh approximate charges during geometry optimization microiterations with electronic embedding but using UFF radii, which are defined for the full periodic table. This is the default.

MK

Specifies that Merz-Kollman-Singh (see Population=MK) approximate charges be used during geometry optimization microiterations with electronic embedding.

Mulliken

Specifies that Mulliken approximate charges be used during geometry optimization microiterations with electronic embedding. (See Population default method.)

HLYGAt

Specifies the Hu, Lu, and Yang (see Population=HLYGAt) charge fitting method be used during geometry optimization microiterations with electronic embedding, but using Gaussian's standard atomic densities instead of those of HLY.

ScaleCharge=*ijklmn*

Specifies scaling parameters for MM charges during electronic embedding in the QM calculations. The integers are multiplied by 0.2 to obtain the actual scale factors. Atoms bonded to the inner layers use a scale factor of $0.2n$, those two bonds away use $0.2m$, and so on. However, the values of i through n must be monotonically decreasing, and the largest value among them is used for all parameters to its left. Thus, 555500, 123500 and 500 are all equivalent. The default value is 500 (i.e., 555500), turning off charges within 2 bonds of the QM region and leaving the rest unscaled. ScaleCharge implies EmbedCharge.

SValue

Requests that the full square be done for testing, to produce substituent values (S-values) for the S-value test [581]. Additional charge and spin multiplicity pair(s) may be specified for the additional calculations (see below).

Compress

Compress operations and storage to active atoms during MO:MM mechanical embedding second derivative calculations; this is the default. NoCompress performs the calculation without compression. Blank does the uncompressed calculation but then discards contributions from inactive atoms (which are currently non-zero only for nuclear moment perturbations: shielding and spin-spin coupling tensors). FullMatrix causes the full Hessian to be stored and used in mechanical embedding Opt=QuadMac, instead of using the molecular mechanics Hessian for the real system in operator form. This is faster for medium-sized systems but uses more disk space for large ones.

InputFiles

Prints out an input file for each intermediate calculation, to facilitate running these calculations separately. OnlyInputFiles just generates the files without doing any calculations.

5.57.4 Availability

Energies, gradients and frequencies. Note that if *any* of the specified models require numerical frequencies, then numerical frequencies will be computed for *all* models, even when analytic frequencies are available.

ONIOM can also perform CIS and TD calculations for one or more layers. The Gen, GenECP and ChkBas keywords may also be specified for relevant models. Density fitting sets may also be used when applicable, and they are specified in the usual manner (see the examples). NMR calculations may be performed with the ONIOM model.

5.57.5 Related Keywords

Geom=Connect, Molecular Mechanics keywords, Opt=QuadMacro, External

5.57.6 Examples

Molecule Specifications for ONIOM Jobs. Here is a simple ONIOM input file:

```
# ONIOM(B3LYP/6-31G(d,p):UFF) Opt

2-layer ONIOM optimization

0 1 0 1 0 1
  F     -1.041506214819     0.000000000000    -2.126109488809 M
  F     -2.033681935634    -1.142892069126    -0.412218766901 M
  F     -2.033681935634     1.142892069126    -0.412218766901 M
  C     -1.299038105677     0.000000000000    -0.750000000000 M H 4
  C      0.000000000000     0.000000000000     0.000000000000 H
  H      0.000000000000     0.000000000000     1.100000000000 H
  O      1.125833024920     0.000000000000    -0.650000000000 H
```

The High layer consists of the final three atoms. The other atoms are placed into the Medium layer. A link atom is defined for the first carbon atom.

Here is an input file for a two-layer ONIOM calculation using a DFT method for the high layer and Amber for the low layer. The molecule specification includes atom types (which are optional with UFF but required by Amber). Note that atom types are used for both the main atom specifications and the link atoms:

```
# ONIOM(B3LYP/6-31G(d):Amber) Geom=Connectivity

2 layer ONIOM job

0 1 0 1 0 1     Charge/spin for entire molecule (real system), model system-high level & model-low
 C-CA--0.25   0   -4.703834   -1.841116   -0.779093 L
 C-CA--0.25   0   -3.331033   -1.841116   -0.779093 L H-HA-0.1   3
 C-CA--0.25   0   -2.609095   -0.615995   -0.779093 H
 C-CA--0.25   0   -3.326965    0.607871   -0.778723 H
 C-CA--0.25   0   -4.748381    0.578498   -0.778569 H
 C-CA--0.25   0   -5.419886   -0.619477   -0.778859 L H-HC-0.1   5
 H-HA-0.1     0   -0.640022   -1.540960   -0.779336 L
 H-HA-0.1     0   -5.264565   -2.787462   -0.779173 L
```

```
...
```

This input also illustrates the use of multiple charge and spin multiplicity values for ONIOM jobs.

A Complex ONIOM Route. Here is an example of a complex ONIOM route section:

```
# ONIOM(BLYP/6-31G(d)/Auto TD=(NStates=8):UFF)
```

This example uses density fitting for the DFT high layer time-dependent excited states calculation.

Freezing Atoms During ONIOM Optimizations. ONIOM optimizations can take advantage of the optional second field within molecule specifications. This field defaults to 0 if omitted. If it is set to -1, then the corresponding atom is frozen during geometry optimizations, e.g.:

```
C -1 0.0 0.0 0.0
H  0 0.0 0.0 0.9
...
```

For ONIOM jobs only, if the field is set to a negative value other than -1, it is treated as part of a rigid fragment during the optimization: all atoms with the same value (< -1) move only as a rigid block.

Handling an SCF convergence issue limited to one layer. For cases where it is difficult to converge the initial SCF or to get it to converge to the lowest solution, the following procedure is helpful. Here is the initial ONIOM input file:

```
%Chk=mychk
# ONIOM(BLYP/3-21G:UFF) Opt Freq
```

input file continues ···

First, create an input file for the high-level model system calculation by running the job and adding the OnlyInputFiles option to ONIOM, which prints input files for each of the 3 individual calculations:

```
# ONIOM(BLYP/3-21G:UFF)=OnlyInputFiles
```

Use the input file for the high-level model calculation to obtain a converged SCF for this system, being sure to save its checkpoint file, called for example *highmod.chk*. Use whatever options are required to get SCF convergence (e.g., Stable=Opt).

Next, run the ONIOM job using Guess=Input:

```
%Chk=mychk
# ONIOM(BLYP/3-21G:UFF) Opt Freq Guess=Input

ONIOM Opt Freq

molecule specification

highmod.chk                    Checkpoint file for Guess=Input
```

When this job computes the initial guess, it reads a line from the input file saying what to do: generate, read or the name of another checkpoint file, the option used here.

The procedure is similar for an MO:MO calculation. However, in this case, there will be three initial guesses performed (since all of the calculations are MO calculations), with one input line read for each guess when you use Guess=Input. For example, if only the high level calculation on the model system needs to be converged separately, then the input would look like this:

```
%chk=mychk
# ONIOM(BLYP/6-31+G*:HF/STO-3G) Opt Freq Guess=Input

2 Layer MO:MO ONIOM Opt Freq

molecule specification
```

`generate`	*Generate initial guess for the low level real system.*
`highmod.chk`	*Read initial guess from this file for the high level model system.*
`generate`	*Generate initial guess for the low level model system.*

S-Value Test. Here is some output from the ONIOM=SValue option:

```
S-Values (between gridpoints) and energies:

 high      4       -39.322207      7       -39.305712      9
     -114.479426           -153.801632           -193.107344
  med      2       -39.118688      5       -39.106289      8
     -114.041481           -153.160170           -192.266459
  low      1       -38.588420      3       -38.577651      6
     -112.341899           -150.930320           -189.507971
              model                  mid                    real
```

The integers are the gridpoints, and under each one is the energy value. Horizontally between the grid points are the S-values. These are the S-values obtained with the absolute energies. However, be aware that when applying the S-value test, relative energies and S-values need to be used (see reference [581]).

5.58 Optimization

This keyword requests that a geometry optimization be performed. The geometry will be adjusted until a stationary point on the potential surface is found. Analytic gradients will be used if available. For the Hartree-Fock, CIS, MP2, MP3, MP4(SDQ), CID, CISD, CCD, CCSD, QCISD, BD, CASSCF, and all DFT and semi-empirical methods, the default algorithm for both minimizations (optimizations to a local minimum) and optimizations to transition states and higher-order saddle points is the Berny algorithm using GEDIIS [582] in redundant internal coordinates [461, 583–587] (corresponding to the Redundant option). An brief overview of the Berny algorithm is provided in the final subsection of this discussion. The default algorithm for all methods lacking analytic gradients is the eigenvalue-following algorithm (Opt=EF).

Gaussian includes the STQN method for locating transition structures. This method, implemented by H. B. Schlegel and coworkers [461, 587], uses a quadratic synchronous transit approach to get closer to the quadratic region of the transition state and then uses a quasi-Newton or eigenvector-following algorithm to complete the optimization. Like the default algorithm for minimizations, it performs optimizations by default in redundant internal coordinates. This method will converge efficiently when provided with an empirical estimate of the Hessian and suitable starting structures.

This method is requested with the QST2 and QST3 options. QST2 requires two molecule specifications, for the reactants and products, as its input, while QST3 requires three molecule specifications: the reactants, the products, and an initial structure for the transition state, in that order. *The order of the atoms must be identical within all molecule specifications.* See the examples for sample input for and output from this method.

Basic information as well as techniques and pitfalls related to geometry optimizations are discussed in detail in chapter 3 of *Exploring Chemistry with Electronic Structure Methods* [152]. For a review article on optimization and related subjects, see [502].

Gaussian 16 supports generalized internal coordinates (GIC), a facility which allows arbitrary redundant internal coordinates to be defined and used for optimization constraints and other purposes [460]. There are several GIC-related options to Opt, and the *GIC Info* subsection describes using GICs as well as their limitations in the present implementation.

5.58.1 Set Opt. Goal

By default, optimizations search for a local minimum.

QST2

Search for a transition structure using the STQN method. This option requires the reactant and product structures as input, specified in two consecutive groups of title and molecule specification sections. Note that the atoms must be specified in the same order in the two structures. TS should *not* be specified with QST2.

QST3

Search for a transition structure using the STQN method. This option requires the reactant, product, and initial TS structures as input, specified in three consecutive groups of title and molecule specification sections. Note that the atoms must be specified in the same order within the three structures. TS should *not* be specified with QST3.

TS

Requests optimization to a transition state rather than a local minimum, using the Berny algorithm.

Saddle=*N*

Requests optimization to a saddle point of order *N* using the Berny algorithm.

Conical

Search for a conical intersection or avoided crossing using the state-averaged CASSCF method. Avoided is a synonym for Conical. Note that CASSCF=SlaterDet is needed in order to locate a conical intersection between a singlet state and a triplet state.

5.58.2 Options

Options to Modify the Initial Geometry

ModRedundant

Except for any case when it is combined with the GIC option (see below), the ModRedundant option will add, delete, or modify redundant internal coordinate definitions (including scan and constraint information) before performing the calculation. This option requires a separate input section following the geometry specification; when used in conjunction with QST2 or QST3, a ModRedundant input section must follow each geometry specification. AddRedundant is synonymous with ModRedundant.

Lines in a ModRedundant input section use the following syntax:

[*Type*] *N*1 [*N*2 [*N*3 [*N*4]]] [A | F]

[*Type*] *N*1 [*N*2 [*N*3 [*N*4]]] B

[*Type*] *N*1 [*N*2 [*N*3 [*N*4]]] K | R

[*Type*] *N*1 [*N*2 [*N*3 [*N*4]]] D

[*Type*] *N*1 [*N*2 [*N*3 [*N*4]]] H *diag-elem*

[*Type*] *N*1 [*N*2 [*N*3 [*N*4]]] S *nsteps stepsize*

*N*1, *N*2, *N*3, and *N*4 are atom numbers or wildcards (discussed below). Atom numbering begins at 1, and any dummy atoms are not counted.

The atom numbers are followed by a one-character code letter indicating the coordinate modification to be performed; the action code is sometimes followed by additional required parameters as indicated above. If no action code is included, the default action is to add the specified coordinate. These are the available action codes:

A	Activate the coordinate for optimization if it has been frozen.
F	Freeze the coordinate in the optimization.
B	Add the coordinate and build all related coordinates.
K	Remove the coordinate and kill all related coordinates containing this coordinate.
R	Remove the coordinate from the definition list (but not the related coordinates).
D	Calculate numerical second derivatives for the row and column of the initial Hessian for this coordinate.
H	Change the diagonal element for this coordinate in the initial Hessian to *diag-elem*.
S	Perform a relaxed potential energy surface scan. Increment the coordinate by *stepsize* a total of *nsteps* times, performing an optimization from each resulting starting geometry.

An asterisk (*) in the place of an atom number indicates a wildcard. Here are some examples of wildcard use:

*	All atoms specified by Cartesian coordinates.
* *	All defined bonds.
3 *	All defined bonds with atom 3.
* * *	All defined valence angles.
* 4 *	All defined valence angles around atom 4.
* * * *	All defined dihedral angles.
* 3 4 *	All defined dihedral angles around the bond connecting atoms 3 and 4.

By default, the coordinate type is determined from the number of atoms specified: Cartesian coordinates for 1 atom, bond stretch for 2 atoms, valence angle for 3 atoms, and dihedral angle for 4 atoms. Optionally, *type* can be used to designate these and additional coordinate types:

X	Cartesian coordinates.

B	Bond length.
A	Valence angle.
D	Dihedral angle.
L	Linear bend specified by three atoms (if *N*4 is -1) or by four atoms, where the fourth atom is used to determine the 2 orthogonal directions of the linear bend.

See the examples for illustrations of the use of ModRedundant.

ReadOptimize

Read an input section modifying which atoms are to be optimized. The atom list is specified in a separate input section (terminated by a blank line). By default, the atom list contains all atoms in the molecule, unless any atoms are designated as frozen within the molecule specification, in which case the initial atom list excludes them. If the structure is being read in from the checkpoint file, then the list of atoms to be optimized matches that in the checkpoint file. ReadOpt and RdOpt are synonyms for this option. ReadFreeze and RdFreeze are deprecated synonyms.

The input section uses the following format:

atoms=*list* [**notatoms=***list*]

where each *list* is a comma or space-separated list of atom numbers, atom number ranges and/or atom types. Keywords are applied in succession. Here are some examples:

`atoms=3-6,17 notatoms=5`	*Adds atoms 3,4,6,17 to atom list. Removes 5 if present.*
`atoms=3 C 18-30 notatoms=H`	*Adds all C & non-H among atoms 3, 18-30.*
`atoms=C N notatoms=5`	*Adds all C and N atoms except atom 5.*
`atoms=1-5 notatoms=H atoms=8-10`	*Adds atoms 8-10 and non-hydrogens among atoms 1-5.*

Bare integers without a keyword are interpreted as atom numbers:

`1,3,5 7`	*Adds atoms 1, 3, 5 and 7.*

For ONIOM optimizations only, block and notblock can be similarly used to include/not include rigid blocks defined in ONIOM molecule specifications. If there are contradictions between atoms specified as atoms and within blocks – e.g., an atom is included within a block but excluded by atom type – Gaussian 16 generates an error.

You can start from an empty atom list by placing noatoms as the first item in the input section. For example, the following input optimizes all non-hydrogen atoms within atoms 1-100 and freezes all other atoms in the molecule:

```
noatoms atoms=1-100 notatoms=H
```

Atoms can also be specified by ONIOM layer via the [not]layer keywords, which accept these values: real for the real system, model for the model system in a 2-layer ONIOM, middle for the middle layer in a 3-layer ONIOM, and small for the model layer of a 3-layer ONIOM. Atoms may be similarly included/excluded by residue with residue and notresidue, which accept lists of residue names. Both keyword pairs function as shorthand forms for atom lists.

Separate sections are read for each geometry for transition state optimizations using QST2 or QST3. Be aware that providing contradictory input – e.g., different frozen atoms for the reactants and products –

will produce unpredictable results.

NoFreeze

Activates (unfreezes) all variables, in other words freeze nothing and optimize all atoms. This option is useful when reading in a structure from a checkpoint file that contains frozen atoms (i.e. with Geom=Check). This option should not be used with GICs; use UnFreezeAll in the GIC input section instead.

General Procedural Options

MaxCycles=*N*

Sets the maximum number of optimization steps to *N*. The default is the maximum of 20 and twice the number of redundant internal coordinates in use (for the default procedure) or twice the number of variables to be optimized (for other procedures).

MaxStep=*N*

Sets the maximum size for an optimization step (the initial trust radius) to 0.01*N* Bohr or radians. The default value for *N* is 30.

Restart

Restarts a geometry optimization from the checkpoint file. In this case, the entire route section will consist of the Opt keyword and the same options to it as specified for the original job (along with Restart). No other input is needed (see the examples).

InitialHarmonic=*N*

Add harmonic constraints to the initial structure with force constant *N*/1000000 Hartree/Bohr^2. IHarmonic is a synonym for this option.

ChkHarmonic=*N*

Add harmonic constraints to the initial structure saved on the chk file with force constant *N*/1000000 Hartree/Bohr^2. CHarmonic is a synonym for this option.

ReadHarmonic=*N*

Add harmonic constraints to a structure read in the input stream (in the input orientation), with force constant *N*/1000000 Hartree/Bohr^2. RHarmonic is a synonym for this option.

MaxMicroiterations=*N*

Allow up to *N* microiterations. The default is based on *NAtoms* but is at least 5000. MaxMicro is a synonym for this option.

NGoDown=*N*

Mix at most *N* Hessian eigenvectors having negative eigenvalues when trying to go downhill. Defaults to 3. If *N*=-1, do regular RFO step only. NoDownHill is equivalent to NGoDown = -1.

Options Related to Initial Force Constants

Unless you specify otherwise, a Berny geometry optimization starts with an initial guess for the second derivative matrix – also known as the Hessian – which is determined using connectivity derived from atomic radii and a simple valence force field [461, 588]. The approximate matrix is improved at each point using the computed first derivatives. This scheme usually works fine, but for some cases the initial guess may be so poor that the optimization fails to start off properly or spends many early steps improving the Hessian without nearing the optimized structure. In addition, for optimizations to transition states, some knowledge of the curvature around the saddle point is essential, and the default approximate Hessian must always be improved.

There are a variety of options which retrieve or compute improved force constants for a geometry optimization. They are listed following this preliminary discussion.

There are two other approaches to providing the initial Hessian which are sometimes useful:

◊ **Input new guesses**: The default approximate matrix can be used, but with new guesses read in for some or all of the diagonal elements of the Hessian. This is specified in the ModRedundant input or on the variable definition lines in the Z-matrix. For example:

```
1 2 H 0.55
```

The letter H indicates that a diagonal force constant is being specified for this coordinate and that its value is 0.55 Hartree/au^2.

◊ **Compute some or all of the Hessian numerically**: You can ask the optimization program to compute part of the second derivative matrix numerically. In this case each specified variable will be stepped in only one direction, not both up and down as would be required for an accurate determination of force constants. The resulting second-derivatives are not as good as those determined by a frequency calculation but are fine for starting an optimization. Of course, this requires that the program do an extra gradient calculation for each specified variable. This procedure is requested by a flag (D) on the variable definition lines:

```
1 2 D
1 2 3 D
```

This input tells the program to do three points before taking the first optimization step: the usual first point, a geometry with the bond between atoms 1 and 2 incremented slightly, and a geometry with the angle between atoms 1, 2 and 3 incremented slightly. The program will estimate all force constants (on and off diagonal) for bond(1,2) and angle(1,2,3) from the three points. This option is only available with the Berny and EF algorithms.

The following options select methods for providing improved force constants:

ReadFC

Extract force constants from a checkpoint file. These will typically be the final approximate force constants from an optimization at a lower level, or (much better) the force constants computed correctly by a lower-level frequency calculation (the latter are greatly preferable to the former).

CalcFC

Specifies that the force constants be computed at the first point using the current method (available for the HF, CIS, MP2, CASSCF, DFT, and semi-empirical methods only).

RCFC

Specifies that the computed force constants in Cartesian coordinates (as opposed to internal) from a frequency calculation are to be read from the checkpoint file. Normally it is preferable to pick up the force constants already converted to internal coordinates as described above (CalcFC). However, a frequency calculation occasionally reveals that a molecule needs to distort to lower symmetry. In this case, the computed force constants in terms of the old internal coordinates cannot be used, and instead Opt=RCFC is used to read the Cartesian force constants and transform them. Note that Cartesian force constants are only available on the checkpoint file after a frequency calculation. You cannot use this option after an optimization dies because of a wrong number of negative eigenvalues in the approximate second derivative matrix. In the latter case, you may want to start from the most recent geometry and compute some derivatives numerically (see below). ReadCartesianFC is a synonym for RCFC.

CalcHFFC

Specifies that the analytic HF force constants are to be computed at the first point. CalcHFFC is used with MP2 optimizations, and it is equivalent to CalcFC for DFT methods, AM1, PM3, PM3MM, PM6 and PDDG.

CalcAll

Specifies that the force constants are to be computed at every point using the current method (available for the HF,CIS, MP2, CASSCF, DFT, and semi-empirical methods only). Note that vibrational frequency analysis is automatically done at the converged structure and the results of the calculation are archived as a frequency job.

RecalcFC=*N*

Do analytic second derivatives at step 1 and every N^{th} step thereafter during an optimization.

VCD

Calculate VCD intensities at each point of a Hartree-Fock or DFT Opt=CalcAll optimization.

NoRaman

Specifies that Raman intensities are not to be calculated at each point of a Hartree-Fock Opt=CalcAll job (since it includes a frequency analysis using the results of the final point of the optimization). The Raman intensities add 10-20% to the cost of each intermediate second derivative point. NoRaman is the default for methods other than Hartree-Fock.

StarOnly

Specifies that the specified force constants are to be estimated numerically but that no optimization is to be done. Note that this has nothing to do with computation of vibrational frequencies.

NewEstmFC

Estimate the force constants using a valence force field. This is the default.

EstmFC

Estimate the force constants using the old diagonal guesses. Only available for the Berny algorithm.

FCCards

Requests that read the energy (although value is not used), cartesian forces and force constants from the input stream, as written out by Punch=Derivatives. The format for this input is:

Energy	*Format (D24.16)*
Cartesian forces	*Lines of format (6F12.8)*
Force constants	*Lines of format (6F12.8)*

The force constants are in lower triangular form: *((F(J,I),J=1,I),I=1,3N$_{atoms}$)*, where 3N$_{atoms}$ is the number of Cartesian coordinates.

Convergence-Related Options

These options are available for the Berny algorithm only.

Tight

This option tightens the cutoffs on forces and step size that are used to determine convergence. An optimization with Opt=Tight will take several more steps than with the default cutoffs. For molecular systems with very small force constants (low frequency vibrational modes), this may be necessary to ensure adequate convergence and reliability of frequencies computed in a subsequent job step. This option can only be used with Berny optimizations. For DFT calculations, Int=UltraFine should be specified as well.

VeryTight

Extremely tight optimization convergence criteria. VTight is a synonym for VeryTight. For DFT calculations, Int=UltraFine should be specified as well.

EigenTest

EigenTest requests and NoEigenTest suppresses testing the curvature in Berny optimizations. The test is on by default only for transition states in internal (Z-matrix) or Cartesian coordinates, for which it is recommended. Occasionally, transition state optimizations converge even if the test is not passed, but NoEigenTest is only recommended for those with large computing budgets.

Expert

Relaxes various limits on maximum and minimum force constants and step sizes enforced by the Berny program. This option can lead to faster convergence but is quite dangerous. It is used by experts in cases where the forces and force constants are very different from typical molecules and Z-matrices, and sometimes in conjunction with Opt=CalcFC or Opt=CalcAll. NoExpert enforces the default limits and is the default.

Loose

Sets the optimization convergence criteria to a maximum step size of 0.01 au and an RMS force of 0.0017 au. These values are consistent with the Int(Grid=SG1) keyword, and may be appropriate for initial optimizations of large molecules using DFT methods which are intended to be followed by a full convergence optimization using the default (Fine) grid. It is *not* recommended for use by itself.

Algorithm-Related Options

GEDIIS

Use GEDIIS optimization algorithm. This is the default for minimizations when gradients are available.

RFO

Requests the Rational Function Optimization [589] step during Berny optimizations. It is the default for transition state optimizations (Opt=TS). This was also the default algorithm for minimizations using gradients in Gaussian 03.

EF

Requests an eigenvalue-following algorithm [589–591], which is useful only for methods without derivatives (for which it is the default). Available for both minima and transition states. and EigenvalueFollow are all synonyms for EF. When used with Opt=Z-Matrix, a maximum of 50 variables may be optimized.

ONIOM-Related Options

Micro

Use microiterations in ONIOM(MO:MM) optimizations. This is the default, with selection of L120 or L103 for the microiterations depending on whether electronic embedding is on or off. NoMicro forbids microiterations during ONIOM(MO:MM) optimizations. Mic120 says to use microiterations in L120 for ONIOM(MO:MM), even for mechanical embedding. This is the default for electronic embedding. Mic103 says to perform microiterations in L103 for ONIOM(MO:MM). It is the default for mechanical embedding, and it cannot be used with electronic embedding.

QuadMacro

Controls whether the coupled, quadratic macro step is used during ONIOM(MO:MM) geometry optimizations [579]. NoQuadMacro is the default.

Coordinate System Selection Options

Redundant

Build an automatic set of redundant internal coordinates such as bonds, angles, and dihedrals from the current Cartesian coordinates or Z-Matrix values, using the old algorithm available in Gaussian 16. Perform the optimization using the Berny algorithm in these redundant internal coordinates. This is the default for methods for which analytic gradients are available.

Z-matrix

Perform the optimization with the Berny algorithm using internal coordinates [592–594]. In this case, the keyword FOpt rather than Opt requests that the program verify that a full optimization is being done (i.e., that the variables including inactive variables are linearly independent and span the degrees of freedom allowed by the molecular symmetry). The POpt form requests a partial optimization in internal coordinates. It also suppresses the frequency analysis at the end of optimizations which include second derivatives at every point (via the CalcAll option). See *Constructing Z-Matrices* for details and examples of Z-matrix molecule specifications.

Cartesian

Requests that the optimization be performed in Cartesian coordinates, using the Berny algorithm. Note that the initial structure may be input using any coordinate system. No partial optimization or freezing of variables can be done with purely Cartesian optimizations; the mixed optimization format with all atoms specified via Cartesian lines in the Z-matrix can be used along with Opt=Z-matrix if these features are needed. When a Z-matrix without any variables is used for the molecule specification, and Opt=Z-matrix is specified, then the optimization will actually be performed in Cartesian coordinates. Note that a variety of other coordinate systems, such as distance matrix coordinates, can be constructed using the ModRedundant option.

Generalized Internal Coordinate (GIC) Options

GIC

Build an automatic set of redundant internal coordinates using the new GIC algorithm. Perform the optimization using the Berny algorithm in the GIC-type internal coordinates. Note that the coordinates generated with this option can be the same bonds, angles, and dihedrals generated by the default algorithm. However, these coordinates are internally stored and manipulated as the generalized ones (e.g., relevant analytical derivatives with respect to Cartesian coordinates displacements can be calculated automatically via an auto differentiation engine). The GICs are more flexible and, in principle, can be any combination of standard mathematical functions. Note that Geom=Checkpoint Opt=GIC option is equivalent to Geom=(Checkpoint,GIC).

AddGIC

Add, delete, or modify GIC-type internal coordinate definitions (including scan and constraint information) before performing the calculation using the new GIC algorithm. This option requires a separate input section following the geometry specification. When used in conjunction with QST2 or QST3, a GIC input section must follow each geometry specification. The syntax of the GIC input section is described in *GIC Info*. Note that Opt=(ModRedundant,GIC) is equivalent to Opt=AddGIC. Note that Geom=Checkpoint Opt=ReadAllGIC is equivalent to Geom=(Checkpoint, ReadAllGIC).

GICOld

Build an automatic set of redundant internal coordinates using the current default algorithm (as with the option Redundant) and then convert the coordinates into the GICs and treat them as such. Perform the optimization using the Berny algorithm in the GIC-type internal coordinates.

ReadAllGIC

Do not build any redundant internal coordinates by default. Instead, read the input stream for user-provided GIC definitions and create the coordinates. Perform the optimization using the Berny algorithm in the GIC-type internal coordinates. This option requires a separate GIC input section following the geometry specification. When used in conjunction with QST2 or QST3, a GIC input section must follow each geometry specification. The syntax of the GIC input section is described in the GIC Considerations tab.

Rarely Used Options

Path=*M*

In combination with either the QST2 or the QST3 option, requests the simultaneous optimization of a transition state and an *M*-point reaction path in redundant internal coordinates [595]. No coordinate may be frozen during this type of calculation.

If QST2 is specified, the title and molecule specification sections for both reactant and product structures are required as input as usual. The remaining *M*-2 points on the path are then generated by linear interpolation between the reactant and product input structures. The highest energy structure becomes the initial guess for the transition structure. Each point is optimized to lie in the reaction path and the highest point is optimized toward the transition structure.

If QST3 is specified, a third set of title and molecule specification sections must be included in the input as a guess for the transition state as usual. The remaining *M*-3 points on the path are generated by two successive linear interpolations, first between the reactant and transition structure and then between the transition structure and product. By default, the central point is optimized to the transition structure, regardless of the ordering of the energies. In this case, *M* must be an odd number so that the points on the path may be distributed evenly between the two sides of the transition structure.

In the output for a simultaneous optimization calculation, the predicted geometry for the optimized transition structure is followed by a list of all *M* converged reaction path structures.

The treatment of the input reactant and product structures is controlled by other options: OptReactant, OptProduct, BiMolecular.

Note that the SCF wavefunction for structures in the reactant valley may be quite different from that of structures in the product valley. Guess=Always can be used to prevent the wavefunction of a reactant-like structure from being used as a guess for the wavefunction of a product-like structure.

OptReactant

Specifies that the input structure for the reactant in a path optimization calculation (Opt=Path) should be optimized to a local minimum. This is the default. NoOptReactant retains the input structure as a point that is already on the reaction path (which generally means that it should have been previously optimized to a minimum). OptReactant may not be combined with BiMolecular.

BiMolecular

Specifies that the reactants or products are bimolecular and that the input structure will be used as an anchor point in an Opt=Path optimization. This anchor point will not appear as one of the *M* points on the path. Instead, it will be used to control how far the reactant side spreads out from the transition state. By default, this option is off.

OptProduct

Specifies that the input structure for the product in a path optimization calculation (Opt=Path) should be optimized to a local minimum. This is the default. NoOptProduct retains the input structure as a point that is already on the reaction path (which generally means that it should have been previously optimized to a minimum). OptProduct may not be combined with BiMolecular.

Linear

Linear requests and NoLinear suppresses the linear search in Berny optimizations. The default is to use the linear search whenever possible.

TrustUpdate

TrustUpdate requests and NoTrustUpdate suppresses dynamic update of the trust radius in Berny optimizations. The default is to update for minima.

Newton

Use the Newton-Raphson step rather than the RFO step during Berny optimizations.

NRScale

NRScale requests that if the step size in the Newton-Raphson step in Berny optimizations exceeds the maximum, then it is to be scaled back. NoNRScale causes a minimization on the surface of the sphere of maximum step size [596]. Scaling is the default for transition state optimizations and minimizing on the sphere is the default for minimizations.

Steep

Requests steepest descent instead of Newton-Raphson steps during Berny optimizations. This is only compatible with Berny local minimum optimizations. It may be useful when starting far from the minimum, but is unlikely to reach full convergence.

UpdateMethod=*keyword*

Specifies the Hessian update method. *Keyword* is one of: Powell, BFGS, PDBFGS, ND2Corr, OD2Corr, D2CorrBFGS, Bofill, D2CMix and None.

HFError

Assume that numerical errors in the energy and forces are those appropriate for HF and post-SCF calculations (1.0D-07 and 1.0D-07, respectively). This is the default for optimizations using those methods and also for semi-empirical methods.

FineGridError

Assume that numerical errors in the energy and forces are those appropriate for DFT calculations using the default grid (1.0D-07 and 1.0D-06, respectively). This is the default for optimizations using a DFT method and using the default grid (or specifying Int=FineGrid).

SG1Error

Assume that numerical errors in the energy and forces are those appropriate for DFT calculations using the SG-1 grid (1.0D-07 and 1.0D-05, respectively). This is the default for optimizations using a DFT method and Int(Grid=SG1Grid).

5.58.3 Availability

Analytic gradients are available for the HF, all DFT methods, CIS, MP2, MP3, MP4(SDQ), CID, CISD, CCD, CCSD, QCISD, CASSCF, and all semi-empirical methods.

The Tight, VeryTight, Expert, Eigentest and EstmFC options are available for the Berny algorithm only.

Optimizations of large molecules which have many very low frequency vibrational modes with DFT will often proceed more reliably when a larger DFT integration grid is requested (Int=UltraFine).

5.58.4 Related Keywords

IRC, IRCmax, Scan, Force, Frequency, Geom

5.58.5 Examples

Output from Optimization Jobs. The string `GradGradGrad···` delimits the output from the Berny optimization procedures. On the first, initialization pass, the program prints a table giving the initial values of the variables to be optimized. For optimizations in redundant internal coordinates, all coordinates in use are displayed in the table (not merely those present in the molecule specification section):

```
 GradGradGradGradGradGradGradGradGradGradGradGradGradGradGradGradGrad
 Berny optimization.      The opt. algorithm is identified by the header format & this line.
 Initialization pass.
                   ----------------------------
                   !    Initial Parameters    !
                   ! (Angstroms and Degrees)  !
--------------------                          ----------------------
! Name  Definition              Value          Derivative Info.    !
--------------------------------------------------------------------
! R1    R(2,1)                  1.             estimate D2E/DX2   !
! R2    R(3,1)                  1.             estimate D2E/DX2   !
! A1    A(2,1,3)              104.5            estimate D2E/DX2   !
--------------------------------------------------------------------
```

The manner in which the initial second derivative are provided is indicated under the heading `Derivative Info.` In this case the second derivatives will be estimated.

Each subsequent step of the optimization is delimited by lines like these:

```
GradGradGradGradGradGradGradGradGradGradGradGradGradGradGradGradGrad
Berny optimization.
Search for a local minimum.
Step number   4 out of a maximum of  20
```

Once the optimization completes, the final structure is displayed:

```
Optimization completed.
   -- Stationary point found.
```

```
                    ----------------------------
                    !   Optimized Parameters   !
                    ! (Angstroms and Degrees)  !
--------------------                            --------------------
! Name  Definition              Value          Derivative Info.    !
--------------------------------------------------------------------
! R1    R(2,1)                  0.9892         -DE/DX =    0.0002  !
! R2    R(3,1)                  0.9892         -DE/DX =    0.0002  !
! A1    A(2,1,3)              100.004          -DE/DX =    0.0001  !
--------------------------------------------------------------------
```

The redundant internal coordinate definitions are given in the second column of the table. The numbers in parentheses refer to the atoms within the molecule specification. For example, the variable `R1`, defined as `R(2,1)`, specifies the bond length between atoms 1 and 2. The energy for the optimized structure will be found in the output from the final optimization step, which *precedes* this table in the output file.

Compound Jobs. Optimizations are commonly followed by frequency calculations at the optimized structure. To facilitate this procedure, the Opt keyword may be combined with Freq in the route section of an input file, and this combination will automatically generate a two-step job.

It is also common to follow an optimization with a single point energy calculation at a higher level of theory. The following route section automatically performs an HF/6-31G(d,p) optimization followed by an MP4/6-31G(d,p) single point energy calculation:

```
# MP4/6-31G(d,p)//HF/6-31G(d,p) Test
```

Note that the Opt keyword is not required in this case. However, it may be included if setting any of its options is desired.

Modifying Redundant Internal Coordinates. The following input file illustrates the method for modifying redundant internal coordinates within an input file:

```
# HF/6-31G(d) Opt=ModRedun Test

Opt job

0,1
C1  0.000   0.000   0.000
C2  0.000   0.000   1.505
O3  1.047   0.000  -0.651
H4 -1.000  -0.006  -0.484
H5 -0.735   0.755   1.898
H6 -0.295  -1.024   1.866
O7  1.242   0.364   2.065
H8  1.938  -0.001   1.499

3  8             Adds hydrogen bond (but not angles or dihedrals).
2  1  3          Adds C-C-O angle.
```

This structure is acetaldehyde with an OH substituted for one of the hydrogens in the methyl group; the first input line for ModRedundant creates a hydrogen bond between that hydrogen atom and the oxygen atom in the carbonyl group. Note that this adds only the bond between these two atoms The associated angles and dihedral angles could be added as well using the B action code:

```
3  8  B
```

Displaying the Value of a Desired Coordinate. The second input line for ModRedundant specifies the C-C=O bond angle, ensuring that its value will be displayed in the summary structure table for each optimization step.

Using Wildcards in Redundant Internal Coordinates. A distance matrix coordinate system can be activated using the following input:

`* * B`	*Define all bonds between pairs of atoms*
`* * * K`	*Remove all other redundant internal coordinates*

The following input defines partial distance matrix coordinates to connect only the closest layers of atoms:

`* * B 1.1`	*Define all bonds between atoms within 1.1 Å*
`* * * K`	*Remove all other redundant internal coordinates*

The following input sets up an optimization in redundant internal coordinates in which atoms *N1* through *Nn* are frozen (such jobs may require the NoSymm keyword). Note that the lines containing the B action code will generate Cartesian coordinates for all of the coordinates involving the specified atom since only one atom number is specified:

`N1 B`	*Generate Cartesian coordinates involving atom N1*
`...`	
`Nn B`	*Generate Cartesian coordinates involving atom Nn*
`* F`	*Freeze all Cartesian coordinates*

The following input defines special “spherical” internal coordinates appropriate for molecules like C_{60} by removing all dihedral angles from the redundant internal coordinates:

`* * * * R`	*Remove all dihedral angles*

Additional examples are found in the section on relaxed PES scans below.

Performing Partial Optimizations. The following job illustrates the method for freezing variables during an optimization:

```
# B3LYP/6-31G(d) Opt=ReadOpt

Partial optimization of Fe2S2
cluster with phenylthiolates.

-2,1
Fe 15.2630 -1.0091  7.0068
S  14.8495  1.1490  7.0431
Fe 17.0430  1.0091  7.0068
S  17.4565 -1.1490  7.0431
```

```
S  14.3762 -2.1581  8.7983
C  12.5993 -2.1848  8.6878
...
C  14.8285 -3.8823  3.3884
H  14.3660 -3.3149  2.7071

noatoms atoms=1-4
```
ReadOpt input.

The central cluster (the first four atoms) will be optimized while the phenylthiolates are frozen.

Restarting an Optimization. A failed optimization may be restarted from its checkpoint file by simply repeating the route section of the original job, adding the Restart option to the Opt keyword. For example, this route section restarts a B3LYP/6-31G(d) Berny optimization to a second-order saddle point:

```
%Chk=saddle2
# Opt=(TS,Restart,MaxCyc=50) Test
```

The model chemistry and starting geometry are retrieved from the checkpoint file. Options specifying the optimization type and procedure are required in the route section for the restart job (e.g., TS in the preceding example). Some parameter-setting options can be omitted to use the same values are for the original job, or they can be modified for the restarted job, such as MaxCycle in the example. Note that you must include CalcFC to compute the Hessian at the first point of the restarted job. Second derivatives are computed only when this option is present in the route section of the restarted job, regardless of whether it was specified for the original job.

Reading a Structure from the Checkpoint File. Redundant internal coordinate structures may be retrieved from the checkpoint file with Geom=Checkpoint as usual. The read-in structure may be altered by specifying Geom=ModRedundant as well; modifications have a form identical to the input for Opt=ModRedundant: *[Type] N1 [N2 [N3 [N4]]] [Action [Params]] [[Min] Max]]*

Locating a Transition Structure with the STQN Method. The QST2 option initiates a search for a transition structure connecting specific reactants and products. The input for this option has this general structure (blank lines are omitted):

`# HF/6-31G(d) Opt=QST2`	`# HF/6-31G(d) (Opt=QST2,ModRedun)`
First title section	*First title section*
Molecule specification for the reactants	*Molecule specification for the reactants*
Second title section	*ModRedundant input for the reactants*
Molecule specification for the products	*Second title section*
	Molecule specification for the products
	ModRedundant input for the products (optional)

Note that each molecule specification is preceded by its own title section (and separating blank line). If the ModRedundant option is specified, then each molecule specification is followed by any desired modifications to the redundant internal coordinates.

Gaussian will automatically generate a starting structure for the transition structure midway between the reactant and product structures, and then perform an optimization to a first-order saddle point.

The QST3 option allows you to specify a better initial structure for the transition state. It requires the two title and molecule specification sections for the reactants and products as for QST2 and also additional, third title and molecule specification sections for the initial transition state geometry (along with the usual blank line separators), as well as three corresponding modifications to the redundant internal coordinates if the ModRedundant option is specified. The program will then locate the transition structure connecting the reactants and products closest to the specified initial geometry.

The optimized structure found by QST2 or QST3 appears in the output in a format similar to that for other types of geometry optimizations:

```
                     ----------------------------
                     !   Optimized Parameters   !
                     ! (Angstroms and Degrees)  !
---------------------                          ----------------------
! Name  Definition     Value     Reactant  Product  Derivative Info.  !
---------------------------------------------------------------------
! R1    R(2,1)         1.0836    1.083     1.084    -DE/DX =    0.    !
! R2    R(3,1)         1.4233    1.4047    1.4426   -DE/DX =   -0.    !
! R3    R(4,1)         1.4154    1.4347    1.3952   -DE/DX =   -0.    !
! R4    R(5,3)         1.3989    1.3989    1.3984   -DE/DX =    0.    !
! R5    R(6,3)         1.1009    1.0985    1.0995   -DE/DX =    0.    !
! ...                                                                 !
---------------------------------------------------------------------
```

In addition to listing the optimized values, the table includes those for the reactants and products.

Performing a Relaxed Potential Energy Surface Scan. The Opt=ModRedundant option may also be used to perform a relaxed potential energy surface (PES) scan. Like the facility provided by Scan, a relaxed PES scan steps over a rectangular grid on the PES involving selected internal coordinates. It differs from Scan in that a constrained geometry optimization is performed at each point.

Relaxed PES scans are available only for the Berny algorithm. If any scanning variable breaks symmetry during the calculation, then you must include NoSymm in the route section of the job, since it may fail with an error.

Redundant internal coordinates specified with the Opt=ModRedundant option may be scanned using the S code letter: *N1 N2 [N3 [N4]]* S steps *step-size*. For example, this input adds a bond between atoms 2 and 3, specifying three scan steps of 0.05 Å each:

```
2 3 S 3 0.05
```

Wildcards in the ModRedundant input may also be useful in setting up relaxed PES scans. For example, the following input is appropriate for a potential energy surface scan involving the N1-N2-N3-N4 dihedral angle:

`N1 N2 N3 N4 S 20 2.0` *Specify a relaxed PES scan of 20 steps in 2^o increments*

Cartesian coordinates can also include scan specifications:

X *atom S x-steps x-size y-steps y-size z-steps z-size*

For example, the following ModRedundant input performs a relaxed potential energy surface scan. Starting at the initial position of atom 1 and moving in five 0.2 Angstrom steps in the X direction and three 0.1 Angstrom steps in the Z direction:

```
X 1 S 5 0.2 0 0.0 3 0.1
```

The Berny geometry optimization algorithm in Gaussian is based on an earlier program written by H. B. Schlegel which implemented his published algorithm [592]. The program has been considerably enhanced since this earlier version using techniques either taken from other algorithms or never published, and consequently it is appropriate to summarize the current status of the Berny algorithm here.

At each step of a Berny optimization the following actions are taken:

◊ The Hessian is updated unless an analytic Hessian has been computed or it is the first step, in which case an estimate of the Hessian is made. Normally the update is done using an iterated BFGS for minima and an iterated Bofill for transition states in redundant internal coordinates, and using a modification of the original Schlegel update procedure for optimizations in internal coordinates. By default, this is derived from a valence force field [588], but upon request either a unit matrix or a diagonal Hessian can also be generated as estimates.

◊ The trust radius (maximum allowed Newton-Raphson step) is updated if a minimum is sought, using the method of Fletcher [597–599].

◊ Any components of the gradient vector corresponding to frozen variables are set to zero or projected out, thereby eliminating their direct contribution to the next optimization step.

If a minimum is sought, perform a linear search between the latest point and the best previous point (the previous point having lowest energy). If second derivatives are available at both points and a minimum is sought, a quintic polynomial fit is attempted first; if it does not have a minimum in the acceptable range (see below) or if second derivatives are not available, a constrained quartic fit is attempted. This fits a quartic polynomial to the energy and first derivative (along the connecting line) at the two points with the constraint that the second derivative of the polynomial just reach zero at its minimum, thereby ensuring that the polynomial itself has exactly one minimum. If this fit fails or if the resulting step is unacceptable, a simple cubic is fit is done.

Any quintic or quartic step is considered acceptable if the latest point is the best so far but if the newest point is not the best, the linear search must return a point in between the most recent and the best step to be acceptable. Cubic steps are never accepted unless they are in between the two points or no larger than the previous step. Finally, if all fits fail and the most recent step is the best so far, no linear step is taken. If all fits fail and the most recent step is not the best, the linear step is taken to the midpoint of the line connecting the most recent and the best previous points.

◊ If the latest point is the best so far or if a transition state is sought, a quadratic step is determined using the current (possibly approximate) second derivatives. If a linear search was done, the quadratic step is taken from the point extrapolated using the linear search and uses forces at that point estimated by interpolating between the forces at the two points used in the linear search. By default, this step uses the Rational Function Optimization (RFO) approach [589, 591, 600, 601]. The RFO step behaves better than the Newton-Raphson method used in earlier versions of Gaussian when the curvature at the current point is not that desired. The old Newton-Raphson step is available as an option.

◊ Any components of the step vector resulting from the quadratic step corresponding to frozen variables are set to zero or projected out.

◊ If the quadratic step exceeds the trust radius and a minimum is sought, the step is reduced in length to the trust radius by searching for a minimum of the quadratic function on the sphere having the trust radius, as discussed by Jørgensen [596]. If a transition state is sought or if NRScale was requested, the quadratic step is simply scaled down to the trust radius.

◊ Finally, convergence is tested against criteria for the maximum force component, root-mean square force, maximum step component, and root-mean-square step. The step is the change between the most recent point and the next to be computed (the sum of the linear and quadratic steps).

Examples of Using GICs

Basic GIC input. Here is an example of using the generalized internal coordinates defined by the user from scratch for the geometry optimization of the water molecule.

```
# HF opt=readallgic

Title

 0 1
O    0.0000    0.0000    0.0000
H    0.0000    0.0000    1.3112
H    1.0354    0.0000   -0.6225

R(1,2)
R(1,3)
HOH=A(2,1,3)
```

The atomic indexes 1, 2, and 3 refer to the oxygen atom, the first and the second hydrogen atom, respectively. The first and the second expression define the O-H bonds, and the third one defines the H-O-H valence angle (with the user-provided label "HOH"). An excerpt of the output with a table containing the initial values of the GICs is shown below.

```
                          ----------------------------
                          !    Initial Parameters    !
                          ! (Angstroms and Degrees)  !
 --------------------------                            --------------------------
 ! Name  Definition              Value          Derivative Info.                !
 --------------------------------------------------------------------------------
 ! R1    R(1,2)                  1.3112         estimate D2E/DX2                !
 ! R2    R(1,3)                  1.2081         estimate D2E/DX2                !
 ! HOH   A(2,1,3)              121.015          estimate D2E/DX2                !
 --------------------------------------------------------------------------------
```

Note that the labels "R1" and "R2" above were assigned by default. The coordinates R1=R(1,2) and R2=R(1,3) are parsed as pure distances and given here in Angstroms, and the HOH=A(2,1,3) is a pure valence angle in degrees.

```
# HF opt=readallgic

Title

 0 1
O    0.0000    0.0000    0.0000
H    0.0000    0.0000    1.3112
H    1.0354    0.0000   -0.6225
```

```
OHSym1=(R(1,2)+R(1,3))/sqrt(2)
OHSym2=(R(1,2)-R(1,3))/sqrt(2)
HOH=A(2,1,3)
```

The first and the second expression in the example above define the symmetrized O-H bonds, and the third one is the H-O-H valence angle.

```
                            ----------------------------
                            !    Initial Parameters    !
                            ! (Angstroms and Degrees)  !
 --------------------------                            --------------------------
 ! Name   Definition              Value          Derivative Info.                !
 --------------------------------------------------------------------------------
 ! OHSym1 GIC-1                     3.3664         estimate D2E/DX2                !
 ! OHSym2 GIC-2                     0.1377         estimate D2E/DX2                !
 ! HOH    A(2,1,3)                121.015          estimate D2E/DX2                !
 --------------------------------------------------------------------------------
 NOTE: GIC-type coordinates are in arbitrary units.
```

The coordinates OHSym1 and OHSym2 are parsed as generic GICs and therefore given here in arbitrary units. The units are actually Bohrs in this case because the $2^{-1/2}$ factor is taken as dimensionless and the values of R(1,2) and R(1,3) are taken in Bohrs.

```
# HF opt=readallgic

Title

 0 1
O
H  1  1.3
H  1  1.2  2  120.

R12=SQRT[{X(2)-X(1)}^2+{Y(2)-Y(1)}^2+{Z(2)-Z(1)}^2]
R13=SQRT[{X(3)-X(1)}^2+{Y(3)-Y(1)}^2+{Z(3)-Z(1)}^2]
A0(Inactive)=DotDiff(2,1,3,1)/{R12*R13}
A213=ArcCos(A0)
```

The GIC input section above defines two bond distances and one valence angle expressed via Cartesian coordinates. The coordinate A0 is defined as the dot-product (DotDiff) of the vectors $\vec{R}_{12}$ and $\vec{R}_{13}$ divided by the product of their lengths, and it is selected as “inactive” (i.e., excluded from the geometry optimization). An excerpt of the output with a table containing the initial values of the GICs is shown below.

```
                            ----------------------------
                            !    Initial Parameters    !
                            ! (Angstroms and Degrees)  !
 --------------------------                            --------------------------
 ! Name  Definition               Value          Derivative Info.                !
 --------------------------------------------------------------------------------
 ! R12   GIC-1                      2.4566         estimate D2E/DX2                !
 ! R13   GIC-2                      2.2677         estimate D2E/DX2                !
```

```
! A213  GIC-3                   2.0944        estimate D2E/DX2               !
 --------------------------------------------------------------------------------
 NOTE: GIC-type coordinates are in arbitrary units.
```

The values of R12, R13, and the dot-product are calculated using the Cartesian coordinates given in Bohrs. The GIC arbitrary units are Bohrs (for R12 and R13) and radians (for A213).

5.58.6 GIC Info

The options that do not mention GIC and can be used with the Opt keyword should work as described – except for NoFreeze, which should not be combined with any GIC-related option. In the latter case, use the UnFreezeAll flag in the GIC input section.

This section discusses specifying generalized internal coordinates (GICs) in Gaussian input files. GICs have many potential uses: defining additional coordinates whose values are reported during geometry optimizations, freezing various structural parameters during the optimization of a molecular system, specifying parameters over which to perform a scan, defining constraints for geometry optimizations based on structural parameters or complex relationships between them, requesting calculation of parts of the Hessian, and other purposes.

The GIC input section is separated from the earlier input by a blank line. It has one or more lines containing coordinate definitions, expressions or standalone options. Here is a simple GIC input section for water illustrating some of the possible features:

`R(1,2)`	*Define a bond length coordinate for atoms 1 and 2*
`Bond2=R[1,3]`	*Define another bond length coordinate named Bond2*
`HOH(freeze)=A(2,1,3)`	*Define an optimization constraint: a bond angle coordinate named HOH (∠2-1-3)*

For an optimization, these coordinates will result in the bond angle remaining fixed at its initial value and the two bond distances being optimized.

The basic form of a coordinate is the following:

label(options)=expression

All of the components are optional. In the preceding examples, all components were present only in the third line. The first line contained only a coordinate expression, while the second line also contained a label without options. Note that options may also be placed following the expression:

```
HOH=A(2,1,3) Freeze
```

Labels are user-assigned identifiers for the coordinate. They are not case sensitive. Labels many contain letters and number, but must begin with a letter. If no label is specified, a generic one will be assigned by the program (e.g., R1, R2, A1, etc.). A parenthesized, comma-separated list of options can be included following the label if desired. Note that square brackets or braces may be substituted for parentheses anywhere in a coordinate definition.

Structural Parameters

Coordinates are defined by expressions. The simplest expressions simply identify a specific structural parameter within the molecule, using the following constructs. Note that an asterisk may be used as a wildcard for any atom number (see the examples).

R(*i*,*j*)

Define a bond coordinate between atoms *i* and *j*. B, Bond and Stretch are synonyms for R.

A(*i*,*j*,*k*)

Define a non-linear angle coordinate involving atoms *i*, *j* and *k* where the angle vertex is at atom *j*. Angle and Bend are synonyms for A.

D(*i*,*j*,*k*,*l*)

Define a dihedral angle between the plane containing atoms *i*, *j* and *k* and the plane containing atoms *j*, *k* and *l*. Dihedral and Torsion are synonyms for D.

L(*i*,*j*,*k*,*l*,*M*)

Define the linear bend coordinate involving atoms *i*, *j* and *k* where the angle vertex is at atom *j*. Linear and LinearBend are synonyms for L.

A linear bend definition has two components, indicated by *M* values of -1 and -2 for the first and second components, respectively (no other values are permitted). A linear bend is specified by defining its two orthogonal directions. These can be indicated in two ways:

◊ For a nonlinear molecule with more than 3 atoms, a fourth atom which does not form a linear angle with *i*, *j* and *k* in any combination can be used. In this case, *l* can be set to its atom number. For example, the following may be used to specify a linear bend involving atoms 1, 2 and 3 using atom 6 to determine the two orthogonal directions:

```
L(1,2,3,6,-1)
L(1,2,3,6,-2)
```

If *l* is set to -4, then the fourth atom will be determined automatically based on the molecular geometry.

◊ The other method is to project the linear bend onto one of the coordinate system's axial planes: the values of -1, -2 and -3 for *l* specify the YZ, XZ and XY planes (respectively). The value 0 may also be used to request that the appropriate plane be determined automatically:

```
L(1,2,3,0,-1)
L(1,2,3,0,-2)
```

X(*i*)

Define the x Cartesian coordinate for atom *i*. Cartesian(*i*,-1) and Cartesian(*i*,X) are synonyms, and Cartesian may be abbreviated as Cart.

Y(*i*)

Define the y Cartesian coordinate for atom *i*. Cartesian(*i*,-2) and Cartesian(*i*,Y) are synonyms, and Cartesian may be abbreviated as Cart.

Z(*i*)

Define the z Cartesian coordinate for atom *i*. Cartesian(*i*,-3) and Cartesian(*i*,Z) are synonyms, and Cartesian may be abbreviated as Cart.

XCntr(*atom-list*)

YCntr(*atom-list*)

ZCntr(*atom-list*)

Define x, y or z Cartesian coordinate for the geometric center (centroid) of a molecular fragment that contains specified atoms. The atom list is a comma-separated list of atom numbers and/or ranges. For example, XCntr(1,12-15,27) defines the x coordinate of the fragment containing atoms 1, 12, 13, 14, 15 and 27. If the atom list is omitted, it defaults to the entire molecule.

DotDiff(*i,j,k,l*)

Define the dot product *(a·b)* of the two Cartesian coordinate difference vectors *a* and *b* for atoms *i*,*j*,*k* and *l* determined as $a = (X_i - X_j, Y_i - Y_j, Z_i - Z_j)$ and $b = (X_k - X_l, Y_k - Y_l, Z_k - Z_l)$.

Compound Expressions

Complex expressions may be constructed by combining multiple items using one or more mathematical operations. The argument(s) A and B can be the labels of a previously defined coordinate, a valid GIC expression or even constants (integer or floating-point). The operation names are not case sensitive. The following operations are available:

◊ Square root: SQRT(A).
◊ Power of *e*: EXP(A) for e^A.
◊ Trigonometric functions: SIN(A), COS(A), TAN(A).
◊ Inverse cosine: ARCCOS(A).
◊ Addition: A+B
◊ Subtraction: A-B
◊ Multiplication: A*B
◊ Division: A/B
◊ Exponentiation: A***n* for A^n (*n* is an integer). The form A^*n* is also accepted.

Here are some simple examples which define symmetrized OH bonds in water:

```
R12(inactive)=B(1,2)
R13(inactive)=B(1,3)
RSym  = (R12 + R13)/SQRT(2)
RASym = [Bond(1,2) - Bond(1,3)]/SQRT(2)
```

The first two coordinates are set as inactive since they are intermediates not intended to be used in the optimization. Line 3 illustrates an expression using previously defined labels, while line 4 shows the use of literal expressions with operators. Note that the argument to the square root function is the constant 2.

Options

A comma separated list of options can follow the coordinate label, enclosed in parentheses. Alternatively, options may follow the expression, separated from it and from one another by spaces. All options are case insensitive.

For the purposes of geometry optimizations, a coordinate can be designated as:

◊ *Active*: The coordinate is part of the list of internal coordinates used in the geometry optimzation. In contrast, *Inactive* coordinates are not included in the set used for the geometry optimization. By default, active coordinates are unfrozen: allowed to change value (see the next bullet).

◊ *Frozen*: A coordinate whose value is held constant during the course of a geometry optimization. The values of active, unfrozen coordinates change during a geometry optimization. The frozen or unfrozen status of inactive coordinates is irrelevant during an optimization.

In the descriptions that follow, coordinates that “already exist” refers to previously-defined coordinates with the same label or the same value expression. Such coordinates may have been defined earlier in the input stream or retrieved from the checkpoint file from a previous job.

Active

If the specified coordinate does not already exist, build a new coordinate defined by the given expression, and flag it as active and unfrozen. If the coordinate was previously defined, then flag it as active and unfrozen (whatever its previous status). It is the default. Activate, Add and Build are synonyms for Active. May be abbreviated to A when specified following the expression.

Frozen

Build a coordinate defined by the expression if it does not exist, and flag the coordinate as active for geometry optimizations and freeze it at the current value.

Freeze is a synonym for Frozen. May be abbreviated to F when specified following the expression.

Inactive

If the coordiante does not already exist, build a new coordinate defined by the expression and flag it inactive. If the coordinate with the given label or for the given expression has been already built and flagged as active (frozen or unfrozen), then remove it from the geometry optimization by flagging it as inactive. Remove is a synonym for Inactive. May be abbreviated to R when specified following the expression.

Kill

Remove the coordinate from the list of internal coordinates used in geometry optimization along with any dependent coordinates by flagging all of them as inactive. The dependent coordinates include any coordinate that depends on the same atoms as the given coordinate. For example, R(1,5) Kill will result in removing the coordinate R(1,5) – the internuclear distance between atoms 1 and 5 – as well as the valence angles, dihedral angles and any other coordinate that depends on the Cartesian coordinates of atoms 1 and 5 in combination with other atoms in the molecule. RemoveAll is a synonym for Kill. May be abbreviated to K when specified following the expression.

PrintOnly

Include the initial value of the coordinate in the starting geometry in the Gaussian output file, and then flag it as inactive.

Modify

A label must be included in the coordinate specification for this option. It replaces the old coordinate with the specified label with the new expression, and flags the newly modified coordinate as active and unfrozen.

Diff

Calculate numerical second derivatives for the row and column of the initial Hessian corresponding to this coordinate. May be abbreviated to D when specified following the expression.

FC=*x*

Change the diagonal element for the given coordinate in the initial Hessian to *x*, a floating-point number in atomic units. ForceConstant is a synonym for FC.

Value=*x*

Set the initial value for the given internal coordinate to *x*, a floating point value. The units for the value are those of the Gaussian program, as defined by the Units keyword (Angstroms or degrees by default). The current Cartesian coordinates will be adjusted to match this value as closely as possible. This option should be used cautiously and sparingly. It is far easier and more reliable to set the initial molecular structure as desired in a graphical environment like GaussView.

StepSize=*x*,NSteps=*n*

These options are used to specify a relaxed potential energy surface scan in which the coordinate is incremented by x a total of n times, and a constrained optimization is perfromed from each resulting starting geometry. x should be a positive floating-point number in atomic units, n should be an integer >1. When these options follow the expression, the comma separating them should be replaced by a space.

Min=*min*,Max=*max*

This option is used in combination with Active, Freeze or Inactive. It adds, freezes or makes inactive the coordinate when its value satisfies the condition $min \leq value \leq max$. *min* and *max* are floating-point numbers in the units defined by the Units (Angstroms or degrees by default). If Min or Max is omitted, the condition becomes $value \leq max$ or $min \geq min$ respectively. When these options follow the expression, the comma should be replaced by a space.

action* OnlyIf *condition

action* IfNot *condition

These options provide conditional coordinate operations. They can only be placed following the expression defining the current coordinate. *Action* is one of Active, Freeze or Inactive. The *condition* is a label or expression for another coordinate. The specified action will be performed for the current coordinate if the coordinate referred to in *condition* is active for OnlyIf or inactive for IfNot. Note that the conditional test applies only to the *action* specified preceding the option and not to other options that may be present in the coordinate specification.

Standalone Options

The following options are independent of coordinate definitions and apply globally. They should be specified alone on their input line.

FreezeAll

Freeze all internal coordinate previously added as active.

UnFreezeAll

Unfreeze all internal coordinates previously added as active frozen.

RemoveAll

Remove/inactivate all internal coordinate previously added as active (frozen or unfrozen).

Atom *i* *action*

Apply the specified *action* to the Cartesian coordinates of atom i. If i is an asterisk, then the action applies to all atoms. *Action* is one of Active, Freeze, UnFreeze, Remove (make inactive), RemoveAll and XYZOnly. These options are as defined above; XYZOnly says to remove any internal coordinates that depend on atom i but to add/retain the coordinates of that atom. The default *action* is Active.

Examples

The following example manipulates some automatically-generated coordinates, defines some new ones, and then uses wildcards to remove coordinates related to specific atoms:

```
R(5,9) freeze                                Freeze bond distance R(5,9).
R(8,9)                                       Add a new active coordinate R(8,9) with a default label.
```

`Ang189 = A(1,8,9)`	*Add a new active coordinate A(1,8,9) labeled Ang189.*
`R10(remove)`	*Remove a coordinate labeled R10.*
`Dih6123(remove) = D(6,1,2,3)`	*If D(6,1,2,3) exists, then remove the coordinate.*
`Dis79(freeze) = R(7,9)`	*Freeze the coordinate R(7,9): if it is new, then label it Dis79; if it already exists, retain the old label.*
`G1 = (R16+R19)*0.529177`	*Add a new coordinate labeled G1.*
`Ang189a(modify)=cos(g2)*57.29577951`	*Change the definition of coordinate Ang189a.*
`R(11,*) remove`	*Remove distances between atom 11 and any other atom.*
`D(*,1,17,*) remove`	*Remove any dihedral built around the 1-17 bond.*

Note that if a specified coordinate already exists, then an entry adding it will result in an error (e.g., lines 1-3 above).

The following example first defines the centroids of two fragments. Then, it defines the interfragment distance as an optimization coordinate:

Define the center of Fragment 1, but don't include it in the optimization.
```
XC1(Inactive)=XCntr(1-10)
YC1(Inactive)=YCntr(1-10)
ZC1(Inactive)=ZCntr(1-10)
```
Define the center of Fragment 2, but don't include it in the optimization.
```
XC2(Inactive)=XCntr(11-21)
YC2(Inactive)=YCntr(11-21)
ZC2(Inactive)=ZCntr(11-21)
```
Define the distance F1-F2 and include it in the optimization. Its value will be reported in Å:
```
F1F2=sqrt[(XC1-XC2)^2+(YC1-YC2)^2+(ZC1-ZC2)^2]*0.529177
```

The following example requests a relaxed PES scan over the same coordinate:
```
F1F2(NSteps=10,StepSize=0.2)
```
The following example removes an angle coordinate generated by default if $\geq 179.9^o$, substituting a linear bend:

`A(1,2,3) Remove Min=179.9`	*Remove angle coordinate if too large.*
`L(1,2,3,0,-1) Add IfNot A(1,2,3)`	*Add linear bend only if the angle coordinate not active.*
`L(1,2,3,0,-2) Add IfNot A(1,2,3)`	

The following example removes an angle coordinate if it is $\leq$ the specified value, setting the corresponding force constant is set to 0.2 au. The latter applies whenever it is needed: as the initial force constant and the force constant to use should be variable be reactivated. The second line specifies the force constant for a bond coordinate:
```
A(1,2,3) Remove Min=3.139847 ForceConstant=0.2
R(1,2) FC=0.5
```
The following example sets the force constants for various coordinates. It also inactivates bond angle coordinates $\geq 179.8^o$:

```
R(1,*) FC=0.8
```

```
D(*,4,5,*) FC=0.4
A(*,1,*) FC=0.5
A(*,*,*) R Min=179.8
```

Limitations of GICs in the Current Implementation

In the current implementation, GICs can be successfully used for many purposes including optimization constraints and PES scans. However, there are potential problems with active composite coordinates including multiple dihedral angles. In general, coordinates comprised of combinations of bond distances and bond angles should behave well. Simple dihedral angles are also welll supported. Complex expressions involving multiple dihedral angles are acceptable for frozen coordinates and for PES scans. However, they should be avoided as active optimization coordinates.

In a non-GIC optimization, or one using GICs with only regular dihedrals, then the program is careful about the periodicity of these coordinates. For example, in deciding whether a step in the geometry is too big and needs to be scaled back, it recognizes that a change in value from 1 degree to 359 degrees is really a change of -2 degrees rather than 358 degrees. Similarly, in numerically differentiating the forces in order to update the Hessian, displacements between geometries in internal coordinates are needed, and the periodicity is accounted for. A problem can arise when a GIC is a combination of parts for which such periodicity is important, typically, combinations of multiple dihedral angles. For example, consider these GICs:

```
D1 = D(1,2,3,4)
D2 = D(5,6,7,8)
V1 = D1 + 2*D2
```

D1 and D2 are dihedral angles, but they are intermediates and are not used as variables in the optimization. Their periodicity is not currently recognized in the composite coordinate V1. Suppose they have values of 1 and 2 degrees at one geometry and 1 and 359 degress at the next. The change in the optimization variable V1 should be 0 + 2*(-3) = -6 degrees, but it is actually 0 + 2*(357) = 714 degrees, which looks like an enormous change. This will result in the optimization algorithm performing very poorly. V1 isn't a simple periodic function; it is necessary to apply periodicity to its component parts as it is computed, which is not done in the current GIC implementation.

GIC Units in Gaussian Output

The values of the GICs defined as pure distances and angles (including valence angles, linear bends and dihedral angles/torsions) are computed from the Cartesian coordinates in atomic units (Bohrs) and stored internally in Bohrs and radians. However, for the user's convenience, they are expressed as usual in Angstroms and degrees in the Gaussian output. In the case of a generic GIC (i.e., when the GIC is not a pure Cartesian coordinate, bond distance or angle), the GIC value is computed as a function of Cartesian coordinates and bond distances in Bohrs and angles in radians, combined with optional constants in user-defined units. Such generic GIC values (labeled as GIC) are computed, stored and output in these same units: i.e., if the GIC is a combination of bonds or a combination of valence angles, then the arbitrary units become Bohrs for the bonds and radians for the angles.

Use of ModRedundant Format Input

Modifications to the GICs can be read in using the ModRedundant format from the current internal coordinate algorithm. However, the old format is only available with the GICs that include only pure bond distances, bond angles or torsion angles. In addition, the old format and the new GIC format described above cannot be mixed together in the same input section.

5.59 Output

The Output keyword is most often used to create input file for various external programs. It can also write Fortran unformatted files containing calculation results. Its options control the contents of the created file.

5.59.1 Options

WFN

Write a PROAIMS wavefunction (*.wfn*) file. The name for the created file is read from the input stream, on a separate line. PSI is a synonym for WFN.

WFX

Write a wavefunction file used by the newer versions of AIMPAC (*.wfx* files). The name for the created file is read from the input stream, on a separate line. WfnX is a synonym for WFX.

GIAOCx

Include GIAO Cx in *.wfn* or *.wfx* file.

CSGTCx

Include CSGT Cx in *.wfn* or *.wfx* file.

Pickett

Write g tensors and other tensors for hyperfine spectra [602–607] to the output file in the form of input for Pickett's program [608] (see `spec.jpl.nasa.gov`). The following tensors can be computed by Gaussian [410, 413–415, 609, 610]:

- ◊ Nuclear electric quadrupole constants: all jobs
- ◊ Rotational constants: Freq=(VibRot[,Anharmonic])
- ◊ Quartic centrifugal distortion terms: Freq=(VCD,Anharmonic)
- ◊ Electronic spin rotation terms: NMR
- ◊ Nuclear spin rotation terms: NMR
- ◊ Dipolar hyperfine terms: all jobs
- ◊ Fermi contact terms: all jobs

SpinRotation

Synonym for NMR Output=Pickett. Includes all hyperfine tensors which can be computed without doing a vibrational frequency calculation.

RotationalConstants

Synonym for Freq=VibRot Output=Pickett. Includes almost all hyperfine tensors which can be computed while performing only a harmonic vibrational frequency calculation.

For HF and DFT, you can combine the two preceding options. Output=(RotatationalConstants, SpinRotation) includes all the tensors computable with no more than second derivatives. It is equivalent to Freq=(VCD,VibRot) Output=Pickett.

QuarticCentrifugal

Synonym for Freq=(VibRot,Anharm) Output=Pickett. Includes quartic rotation-vibration coupling, but does not include spin-rotation tensors which must be computed separately.

ReadAtoms

Read a list of the atoms to include in the input for Pickett's program (note that this program only accepts tensors for eight nuclei). Atoms numbers are specified in free format, and this input section is blank-terminated. By default, eight interesting atoms are selected automatically by the program.

Gaussian Interfacing-Related Options

MatrixElement

Requests that a text data file for interfacing to other programs be generated. RawMatrixElement requests a binary matrix element file be generated. See *Interfacing to Gaussian 16* for details.

I4Labels

When combined with the MatrixElement option, use Integer*4 values rather than the current Gaussian integer size when writing the matrix element file.

MO2ElectronIntegrals

When combined with the MatrixElement option, include two-electron integrals over MOs when writing the matrix element file. In order to include the two-electron integrals in the AO basis within the matrix element file, combine this option with SCF=Conventional.

DerivativeDensities

When combined with the MatrixElement option, include derivative densities from CPHF when writing the matrix element file.

GIAOInts

When combined with the MatrixElement option, include GIAO L/R**3 and d2H/dBdm one-electron integral derivatives when writing the matrix element file.

Files

When combined with the MatrixElement option, include the contents of the specified internal Gaussian file within the generated matrix element file. For example, the following option:

```
Output=(Matrix,Files=(123,(456,offset=1,integer=27)))
```

will cause the contents of internal file 123 (assumed by default to be real values) to be included in the matrix element file (labeled as "File 123"). The second item in the file list specifies the 27 integers in internal file 456 starting after the first (8-byte) word (labeled as "File 456 integers section 001"), as well as any real values following the integers (labeled "File 456 reals section 002").

5.59.2 Related Keywords

Punch, SCRF=COSMORS, Pop=MK(Antechamber), NMR=CSGT (ACID).

5.60 PBC

This keyword allows you to specify options for Periodic Boundary Conditions jobs. Note PBC is turned on simply by including translation vectors in the input structure, and this keyword is used only to control how PBC calculations are performed. If you do not need any of these options, you do not have to include the keyword to perform a PBC calculation.

5.60.1 Options

GammaOnly

Do just the Γ point (k=0) rather than full k-integration.

NKPoint=*N*

Do approximately *N* k-points.

CellRange=*N*

Go out *N* Bohr in each direction in setting up image cells.

NCellMin=*N*

Include at least *N* cells.

NCellMax=*N*

Include at most *N* cells in any part of the calculation.

NCellDFT=*N*

Include at least *N* cells in DFT XC quadrature. NCellXC is a synonym for this option.

NCellK=*N*

Include at least *N* cells in exact exchange. By default, if exact exchange is included, then this is twice the number of cells used for overlap-related quantities and XC quadrature.

5.60.2 Availability

HF and DFT analytic energy and optimizations, and numerical frequencies (no IR intensities). Not valid with SCRF or Charge. For periodic systems of any reasonable size, acceptable performance may only be feasible by using a pure DFT functional in combination with density fitting.

5.60.3 Examples

Periodic systems are specified with a normal molecule specification for the unit cell. The only additional required input are one, two or three translation vectors appended to the molecule specification (with no intervening blank line), indicating the replication direction(s). For example, the following input specifies a one-dimensional PBC single point energy calculation for neoprene:

```
# PBEPBE/6-31g(d,p)/Auto SCF=Tight

neoprene, -CH2-CH=C(Cl)-CH2- optimized geometry

0 1
C,-1.9267226529,0.4060180273,0.0316702826
H,-2.3523143977,0.9206168644,0.9131400756
H,-1.8372739404,1.1548899113,-0.770750797
C,-0.5737182157,-0.1434584477,0.3762843235
H,-0.5015912465,-0.7653394047,1.2791284293
C,0.5790889876,0.0220081655,-0.3005160849
C,1.9237098673,-0.5258773194,0.0966261209
H,1.772234452,-1.2511397907,0.915962512
H,2.3627869487,-1.0792380182,-0.752511583
Cl,0.6209825739,0.9860944599,-1.7876398696
TV,4.8477468928,0.1714181332,0.5112729831
```

The final line specifies the translation vector. Note that it specifies TV as the atom symbol.

The following molecule specification could be used for a two-dimensional PBC calculation on a graphite sheet:

```
0 1
C                   0.000000    0.000000    0.000000
C                   0.000000    1.429118    0.000000
TV                  2.475315    0.000000    0.000000
TV                 -1.219952    2.133447    0.000000
```

Here is the molecule specification that could be used for a three-dimensional PBC calculation on gallium arsenide:

```
0 1
 Ga                   0.000000    0.000000    0.000000
 Ga                   0.000000    2.825000    2.825000
 Ga                   2.825000    0.000000    2.825000
 Ga                   2.825000    2.825000    0.000000
 As                   1.412500    1.412500    1.412500
 As                   1.412500    4.237500    4.237500
 As                   4.237500    1.412500    4.237500
 As                   4.237500    4.237500    1.412500
 TV                   5.650000    0.000000    0.000000
 TV                   0.000000    5.650000    0.000000
 TV                   0.000000    0.000000    5.650000
```

5.61 Polar

This method keyword requests that the dipole electric field polarizabilities (and hyperpolarizabilities, if possible) be computed. No geometry change or derivatives are implied, but this keyword may be combined in the same job with numerical differentiation of forces by specifying both Freq and Polar in the route section. Freq and Polar may *not* be combined for methods lacking analytic gradients (MP4(SDTQ), QCISD(T), CCSD(T), and so on). Note that Polar is done by default when second derivatives are computed analytically.

The polarizability and hyperpolarizability are presented in the output in the standard orientation in lower triangular and lower tetrahedral order, respectively: α_{xx}, α_{xy}, α_{yy}, α_{xz}, α_{yz}, α_{zz} and β_{xxx}, β_{xxy}, β_{xyy}, β_{yyy}, β_{xxz}, β_{xyz}, β_{yyz}, β_{xzz}, β_{yzz}, β_{zzz}.

Normally, polarizabilities and hyperpolarizabilities are computed using static frequencies. However, frequency-dependent polarizabilities and hyperpolarizabilities [611–615] may be computed by including CPHF=RdFreq in the route section and specifying the desired frequency in the input file.

Optical rotations [616–625] may also be predicted via the OptRot option [396, 626–633]. See [634–636] for example applications.

Predicting ROA and Raman Spectra

Raman and ROA intensities can be calculated separately from calculation of the force constants and normal modes, to facilitate using a larger basis for these properties as recommended in [401]. The options Polar=Raman and Polar=ROA request that the force constants be picked up from the checkpoint file (i.e., from a previous Freq calculation) and new polarizability derivatives (and the other two tensor derivatives for ROA) be computed and combined with the force constants in predicting intensities and spectra. Test job *931* provides an example of a two-step ROA calculation. For these jobs, be aware that the energy reported in the log file archive entry and in the final checkpoint file/formatted checkpoint file is the one computed in the frequency job – i.e., the value read from the checkpoint file at the start of the job – and not the energy computed with the model chemistry of the Raman/ROA calculation and reported in the final `SCF Done` output line.

5.61.1 Options

ROA

Compute dynamic analytic Raman optical activity intensities using GIAOs [401]. This procedure requires one or more incident light frequencies to be supplied in the input to be used in the electromagnetic perturbations (CPHF=RdFreq is the default with Freq=ROA). See the Examples for a sample input file. This option is valid for Hartree-Fock and DFT methods.

Raman

Calculate Raman spectrum from force constants read-in from the checkpoint file.

OptRot

Perform optical rotation calculation. Use CPHF=RdFreq to specify the desired frequencies. Available for HF and DFT only. This option cannot be combined with NMR. Include IOp(10/46=7) in the route section to include the dipole-quadrupole contribution to the dipole-magnetic dipole polarizability in order to compute the full optical rotation tensor [399, 627]; the latter will be labeled as `Optical Rotation G' tensor` in the output. Note that doing so does not change the optical rotation.

DCSHG

Do extra frequency-dependent CPHF for dc-SHG (direct current second harmonic generation) hyperpolarizabilities. This option implies CPHF=RdFreq as well.

Gamma

Equivalent to Polar=(DCSHG,Cubic) to do 2nd hyperpolarizabilities.

Analytic

Analytically compute the polarizability and the hyperpolarizability when analytic third derivatives are available. This option is the default for method with analytic second derivatives: RHF and UHF, CASSCF, CIS, MP2 and DFT methods. Note that the polarizability is always computed during analytic frequency calculations.

WorkerPerturbations

During numerical frequencies using Linda parallelism, run separate displacements on each worker instead of parallelizing each energy+derivative evaluation across the cluster. More efficient, but requires specifying an extra worker on the master node. This is the default if at least 3 Linda workers were specified. NoWorkerPerturbations suppresses this behavior.

FourPoint

Do four displacements instead of two for each degree of freedom during numerical frequencies, polarizabilities, or freq=anharm. This gives better accuracy and less sensitivity to step size at the cost of doing twice as many calculations.

DoubleNumer

Computes hyperpolarizabilities in addition to polarizabilities for methods with analytic gradients (first derivatives). Computes polarizabilities by double numerical differentiation of the energy for methods without analytic derivatives. EnOnly is a synonym for DoubleNumer.

Cubic

Numerically differentiate analytic polarizabilities to produce hyperpolarizabilities. Applicable only to methods having analytic frequencies but no analytic third derivatives.

Numerical

Computes the polarizability as a numerical derivative of the dipole moment (it is the analytic derivative of the energy, of course, not the expectation value in the case of MP2 or CI energies). The default for methods for which only analytic first derivative gradients are available.

Step=*N*

Specifies the step size in the electric field to be 0.0001*N* atomic units (applies to numerical differentiation).

Restart

Restarts a *numerical* calculation from the checkpoint file. A failed Polar calculation may be restarted from its checkpoint file by simply repeating the route section of the original job, adding the Restart option to the Polar keyword. No other input is required.

Susceptibility

Compute magnetic susceptibility as well as other properties (see NMR). Available for HF and DFT only.

TwoPoint

When computing numerical derivatives, make two displacements in each coordinate. This is the default. FourPoint will make four displacements but only works with Link 106 (Polar=Numer). Not valid with Polar=DoubleNumer.

Dipole

Compute the dipole polarizabilities (the default).

5.61.2 Availability

The following table summarizes the options to Polar that are required to compute polarizabilities and hyperpolarizabilities for the available methods.

Method Capabilities	**Polarizability**	**Hyperpolarizability**
Analytic 3rd derivatives (HF, most DFT)	Polar (*default*=Analytic)	Polar (*default*=Analytic)
Analytic frequencies (MP2, CIS, ⋯)	Polar (*default*=Analytic)	Polar=Cubic
Only analytic gradients (CCSD, BD, ⋯)	Polar (*default*=Numeric)	Polar=DoubleNumer
No analytic derivatives (CCSD(T), ⋯)	Polar (*default*=DoubleNumer)	N/A

Frequency-dependent polarizabilities and hyperpolarizabilities (i.e., Polar CPHF=RdFreq) are available only for HF and DFT methods.

5.61.3 Related Keywords

Freq, CPHF=RdFreq

5.61.4 Examples

Frequency-Dependent Properties. The following job will compute frequency-dependent polarizabilities and hyperpolarizabilities using ω=0.1 Hartrees:

```
# Polar CPHF=RdFreq HF/6-31G(d)

Frequency-dependent calculation: w=0.1

molecule specification

0.1
```

Performing a frequency-dependent Polar calculation produces the results for the specified frequency following those for the static case within the output. For example, here are the polarizability values for a frequency-dependent job (ω=0.1 Hartree):

```
 SCF Polarizability for W=    0.000000:
                1              2              3
      1  0.482729D+01
      2  0.000000D+00  0.112001D+02
      3  0.000000D+00  0.000000D+00  0.165696D+02
 Isotropic polarizability for W=    0.000000       10.87 Bohr**3.
 SCF Polarizability for W=    0.100000:
                1              2              3
      1  0.491893D+01
      2  0.000000D+00  0.115663D+02
      3  0.000000D+00  0.000000D+00  0.171826D+02

 Isotropic polarizability for W=    0.100000       11.22 Bohr**3.
```

A static polarizability calculation would include only the first section. Similar output follows for hyperpolarizabilities and additional properties.

Optical Rotations. Here is the key part of the output for optical rotation jobs (OptRot option). In this case, we have performed a frequency-dependent calculation by including CPHF=RdFreq in the route section and specified a frequency of 589.3 nm:

```
 Dipole-magnetic dipole polarizability for W=    0.077318:
                 1              2              3
      1  -0.428755D+01 -0.175571D+01  0.000000D+00
      2  -0.552645D+01  0.987070D+01  0.000000D+00
      3   0.000000D+00  0.000000D+00 -0.676292D+00
 w=    0.077318 a.u., Optical Rotation Beta=     -1.6356 au.
 Molar Mass =  172.2694 grams/mole, [Alpha]( 5893.0 A) = -366.99 deg.
```

The specific rotation value is highlighted in the example output.

Recommended Model Chemistries for ROA. The following two-step job illustrates the recommended models for predicting ROA spectra from [401]:

```
%chk=freq.chk
# APFD/6-311+g(2d,2p) Opt Freq ···

Freq + ROA: optimization+frequency

0 1
molecule specification

--Link1--
%Oldchk=freq.chk
%Chk=roa.chk
# APFD/spAug-cc-pVTZ Polar=ROA Guess=Read Geom=Check ···

Freq + ROA: ROA calculation with larger basis set

0 1

532nm
```

5.62 Population

This properties keyword controls printing of molecular orbitals and several types of population analysis and atomic charge assignments. The default is to print just the total atomic charges and orbital energies, except for Guess=Only jobs, for which the default is Pop=Full (see Options). Populations are done once for single-point calculations and at the first and last points of geometry optimizations. Note that the Population keyword requires an option.

The density that is used for the population analysis is controlled by the Density keyword. Note that only one density and method of charge fitting can be used in a job step. If several combinations are of interest, additional jobs steps can be added by specifying Guess=Only Density=Check, to avoid repeating any costly calculations.

Population analysis results are given in the standard orientation.

Output controlled by the Population keyword includes:

- ◊ Molecular orbitals and orbital energies. By default, all orbitals are included, but the output can be limited to a specific orbital range with the Orbitals option.
- ◊ Atomic charge distribution. The total charge per fragment is also reported if applicable.
- ◊ Multipole moments: dipole through hexadecapole.

APT charges are also computed by default during vibrational frequency calculations [637].

5.62.1 Options

Options Controlling Output File Contents

None

No orbitals are printed, and no population analysis is done. This is the default for all calculations using the ZIndo method.

Minimal

Total atomic charges and orbital energies are printed. This is the default for all job types and methods except Guess=Only and/or ZIndo.

Regular

The five highest occupied and five lowest virtual orbitals are printed, along with the density matrices and a full (orbital by orbital and atom by atom) Mulliken population analysis. Since the size of the output depends on the square of the size of the molecule, it can become quite substantial for larger molecules.

Full

Same as the Regular population analysis, except that *all* orbitals are printed. This is the default for Guess=Only jobs.

Always

Perform a population analysis at every optimization step rather than just the initial and final ones.

Orbitals[=*N*]

AllOrbitals

Perform a population analysis of the highest *N* occupied and lowest *N* virtual orbitals, including the atomic contributions to each MO (see the examples). *N* defaults to 10. AllOrbitals may also be specified instead of Orbitals=*N* to request analysis of all orbitals. For open shell calculations, both alpha and beta orbitals are included.

ThreshOrbitals=*N*

Set the minimum contribution percentage to include in individual orbital population analysis. The default is 10.

Population Analysis Options

Bonding

Do a bonding population analysis in addition to the standard analysis. This is a Mulliken population analysis in which only density terms involving pairs of basis functions on different centers are retained.

MBS

Perform minimum basis set Mulliken population analysis [110, 181]. NoMBS tells the program not to perform MBS Mulliken population analysis.

Hirshfeld

Perform Hirshfeld population analysis [638–640]. CM5 atomic charges [641] are computed along with Hirshfeld charges. CM5 is a synonym for Hirshfeld. The HirshfeldEE option requests interatomic electrostatic interactions as well.

Biorthogonalize

Biorthogonalize unrestricted molecular orbitals in order to maximally align electron pairs.

SaveBiorth

Save biorthogonalized orbitals in the checkpoint file over the canonical MOs.

DCT

Compute Ciofini's D_{CT} charge transfer diagnostic [642, 643] for whichever unrelaxed excited-state density/densities are available and, if a relaxed density was requested, for that as well.

NaturalTransitionOrbitals

Requests Natural Transition Orbital analysis [644] of a CI-Singles or TD-DFT excited state. Must be accompanied by Density=(Check,Transition=*N*) in order to specify which transition density is to be used to generate the orbitals. To print the orbitals from several states of interest, run successive Pop=NTO Density=(Check,Transition=*N*) Guess=Only jobs after the initial excited state calculation. NTO is a synonym for this option.

SaveNaturalTransitionOrbitals

Save the generated orbitals in the checkpoint file, replacing the canonical ones if the density was read-in from there. SaveNTO is a synonym for this option. If you want to visualize the orbitals, you need to write them back to the checkpoint file. It is a good idea to do so with a copy of the checkpoint file for each state. After the initial excited state calculation (using `%Chk=ex.chk`), use a technique like the following to generate visualization data for each state:

```
$ g16 <<END                                      Run a Guess=Only job just to
                                                 generate orbitals.
%OldChk=excited.chk                              Work on copy of original check-
                                                 point file.
%Chk=staten.chk                                  Checkpoint file for this state.
# Geom=AllCheck ChkBas Guess=Only
  Density=(Check,Transition=n) Pop=SaveNTO       Repeat for each state.
END                                              End of Gaussian input file.
```

Charge Save Options

NoOrthogonalize

Suppress orthogonalization of saved Pop=(SaveMixed,NoOrthogonalize) orbitals (primarily for visualizing raw NBO results).

Save*Type*

Computed atomic charges can be stored on the checkpoint file for use in a later MM calculation with Geom=Check. The options Pop=SaveMulliken, Pop=SaveESP, Pop=SaveNPA, Pop=SaveCM5, and so on can be used to save the corresponding charges. In the case of a multilayer ONIOM calculation, only explicitly computed charges are saved by default: i.e., charges for atoms in the QM layer(s). Any atomic charges present in the input file will not be used and they will be replaced by the newly fitted charges saved by these options.

The additional option Uncharged will keep the atomic charges present in the input file and only fit charges on uncharged atoms in the QM layers(s). The option combinations Pop=(Uncharged, SaveMulliken), Pop=(Uncharged, SaveCM5) and so on will save both the original atomic charges plus the newly fitted charges for the atoms that were originally uncharged.

Natural Orbital-Related Options

NaturalOrbitals

Do a natural orbital analysis of the total density. NO is a synonym for NaturalOrbitals.

NOAB

Do separate natural orbital analyses for the α and β densities. NaturalSpinOrbitals is a synonym for NOAB.

AlphaNatural

Do separate natural orbital analyses for the α and β densities, but store only the α densities for use in a *.wfn* file (see Output=WFN). NOA is a synonym for AlphaNatural.

BetaNatural

Do separate natural orbital analyses for the α and β densities, but store only the β densities for use in a *.wfn* file (see Output=WFN). NOB is a synonym for BetaNatural.

SpinNatural

Generate natural orbitals for the spin density (with α considered positive).

By default, natural orbitals are not included in the checkpoint file. Use a second job step of this form to place the natural orbitals into the checkpoint file:

```
--Link1--
%Chk=name
# Guess=(Save,Only,NaturalOrbitals) Geom=AllCheck ChkBasis
```

Run the *formchk* utility on the resulting checkpoint file to prepare the orbitals for visualization.

Options For Generating Electrostatic Potential-Derived Charges

MK

Produce charges fit to the electrostatic potential at points selected according to the Merz-Singh-Kollman scheme [645, 646]. ESP and MerzKollman are synonyms for MK. The data file for Antechamber (the AMBER program for generating RESP charges) can be generated using Pop=MK IOp(6/50=1) and specifying the file name on a separate line at the end of the Gaussian input file.

MKUFF

Uses the MK fitting but using UFF radii, which are defined for the full periodic table.

CHelp

Produce charges fit to the electrostatic potential at points selected according to the CHelp scheme [647].

CHelpG

Produce charges fit to the electrostatic potential at points selected according to the CHelpG scheme [648].

HLY

Specifies the Hu, Lu, and Yang charge fitting method [649].

HLYGAt

Specifies the Hu, Lu, and Yang charge fitting method, but using Gaussian's standard atomic densities instead of those of HLY. The authors of HLY only parametrized the atomic densities required for the model for the first 18 elements. This is an alternative version that uses the HLY fitting scheme but with Gaussian's standard atomic densities, which are available for the entire periodic table. For systems which can be done either way, the difference in atomic charges is usually between 1% and 5%.

Dipole

When fitting charges to the potential, constrain them to reproduce the dipole moment. ESPDipole is a synonym for Dipole.

AtomDipole

When fitting charges to the potential, also fit a point dipole at each atomic center.

ReadRadii

Read in alternative radii (in Angstroms) for each element for use in fitting potentials. These are read as pairs of atomic symbol and radius, terminated by a blank line.

ReadAtRadii

Read in alternative radii (in Angstroms) for each atom for use in fitting potentials. These are read as pairs of atom number and radius, terminated by a blank line.

NBO-Related Options (Version 3)

NBO

Requests a full Natural Bond Orbital analysis, using NBO version 3 [14–21].

NCS

Requests a partitioning of the NMR shielding tensors (computed using GIAOs) into magnetic contributions from bonds and lone pairs using the Natural Chemical Shielding analysis of Bohmann et al. [650], which is based upon the NBO analysis method. By default, an analysis of the isotropic shielding is performed. NoNCS skips this analysis.

NCSDiag

Requests an NCS analysis of the diagonal tensor elements.

NCSAll

Requests an NCS analysis of all tensor components.

NPA

Requests just the Natural Population Analysis phase of NBO.

NBORead

Requests a full NBO analysis, with input controlling the analysis read from the input stream. Use this option to specify keywords for NBO. Refer to the NBO documentation for details on this input.

NBODel

Requests NBO analysis of the effects of deletion of some interactions. Only possible with SCF methods. Implies that NBO input will be read; refer to the NBO documentation for details. Note that NBO input starts in column 2 so that the UNIX shell does not interpret the initial $.

SaveNBOs

Save natural bond orbitals in the checkpoint file (for later visualization).

SaveNLMOs

Save natural localized molecular orbitals in the checkpoint file (for later visualization).

SaveMixed

Save the NBOs for the occupied orbitals and the NLMOs for the unoccupied orbitals in the checkpoint file (for later visualization).

NBO-Related Options (Version 6)

An interface to NBO version 6 is provided. Pop=NPA6, Pop=NBO6, Pop=NBO6Read and Pop=NBO6Delete request use of the separate NBO6 program via the external interface. The script required to run NBO6 and the program itself must be obtained from Frank Weinhold (`nbo6.chem.wisc.edu`).

5.62.2 Related Keywords

Density, Output=WFN

5.62.3 Examples

Orbital-by-Orbital Population Analysis. The following route section requests population analysis of the lowest 3 virtual orbitals and the highest 3 occupied orbitals:

```
# UHF/6-311+G(d) Pop=Orbitals=3
```

Here is the resulting output from a calculation on FeO^+ quartet:

```
Atomic contributions to Alpha molecular orbitals:
Alpha occ 16 OE=-0.923 is Fe1-d=1.00
Alpha occ 17 OE=-0.699 is O2-p=0.88
Alpha occ 18 OE=-0.690 is O2-p=0.68 Fe1-s=0.21
Alpha vir 19 OE=-0.253 is Fe1-s=0.70 Fe1-p=0.27
Alpha vir 20 OE=-0.188 is Fe1-p=0.71 O2-p=0.29
Alpha vir 21 OE=-0.133 is Fe1-p=1.04

Atomic contributions to Beta  molecular orbitals:
Beta  occ 13 OE=-0.801 is O2-p=0.79
Beta  occ 14 OE=-0.783 is Fe1-d=1.00
Beta  occ 15 OE=-0.758 is O2-p=0.89
Beta  vir 16 OE=-0.241 is Fe1-s=0.81 Fe1-p=0.17
Beta  vir 17 OE=-0.139 is Fe1-p=0.91 Fe1-d=0.14
```

Note that both alpha and beta orbital information is included. For each orbital, the output reports the orbital energy (labeled OE and given in atomic units), followed by the fractional contribution of all basis functions of a given angular momentum for each relevant atom.

This is an example of a system where it is hard to tell the spin state of the system, because the canonical α and β orbitals are quite different. If you subsequently run a Guess=(Read,Only,BiOrthogonalize) Pop=Orbital calculation to analyze the results, then the program transforms the α and β orbitals to match up as much as possible (occupied and virtual separately). In this case, orbital energies for the transformed orbitals are nor given. Rather, the Pop=Orbital analysis reports the overlaps between the pairs of corresponding orbitals (i.e., α 19 with β 19), with a value of 1 indicating 100% correspondence. Instead of labeling the orbitals as occupied or virtual, they are labeled:

`Docc`	Doubly occupied: alpha and beta are occupied, with a match of 90% or better.
`Asing, Bsing`	Singly occupied: alpha without matching beta occupied, or lower occupieds which do not match up with any orbital of the opposite spin.
`Dvir`	Virtual orbital with nearly the same alpha and beta orbitals.
`AVir, BVir`	Virtual orbital which does not match up with any of the other opposite-spin virtuals.

The program lists the orbitals which match another orbital only once within the output. Also, for each unpaired excess alpha spin occupied orbital, there is always some beta virtual orbital which matches it, and the latter are also omitted.

Here is the output for FeO^+ quartet, which has 3 more alpha electrons than beta but which turns out to really have 4 unpaired alpha occupieds and one unpaired beta occupied (the lowest doubly occupied and highest unoccupied orbitals have been cut from the output):

```
Atomic contributions to molecular orbitals:
Docc. orb 13 abOv=0.999 is Fe1-d=1.00
Docc. orb 14 abOv=0.990 is O2-p=0.72 Fe1-s=0.14 Fe1-d=0.14
Asing orb 15 abOv=0.316 is Fe1-d=0.99
Asing orb 16 abOv=1.000 is Fe1-d=0.94
Asing orb 17 abOv=1.000 is Fe1-d=0.91
Asing orb 18 abOv=1.000 is Fe1-d=0.91
Dvirt orb 19 abOv=1.000 is O2-s=1.92 Fe1-s=-0.67 O2-p=-0.43 Fe1-p=0.14
Dvirt orb 20 abOv=1.000 is Fe1-s=0.52 Fe1-p=0.37
Bsing orb 15 abOv=0.316 is O2-p=0.92
```

Orbitals 1-14 are doubly occupied. Alpha orbital 15 and beta orbital 15 are different singly occupied orbitals; all 4 unpaired alpha spins are on the Fe, and the one unpaired beta spin is on the oxygen. There are no unmatched alpha and beta virtuals within the default range of orbitals analyzed.

Fragment Level Decomposition. If fragment information is present, the output also reports populations over fragments. For Pop=Orbital jobs, the decomposition of each orbital by fragment is reported.

Default output including fragment populations:

```
Mulliken charges with hydrogens summed into heavy atoms:
             1
    1  Pd  -0.265855
    2  P    0.346314
    3  P    0.346314
    4  Cl  -0.168156
    5  Cl  -0.168156
    6  C    0.060982
    7  C    0.060982
    8  C   -0.106213
    9  C   -0.106213
Sum of Mulliken charges with hydrogens summed into heavy atoms =   0.00000
         Condensed to fragments (all electrons):
    1      -0.265855
    2      -0.168156
    3      -0.168156
    4       0.060982
    5       0.060982
    6       0.480203
```

Pop=Orbital output:

```
Alpha occ 60 OE=-0.247 is Cl4-p=0.22 Cl5-p=0.22 P3-p=0.12 P2-p=0.12
                          Fr6=0.36 Fr2=0.22 Fr3=0.22
```

Sample NBO 3 Input. The following input file requests a bond order analysis using NBO 3:

```
# B3LYP/6-31G(d,p) Pop=NBORead

Example of NBO bond orders

 0  1
C      0.000000    0.665676    0.000000
H      0.919278    1.237739    0.000000
H     -0.919239    1.237787    0.000000
C      0.000000   -0.665676    0.000000
H     -0.919278   -1.237739    0.000000
H      0.919239   -1.237787    0.000000

$nbo bndidx $end
```

5.63 Pressure

Specifies the pressure to be used for thermochemistry analysis (in atmospheres). The value should be specified as an option:

`# ··· Pressure=1.5`

The default is 1 atmosphere.

5.63.1 Options

Default

Restores the default pressure if a different value was returned by Geom=AllCheck.

5.64 Prop

This properties keyword tells Gaussian to compute electrostatic properties [97, 609, 610, 651]. By default, the potential, electric field, and electric field gradient at each nucleus are computed. The density used for the electrostatic analysis is controlled by the Density keyword.

5.64.1 Options

Property Selection Options

EFG

Specifies that potential, field and field gradient are to be computed. This is the default.

Potential

Specifies that the potential but not the field or field gradient are to be computed. NoPotential suppresses computation of the electric potential and higher properties.

Field

Specifies that the potential and field, but not the field gradient, are to be computed.

EPR

Compute the anisotropic hyperfine coupling constants (i.e., spin-dipole EPR terms) [97, 609, 610].

Input Source-Related Options

If both Read and Opt are specified, the order of the input sections is fixed points (Read), then optimized points (Opt).

Read

Causes the program to read a list of additional centers at which properties will be computed from the input stream. The Cartesian coordinates of each center in angstroms are read in free field, with one center per line, in the standard orientation.

Opt

Causes the program to read a list of centers as in Prop=Read, but then to locate the minimum in the electric potential closest to each specified point.

FitCharge

Fit atomic charges to the electrostatic potential at the van der Waals surface.

Dipole

Constrain fitted charges to the dipole moment.

Grid

Specifies that the potential is to be calculated at one or more grids of points and written to an external file (generally superseded by the *cubegen* utility). This option requests mapping of the electric potential over a 2D grid of points. The points can be specified as a uniform rectangular grid, as an arbitrary collection read from an auxiliary file (both described below), or via the input format used by *cubegen*.
Three additional input lines are required for a uniform grid:

KTape,XO,YO,ZO	*Fortran unit for write, coords. of map's lower left corner.*
N1,X1,Y1,Z1	*# grid rows & vertical step size.*
N2,X2,Y2,Z2	*# grid column & horizontal step size.*

For points read from an auxiliary file, a single line of input supplies all of the necessary information:
N,NEFG,LTape,KTape
The coordinates of *N* points in Angstroms will be read from unit *LTape*, in format 3F20.12. *LTape* defaults to 52. The potential (*NEFG*=3), potential and field (*NEFG*=2), or potential, field, and field gradient (*NEFG*=1) will be computed and written to unit *KTape*. For example, the following input indicates that 19,696 points for the electrostatic potential (code 3) will be read from Fortran unit 10, with output written to Fortran unit 11:

```
19696,3,10,11
```

5.64.2 Availability

HF, all DFT methods, CIS, TD, MP2, MP3, MP4(SDQ), CID, CISD, CCD, CCSD, and QCISD.

5.64.3 Related Keywords

Density, *Cubegen*

5.65 Pseudo

This keyword requests that a model potential be substituted for the core electrons. The Cards option is by far its most-used mode. Gaussian supports a new effective core potential (ECP) input format (similar to that used by ExtraBasis) which is described in the Format tab. When reading-in pseudopotentials, do not give them the same names as any internally-stored pseudopotentials: CEP, CHF, LANL1, LANL2, LP-31, SDD, and SHC.

If used with ONIOM, the Pseudo and keyword applies to all layer of the ONIOM. If you want to read in ECPs only for one ONIOM layer, then use the GenECP keyword instead.

Without any options, this keyword defaults to Pseudo=Read.

5.65.1 Options

Read

Read pseudopotential data from the input stream. Input is described in the next subsection below. Cards is a synonym for Read.

SOScal

When reading pseudopotentials from input using Pseudo=Read, scale the spin-orbit coefficients (if present) by 2/L as appropriate for published CRENBL potentials. The default is not to scale, which is appropriate for Dolg (Stuttgart) potentials.

CHF

Requests the Coreless Hartree-Fock potentials. This option is normally used with the LP-31G basis sets.

SHC

Requests the SHC potentials.

LANL1

Requests the LANL1 potentials.

LANL2

Requests the LANL2 potentials.

Old

Read pseudopotential data using the old format (used by Gaussian 92 and earlier versions).

5.65.2 Format

Full ECP Input Format

Effective Core Potential operators are sums of products of polynomial radial functions, Gaussian radial functions and angular momentum projection operators. ECP input therefore specifies which potential to use on each atomic center, and then includes a collection of triplets of:

(coefficient, power of R, exponent)

for each potential for each term in each angular momentum of the ECP. Since only the first few angular momentum components have different terms, the potential is expressed as (1) terms for the general case, typically d or f and higher projection, and (2) the extra terms for each special angular momentum. Thus for an LP-31G potential, which includes special s and p projected terms, the input includes the general (d and higher) term, the s-d term (i.e., what to add to the general term to make the s component) and the p-d term.

All ECP input is free-format. Each block is introduced by a line containing the center numbers (from the molecule specification) and/or atomic symbols, specifying the atoms and/or atoms types to which it applies (just as for general basis set input-see the discussion of the Gen keyword). The list ends with a value of 0.

The pseudopotential for those centers/atoms follows:

Name,Max,ICore

Name of the potential, maximum angular momentum of the potential (i.e., 2 if there are special s and p projections, 3 if there are s, p, and d projections), and number of core electrons replaced by the potential. If *Name* matches the name of a previous potential, that potential is reused and no further input other than the terminator line is required.

For each component (*I*=1 to *Max*) of the current potential, a group of terms is read, containing the following information:

Title

A description of the block, not otherwise used.

NTerm

Number of terms in the block.

NPower,Expon,Coeff[,SO]

Power of R, exponent, and coefficient for each of the *NTerm* terms. *NPower* includes the R^2 Jacobian factor. The optional SO coefficient is for use with ECP basis sets which include this term.

Simplified ECP Input Format

Gaussian adds flexibility to ECP input by allowing it to include pre-defined basis sets names. An ECP definition may be replaced by a line containing the standard keyword for a pre-defined basis set. In this case, the ECPs within the specified basis set corresponding to the specified atom type(s) will be used for that atom (see the examples).

5.65.3 Keywords

In Pseudo input, keywords for these ECPs are of the form *XYn* where *n* is the number of core electrons which are replaced by the pseudopotential and *X* denotes the reference system used for generating the pseudopotential (S for a single-valence-electron ion or M for a neutral atom).

Y specifies the theoretical level of the reference data: HF for Hartree-Fock, WB for Wood-Boring quasi-relativistic and DF for Dirac-Fock relativistic. For one- or two-valence electron atoms SDF is a good choice; otherwise MWB or MDF is recommended (although for small atoms or for the consideration of relativistic effects, the corresponding SHF and MHF pseudopotentials may be useful).

5.65.4 SD ECP Availability

The Stuttgart/Dresden ECPs are not uniformly available across the periodic table. The following table shows the availability of the various XY combinations, along with valid values for *n*. The Defaults columns list the equivalencies for the SDD keyword (which selects an all electron basis set through Cl and ECPs thereafter) and when IOp(3/6) is set to 6 (which selects ECPs for all elements).

Note: These ECPs are not available for elements 87 (Fr), 88 (Ra), and 105 and higher.

		IOp(3/6=6)	SDD keyword	Valid values of *n* for given values of X and Y				
Z	Atom	Default	Default	MWB	SDF	SHF	MDF	MHF

1	H	D95	D95					
2	He	D95	D95					
3	Li	SDF2	D95					
4	Be	SDF2	D95		2			
5	B	MWB2	D95	2	2			
6	C	MWB2	D95	2	2			
7	N	MWB2	D95	2	2			
8	O	MWB2	D95	2	2			
9	F	MWB2	D95	2	2			
10	Ne	MWB2	D95	2				2
11	Na	SDF10	6-31G		10			
12	Mg	SDF10	6-31G		10			
13	Al	MWB10	D95	10	10			
14	Si	MWB10	D95	10	10			
15	P	MWB10	D95	10	10			
16	S	MWB10	D95	10	10			
17	Cl	MWB10	D95	10	10			
18	Ar	MWB10	6-31G	10				10
19	K	MWB10	MWB10	10	18	18		
20	Ca	MWB10	MWB10	10	18	18		
21	Sc	MDF10	MDF10				10	10
22	Ti	MDF10	MDF10				10	10
23	V	MDF10	MDF10				10	10
24	Cr	MDF10	MDF10				10	10
25	Mn	MDF10	MDF10				10	10
26	Fe	MDF10	MDF10				10	10
27	Co	MDF10	MDF10				10	10
28	Ni	MDF10	MDF10				10	10
29	Cu	MDF10	MDF10			28	10	10
30	Zn	MDF10	MDF10	28	28		10	10
31	Ga	MWB28	MWB28	28	28			
32	Ge	MWB28	MWB28	28	28	28		
33	As	MWB28	MWB28	28	28			
34	Se	MWB28	MWB28	28	28			
35	Br	MWB28	MWB28	28	28			
36	Kr	MWB28	MWB28	28				28
37	Rb	MWB28	MWB28	28	36	36		
38	Sr	MWB28	MWB28	28	36	36		
39	Y	MWB28	MWB28	28				28
40	Zr	MWB28	MWB28	28				28
41	Nb	MWB28	MWB28	28				28

42	Mo	MWB28	MWB28	28				28
43	Tc	MWB28	MWB28	28				28
44	Ru	MWB28	MWB28	28				28
45	Rh	MWB28	MWB28	28				28
46	Pd	MWB28	MWB28	28				28
47	Ag	MWB28	MWB28	28		46		28
48	Cd	MWB28	MWB28	28				28
49	In	MWB46	MWB46	46	46			
50	Sn	MWB46	MWB46	46	46			
51	Sb	MWB46	MWB46	46	46			
52	Te	MWB46	MWB46	46	46			
53	I	MWB46	MWB46	46	46		46	
54	Xe	MWB46	MWB46	46				46
55	Cs	MWB46	MWB46	46	54	54		
56	Ba	MWB46	MWB46	46	54			
57	La	MWB28	MWB28	28, 46, 47				46, 47
58	Ce	MWB28	MWB28	28, 47, 48				47, 48
59	Pr	MWB28	MWB28	28, 48, 49				48, 49
60	Nd	MWB28	MWB28	28, 49, 50				49, 50
61	Pm	MWB28	MWB28	28, 50, 51				50, 51
62	Sm	MWB28	MWB28	28, 51, 52				51, 52
63	Eu	MWB28	MWB28	28, 52, 53				52, 53
64	Gd	MWB28	MWB28	28, 53, 54				53, 54
65	Tb	MWB28	MWB28	28, 54, 55				54, 55
66	Dy	MWB28	MWB28	28, 55, 56				55, 56
67	Ho	MWB28	MWB28	28, 56, 57				56, 57
68	Er	MWB28	MWB28	28, 57, 58				57, 58
69	Tm	MWB28	MWB28	28, 58, 59				58, 59
70	Yb	MWB28	MWB28	28, 59				59
71	Lu	MWB60	MWB60	28, 60				60
72	Hf	MWB60	MWB60	60				60
73	Ta	MWB60	MWB60	60				60
74	W	MWB60	MWB60	60				60
75	Re	MWB60	MWB60	60				60
76	Os	MWB60	MWB60	60				60
77	Ir	MWB60	MWB60	60				60
78	Pt	MWB60	MWB60	60				60
79	Au	MWB60	MWB60	60		78	60	60
80	Hg	MWB60	MWB60	60, 78			60	60, 78
81	Tl	MWB78	MWB78	78				78
82	Pb	MWB78	MWB78	78				78

83	Bi	MWB78	MWB78	78			78
84	Po	MWB78	MWB78	78			78
85	At	MWB78	MWB78	78			78
86	Rn	MWB78	MWB78	78			78
89	Ac	MWB60	MWB60	60			60
90	Th	MWB60	MWB60	60			60
91	Pa	MWB60	MWB60	60			60
92	U	MWB60	MWB60	60			60
93	Np	MWB60	MWB60	60			60
94	Pu	MWB60	MWB60	60			60
95	Am	MWB60	MWB60	60			60
96	Cm	MWB60	MWB60	60			60
97	Bk	MWB60	MWB60	60			60
98	Cf	MWB60	MWB60	60			60
99	Es	MWB60	MWB60	60			60
100	Em	MWB60	MWB60	60			60
101	Md	MWB60	MWB60	60			60
102	No	MWB60	MWB60	60			60
103	Lr	MWB60	MWB60	60			60
104	Rf					92	

5.65.5 Related Keywords

ChkBasis, ExtraBasis, Gen, GenECP

5.65.6 Examples

Specifying an ECP. This input file runs an RHF/LP-31G calculation on hydrogen peroxide, with the basis set and ECP data read from the input file:

```
# HF/Gen Pseudo=Read Test

Hydrogen peroxide
0,1
O
H,1,R2
O,1,R3,2,A3
H,3,R2,1,A3,2,180.,0

R2=0.96
R3=1.48
A3=109.47
```

General basis set input

```
****
```

```
O 0                              ECPs for the oxygen atoms.
OLP 2 2                          ECP name=OLP, applies to d & higher, replaces 2 electrons.
D component                      Description for the general terms.
3                                Number of terms to follow.
1 80.0000000  -1.60000000
1 30.0000000  -0.40000000
2  1.0953760  -0.06623814
S-D projection                   Corrections for projected terms (lowest angular momentum).
3
0  0.9212952   0.39552179
0 28.6481971   2.51654843
2  9.3033500  17.04478500
P-D                              Corrections for projected terms (highest angular momentum).
2
2 52.3427019  27.97790770
2 30.7220233 -16.49630500
                                 Blank line indicates end of the ECP block.
```

The basis set data follows the molecule specification section. The first line of the ECP data requests that a potential be read in (type 7) for atoms number 1 and 3 (the oxygen atoms). No potential is to be used for atoms 2 and 4 (the hydrogen atoms).

The second line of ECP data begins the input for the centers requiring a read-in potential: in this case, oxygen atoms. The potential on these centers is named OLP, it is a general term and applies to angular momentum 2 (d) and higher, and the potential replaces two electrons. Next comes a title for the general term (D component), and the number of components of that term (3); the individual components follow on the next 3 lines. Next come the corrections for the projected terms in two sections, lowest angular momentum first. Each section again consists of a title line, the number of terms to follow, and then the terms themselves.

Using Standard Basis Set Keywords to Specify ECPs. The following input file illustrates the use of the simplified ECP input format:

```
# Becke3LYP/Gen Pseudo=Read Opt Test

HF/6-31G(d) Opt of Cr(CO)6

0 1
Cr 0.0  0.0  0.0
molecule specification continues ···

C O 0
6-31G(d)
****
Cr 0
LANL2DZ
****

Cr 0          ECP for chromium atom.
LANL2DZ       Use the ECP in this basis set.
```

5.66 Punch

This output specification keyword allows the user to “punch” – in more modern parlance, send to a separate output file – useful information at various points in the calculation. The output is disposed of in whatever manner is usual for Fortran alternate-unit output under the appropriate operating system (for example, unit 7 is sent to the file *fort.7* under UNIX). Options are used to specify what information should be output. All of these options can be combined, except that only one of MO and NaturalOrbitals can be requested. Note, however, that they are distinct and non-interacting. For example, Punch(MO, GAMESS) sends both the molecular orbital and GAMESS input information to the file; it does *not* format the MO information in GAMESS input format.

5.66.1 Options

Archive

Requests that a summary of the important results of the calculation be punched. This output is in the same format used by the Browse Quantum Chemistry Database System.

Title

Outputs the title section.

Coord

Outputs the atomic numbers and Cartesian coordinates in a form which could be read back into Gaussian.

Derivatives

Outputs the energy, Cartesian nuclear coordinate derivatives, and second derivatives in format 6F12.8, suitable for later use with Opt=FCCards.

MO

Outputs the orbitals in a format suitable for Guess=Cards input.

NaturalOrbitals

Outputs natural orbitals (for the density selected with the Density keyword).

HondoInput

Outputs an input deck for one version of Hondo, which is probably easily modified to fit most others.

GAMESSInput

Outputs an input deck for GAMESS.

All

Outputs everything except natural orbitals.

5.66.2 Related Keywords

Output, *chkchk -p*

5.67 QCI

This method keyword requests a Quadratic CI calculation [194], including single and double substitutions. Note that this keyword requests only QCISD and does not include the triples correction [652, 653] by default (see T in Options).

5.67.1 Options

T

Requests a Quadratic CI calculation including single and double substitutions with a triples contribution to the energy added [194].

E4T

Requests a Quadratic CI calculation including single and double substitutions with a triples contribution to the energy and also an evaluation of MP4 triples. Must be specified with the T option.

TQ

Requests a Quadratic CI calculation including single and double substitutions with an energy contribution from triples and quadruples [134] added.

SaveAmplitudes

Saves the converged amplitudes in the checkpoint file for use in a subsequent calculation (e.g., using a larger basis set). Using this option results in a very large checkpoint file, but also may significantly speed up later calculations.

ReadAmplitudes

Reads the converged amplitudes from the checkpoint file (if present). Note that the new calculation can use a different basis set, method (if applicable), etc. than the original one.

T1Diag

Computes the Q1 diagnostic of T. J. Lee and coworkers [195, 654]. Note that Q1 is analogous to the T1 diagnostic for CCSD when it is computed using QCISD instead of the Coupled Cluster method.

FC

All frozen core options are available with this keyword; a frozen core calculation is the default. See the discussion of the FC options for full information.

Conver=*N*

Sets the convergence calculations to 10^{-N} on the energy and $10^{-(N-2)}$ on the wavefunction. The default is N=7 for single points and N=8 for gradients.

MaxCyc=*n*

Specifies the maximum number of cycles. The default is 50.

5.67.2 Availability

Analytic energies and gradients for QCISD, numerical gradients for QCISD(T), and numerical frequencies for all methods.

5.67.3 Related Keywords

CCSD

5.67.4 Examples

The predicted energy from a QCISD calculation appears in the output in the final QCISD iteration:

```
DE(CORR)=  -.54999890D-01       E(CORR)=  -.7501966245D+02
```

When QCISD(T) is specified, the preceding output is followed by the energy including the non-iterative triples contribution:

```
QCISD(T)= -.75019725718D+02
```

5.68 Restart

This keyword restarts a previously-failed job. This method is primarily intended for long jobs that involve sufficiently large amounts of intermediate data, so that saving the restart data in the checkpoint file would make the checkpoint file unmanageably enormous, which would defeat the purpose of having a checkpoint file separate from the read-write file. This procedure depends on naming the read-write file so that it will be saved if the job terminates abnormally, and then using it to restart.

For example, in a frequency calculation you would include following Link 0 commands:

```
%RWF=myrwf
%NoSave
%Chk=mychk
#P Freq ···
```

input continues···

By default, any file which is named with a % line is retained when the job finishes. The %NoSave after the %RWF overrides this default, so that the read-write file will still be deleted if the job finishes normally (but will be left behind if the job terminates early).

Be careful about these points:

◊ The checkpoint file is often useful after the job finishes, so one typically places it after the %NoSave.

◊ The read-write file may be huge and should be put on a suitable file system. For example, the checkpoint file can be placed in a regular user directory, which might have only a moderate amount of free space and/or be NFS-mounted. However, the read-write file for a job large enough to be worth restarting should be on a large, local scratch file system.

◊ A restartable job is one for which execution was stopped before completion. Jobs that finish with an error message such as convergence failure or exceeding number of optimization steps require user intervention.

5.68.1 Availability

Analytic frequency calculations, including properties like ROA and VCD with ONIOM; CCSD and EOM-CCSD calculations; NMR; Polar=OptRot; CID, CISD, CCD, QCISD and BD energies. Calculations that compute multiple geometries, such as geometry optimizations, IRCs, and numerical frequency calculations, are better just restarted off the checkpoint file as before.

5.68.2 Related Keywords

Freq, EOM

5.68.3 Examples

The following input file will restart a job set up as described above:

```
%RWF=myrwf
%NoSave
```

```
%Chk=mychk
#P Restart
```

No other keywords are required in the route section, and no other input is needed.

5.69 SAC-CI

The keyword selects the Symmetry Adapted Cluster/Configuration Interaction (SAC-CI) methods of Nakatsuji and coworkers [655–677]. For detailed information on this method, consult the SAC-CI documentation available at the following web site: `www.qcri.or.jp/sacci/`. For review articles, see [661, 663, 678].

SAC-CI jobs must specify a reference state for the subsequent excited states calculations. For closed shell systems, the default RHF wavefunction used by SAC-CI is appropriate. For open shell ground states, you must either select an ROHF ground state wavefunction by including ROHF in the route section in addition to SAC-CI, or you must specify a closed shell state for the ground state calculation using the AddElectron or SubElectron option. See the examples for more information.

Spin State Options

Singlet=(*additional-options*)

Specifies that singlet states are to be calculated. The parenthesized list of suboptions specifies the desired states and other calculation parameters. Other spin state selection options are CationDoublet (Doublet is a synonym), AnionDoublet, Triplet, Quartet, Quintet, Sextet and Septet. More than one spin state may be specified.

5.69.1 Input

In the options that follow, *SpinState* is replaced by the name of the desired spin state.

***SpinState*=(NState=(i_1,i_2,$\cdots$))**

Sets the number of states of the specified type to be calculated for the various irreducible representations of the molecule's point group. Up to eight values may be specified, depending on the molecular symmetry (e.g., 8 for D_{2h}, 4 for C_{2v}, and so on). The shorthand form NState=*N* specifies a value of N for each irreducible representation. Degeneracies are handled by assuming the closest linear symmetry (e.g., D_2 for T_d).

***SpinState*=(Density)**

Calculate unrelaxed density matrices and perform Mulliken population analysis for all computed SAC-CI states of spin *SpinState*. See the examples for more information.

***SpinState*=(SpinDensity)**

Calculate spin density matrices for all computed SAC-CI states of spin *SpinState*. Implies the FullActive option as well.

***SpinState*=(NoTransitionDensity)**

By default, the transition density and oscillator strength are calculated between the SAC ground state and the SAC-CI singlet excited states when *SpinState* is Singlet, and between the lowest SAC-CI states and SAC-CI excited states for other spin states. NoTransitionDensity disables these calculations for the corresponding spin state.

TargetState=(SpinState=*s*, Symmetry=*m*, Root=*n*)

Specifies the target state for a geometry optimization or a gradient calculation, or for use with the Density keyword. *S* is the keyword indicating its spin multiplicity (i.e., Singlet, Doublet, etc.), *m* is the irreducible representation number of its point group, and *n* is the solution number in the desired spin state (determined by a previous energy calculation).

AddElectron

Add one electron to the open shell reference SCF configuration. This is the default for such systems for CationDoublet, Doublet, Quartet and Sextet.

SubElectron

Subtract one electron from the open shell reference SCF configuration. This is the default for such systems for AnionDoublet.

TransitionFrom=(SpinState=*s*, Symmetry=*m*, Root=*n*)

Specifies the initial state for calculating transition density matrices. *S* is the keyword indicating its spin multiplicity (i.e., Singlet, Doublet, etc.), *m* is the irreducible representation number of its point group, and *n* is the solution number in the desired spin state (as for TargetState above).

AllProperties

Calculate multipole moments through hexadecapole, all *n*th moments to the 4th moment, all electrostatic properties and the diamagnetic terms (shielding and susceptibility). This option applies to all spin states which specify the Density suboption.

NoProperty

Don't calculate any molecular properties.

SelectCISOnly

Terminate the calculation after the CIS initial guess has been calculated. You can use this option to determine the state number of a particular state in which you are interested (e.g., for TargetState). See the examples for an alternative method.

SACOnly

Performs only the calculation for the reference state and does not compute any excited states.

5.69.2 Options

Additional Options for Expert Users

For this set of options, SpinState below is replaced by the name of the desired spin state.

***SpinState*=(MaxR=*N*)**

Set the maximum excitation level to *N*.

***SpinState*=(NonVariational)**

Solve the SAC-CI equations for non-symmetric matrices. Variational proceeds by diagonalizing symmetrized matrices, and it is the default. Note that this option only applies to the excited state portion of the calculation (the ground state calculation always uses a nonvariational procedure).

***SpinState*=(InCoreDiag)**

Force use of the in-core algorithm.

***SpinState*=(Iterative=*item*)**

Force the use of an iterative algorithm. *item* specifies the initial guess type: SInitial for CIS and SDInitial for CISD.

Direct

Requests to use the direct algorithm for the SAC/SAC-CI SD-R calculations. Direct is not compatible with General-R, WithoutR2S2, FullUnlinked, InCoreDiag, and InCoreSAC. The direct SAC-CI code uses different values of internal thresholds, which correspond to the conventional SAC-CI calculations with NoUnlinkedSelection keywords. Therefore, the results obtained with the direct SAC-CI code are usually different from the results with the conventional SAC-CI code. The direct SAC-CI code is efficient and therefore strongly recommended. (available in Rev. B01 and later [1]) [673].

FC

All frozen core options are available with this keyword; a frozen core calculation is the default. See the discussion of the FC options for full information.

In general, the size of the active space greatly affects the accuracy of SAC-CI calculations. For this reason, using a full orbital window is recommended. Full is the default for geometry optimizations and gradient calculations.

LMO=*type*

Use the specified type of localized MO as reference orbitals. The available types are PM (Pipek-Mezey) and Boys.

Window=(*M*[,*N*])

Means that the SAC/SAC-CI calculation is done within the M-th to N-th active orbital space ($M < N$ in the energy order). Window=(M[,N]) is a synonym for ReadWindow=(M[,N]). If spin densities at the nuclei are of interest, as for Electron Spin Resonance or hyperfine splitting, then the core orbitals must be included in the window.

CorePrWindow=(*M*,*N*)

Activates the calculation of core-excited/core-ionized states and specifies core orbitals from which an electron is excited or ionized in the core-electron processes; M and N specify the range of core orbitals. This keyword is used with the FullActive or Window keyword to include core-orbitals in the active space (available in Rev. C01 and later [2]).

MacroIteration=*N*

Requests the use of N macroiterations within an optimization step. The default value of N is 0.

InCoreSAC

For solution of the SAC equations using the in-core algorithm.

MaxItDiag=*N*

Set the maximum number of diagonalization iterations. The default is 64 and the maximum is 999.

MaxItSAC=*N*

Set the maximum number of iterations for solving the SAC equations. The default is 999.

MaxItLin=*N*

Set the maximum number of iterations allowed to solve the SAC linear equations. The maximum is 999.

DConvDiag=*M*

Set the diagonalization energy convergence criteria to 10^{-M}.

DConvSAC=*M*

[1] This is relevant to G09, not G16. – Noted by the editor.

[2] This is relevant to G09, not G16. – Noted by the editor.

Set the energy convergence criteria to 10^{-M} when solving the SAC equations.

SD-R

Perform the calculation using singles and doubles linked excitation operators. This is the default.

General-R

Perform the calculation including linked excitation operators through sextuples.

LevelOne

Set the thresholds for selection of the double excitation operators to the lowest recommended level. LevelThree is the most accurate level, and it is the default. LevelTwo is intermediate in accuracy between the other two levels.

WithoutDegeneracy

By default, perturbation selection is performed so that degeneracies are retained. This option suppresses this test, resulting in reduced computational requirements. Use of this option is not recommended for production use.

NoLinkedSelection

Disables perturbation selection thresholds for linked operators (i.e., all operators are included).

NoUnlinkedSelection

Disables perturbation selection thresholds for unlinked operators (i.e., all operators are included).

FullUnlinked

Include all types of unlinked terms. Forces the use of the in-core algorithm.

In order to include all terms, all three of NoLinkedSelection, NoUnlinkedSelection, and FullUnlinked are required, currently at a considerable performance penalty.

WithoutR2S2

Ignore R2S2 unlinked integrals. This option results in a tradeoff between decreased accuracy and computational requirements.

EgOp

Generate quadruple and higher-order linked operators in the General-R scheme via the exponential generation algorithm. This is the default for single point energy calculations. The highest order excitation level is specified via the MaxR option (up to a maximum of 6). Perturbation selection thresholds are set via the LevelOne, LevelTwo and LevelThree options.

FullRGeneration

Generate all higher-order linked operators in the General-R scheme up to MaxR=4 and then perform perturbation selection as above. This is the default for gradient calculations and geometry optimizations.

These options are used to ensure consistency between all points in multipoint calculation types like potential energy surface scans. The Scan calculation must be performed three times: at the first point with BeforeGSUM, then at some or all subsequent points with CalcGSUM and then finally at all points with AfterGSUM. The actual results are provided by the final calculation. This procedure is only valid for singlet, triplet, ionized and electron-attached states, and it is not compatible with the General-R option.

BeforeGSUM

Initialize a series of linked calculations. Use this option in a calculation at the first point.

CalcGSUM

Collect data and determine the thresholds and operator selections at specified points in order to form a consistent set which can then be used at every point.

AfterGSUM

Perform SAC-CI calculations at each point using the GSUM data collected previously with the CalcG-SUM option.

These options can be used to increase the program default settings after a failed job has indicated that a resource shortfall was the problem.

MaxR2Op=*N*

Set the maximum number of R2 operators after perturbation selection to *N*. The default is 100,000.

MaxEgOp=*N*

Set the maximum number of operators in the General-R method to *N*. The default is 5,000.

5.69.3 Availability

Energy, analytical energy gradient, geometry optimization, and numerical frequencies.

Geometry optimizations default to using a full window. Specifying a different frozen core option for an optimization will result in numerical gradient calculations and correspondingly poorer performance.

5.69.4 Related Keywords

Density

5.69.5 Examples

If you want to locate the lowest two singlet excited states, you could use a route like the following:

```
# SAC-CI=(Singlet=(NState=8))/6-31G(d) NoSymm···
```

This will search for 8 singlet states, ignoring symmetry. The two lowest excited states will probably be among those found by the calculation.

Alternatively, you could use the following route:

```
# SAC-CI=(Singlet=(NState=4))/6-31G(d) ···
```

This calculation will locate the lowest four singlet excited states for each irreducible representation.

To specify the desired number of singlet excited states for each irreducible representation for a molecule with C_{2v} symmetry, use a route like this one:

```
# SAC-CI=(Singlet=(NState=(2,2,1,2)))/6-31G(d) ···
```

Direct SAC-CI. To use the direct SAC-CI code, use the following route:

```
# SAC-CI=(Direct,Singlet=(···),···)/6-31G(d) ···
```

Locating States with an Inexpensive Initial Calculation. You can use a preliminary, lower-accuracy calculation in order to locate a desired excited state at reduced computational cost. For example, the following route will locate 4 singlet excited states of each symmetry type:

```
# SAC-CI=(Singlet=(NState=4),LevelOne)/6-31G(d) ···
```

This job could be followed by a normal (LevelThree) calculation for the state(s) of interest. For example:

```
# SAC-CI=(Singlet=(1,0,1,0))/6-31G(d) ···
```

Calculations on Open Shell Systems. To predict excited states for vinyl radical, a neutral doublet radical, you could use a route like the following:

```
# ROHF/6-31G(d) SAC-CI=(Doublet=(NState=3),Quartet=(NState=3)) ···
```

This specifies the use of an ROHF wavefunction for the ground state, and it computes three doublet and three quartet excited states for each irreducible representation. You could use a similar approach for the triplet ground state of methylene.

Geometry Optimizations. To optimize a specific excited state, use the TargetState option:

```
# Opt SAC-CI=(Singlet=(Nstate=4),
  TargetState=(SpinState=Singlet,Symmetry=1,Root=2))/6-31G(d) ···
```

Computing Densities and Molecular Properties. To compute the unrelaxed density and population analysis for all predicted excited states, use a route like this one:

```
# SAC-CI=(Singlet=(···,Density),Triplet=(···,Density))/6-31G(d) ···
```

If you wanted to compute the unrelaxed density and population analysis only for the triplet states, then you would omit the Density suboption to the Singlet option.

To compute the relaxed density and population analysis for only one specified state, use a route like the following:

```
# SAC-CI=(Singlet=(NState=4),TargetState=(···)) Density=Current ···
```

Note that this job will be much more computationally expensive than the previous one as it requires a full gradient calculation.

SAC-CI Output. SAC-CI calculations produce a table like the following for each requested spin state (this example is for singlet states):

```
 ---------------------------------------------------------------------
 Transition dipole moment of   singlet state from SAC ground state
 ---------------------------------------------------------------------
 Symmetry  Sol Excitation  Transition dipole moment (au)      Osc.
               energy (eV)   X            Y            Z      strength
 ---------------------------------------------------------------------
    A1    0    0.0      Excitations are from this state.
    A1    1    8.7019     0.0000      0.0000      0.4645      0.0460
    A1    2   18.9280     0.0000      0.0000     -0.4502      0.0940
    A1    3   18.0422     0.0000      0.0000     -0.8904      0.3505
    A1    4   18.5153     0.0000      0.0000      0.0077      0.0000
    A2    1    7.1159     0.0000      0.0000      0.0000      0.0000
    A2    2   18.2740     0.0000      0.0000      0.0000      0.0000
    B1    1    1.0334    -0.2989      0.0000      0.0000      0.0023
    B1    2   18.7395    -0.6670      0.0000      0.0000      0.2042
    B1    3   22.1915    -0.1500      0.0000      0.0000      0.0122
    B1    4   15.8155     0.8252      0.0000      0.0000      0.2639
    B2    1   11.0581     0.0000      0.7853      0.0000      0.1671
    B2    2   15.6587     0.0000      1.5055      0.0000      0.8696
    B2    3   24.6714     0.0000     -0.7764      0.0000      0.3644
    B2    4   23.5135     0.0000     -0.1099      0.0000      0.0070
 ----------------------------------------------------------------------
```

Note that the various excited states are grouped by symmetry type – and not in order of increasing energy – in the output.

5.70 Scale

Specifies the frequency scale factor to be used for thermochemistry analysis. The value should be specified as an option:

```
# ··· Scale=0.95
```

The default is 1.0 except for compound methods where the scale factor is implicit in being the definition of the method used.

5.70.1 Options

Default

Restores the default scale factor for harmonic frequencies if a different value was retrieved by Geom=AllCheck.

5.71 Scan

This calculation type keyword requests that a potential energy surface (PES) scan be done. A *rigid* PES scan is performed, which consists of single point energy evaluations over a rectangular grid involving selected internal coordinates. The molecular structure must be defined using Z-matrix coordinates. The number of steps and step size for each variable are specified on the variable definition lines, following the variable's initial value. For example:

```
R1 1.41  3 0.05
A1 104.5 2 1.0
A2 120.0
```

This input causes variable R1 to be stepped 3 times by 0.05. Thus, four R1 values (1.41, 1.46, 1.51 and 1.56) will be done for each combination of other variables. Similarly, 3 values for A1 (104.5, 105.5 and 106.5) will be used, and A2 will be held fixed at 120.0. All in all, a total of 12 energy evaluations will be performed. Any number of variables can be stepped. The units of the step-sizes are controlled by the Units keyword and default to Angstroms and degrees.

A *relaxed* PES scan (with geometry optimization at each point) is requested with the Opt keyword.

If any scanning variable breaks symmetry during the calculation, then you must include NoSymm in the route section of the job, or the job will fail with an error.

5.71.1 Options

Restart

Restarts a PES scan calculation. A failed scan calculation may be restarted from its checkpoint file by simply repeating the route section of the original job, adding the Restart option to the Scan keyword. No other input is required.

5.71.2 Related Keywords

Opt

5.71.3 Examples

Output files from PES scans conclude with a table summarizing the results for the job:

```
Scan completed.

Summary of the potential surface scan:
  N         R           A           SCF
----  ---------  ---------  -----------
  1      0.9600    104.5000    -38.39041
  2      1.0100    104.5000    -38.41306
  3      1.0600    104.5000    -38.42336
  4      0.9600    105.5000    -38.39172
  5      1.0100    105.5000    -38.41430
  6      1.0600    105.5000    -38.42453
  7      0.9600    106.5000    -38.39296
  8      1.0100    106.5000    -38.41547
  9      1.0600    106.5000    -38.42564
 10      0.9600    107.5000    -38.39412
 11      1.0100    107.5000    -38.41657
 12      1.0600    107.5000    -38.42668
----  ---------  ---------  -----------
```

Chapter 6 of *Exploring Chemistry with Electronic Structure Methods* [152, pages 238–44, 255–58] provides a detailed discussion of potential energy surface scans.

5.72 SCF

This keyword controls the functioning of the SCF procedure. Options are used to specify the desired behavior, alternate algorithms, and so on.

The default SCF procedure uses a combination of EDIIS [679] and CDIIS, with no damping or Fermi broadening. In Gaussian 16, SCF=Tight is the default.

The SCF=QC option is often helpful with difficult conversion cases. For difficult-to-converge ROHF wavefunctions, where QC cannot be used, add Use=L506 to the route section.

See reference [680] for a discussion of SCF convergence and stability.

5.72.1 Options

Algorithm Selection Options

DIIS

DIIS calls for and NoDIIS prohibits use of Pulay's Direct Inversion in the Iterative Subspace (DIIS) extrapolation method [681].

CDIIS

Use only CDIIS. CDIIS implies Damp as well.

Fermi

Requests temperature broadening during early iterations [682], combined with CDIIS and damping. NoFermi suppresses Fermi broadening and is the default. By default, Fermi also implies Damp and also includes level shifting.

Damp

Turn on dynamic damping of early SCF iterations. NoDamp is the default. However, damping is enabled if SCF=Fermi or SCF=CDIIS is requested. Note that damping and EDIIS do not work well together.

NDamp=*N*

Allow dynamic damping for up to *N* SCF iterations (the default is 10).

QC

Calls for the use of a quadratically convergent SCF procedure [683]. By default this involves linear searches when far from convergence and Newton-Raphson steps when close (unless the energy goes up). This method is slower than regular SCF with DIIS extrapolation but is more reliable. SCF=QC is not available for restricted open shell (RO) calculations.

XQC

Add an extra SCF=QC step in case the first-order SCF has not converged. XQC defaults to MaxConventional=32.

YQC

Provides a new algorithm that is useful for difficult SCF convergence cases involving very large molecules. It does steepest descent and then scaled steepest descent as in QC, but then switches to regular SCF instead of quadratic convergence, using the quadratic algorithm only if the regular SCF fails to converge. YQC defaults to MaxConventional=32.

MaxConventionalCycles=*N*

Sets the limit on conventional SCF cycles during SCF=XQC and SCF=YQC to *N*.

PseudoDiagonalization=*N*

Use pseudo-diagonalization in Link 502 whenever possible, with full diagonalization only at the early cycles, at the end, and every N^{th} cycle in between. PDiag is a synonym for this option. This is the default for semi-empirical methods (the default is *N*=30).

FullDiagonalization

Forces full diagonalization in Link 502. This is the default for HF and DFT. FDiag is a synonym for this option.

SD

Does steepest descent SCF.

SSD

Does scaled steepest descent SCF.

SaveKPoint

Save k-point information at the conclusion of the SCF. NoSaveKPoint says not to save this data, and it is the default except for numerical frequency calculations for which SaveKPoint is the default.

DM

Calls for use of the direct minimization SCF program [684]. It is usually inferior to SCF=QC and retained for backwards compatibility and as a last resort. Available only for RHF closed shell and UHF open shell calculations.

VShift[=*N*]

Shift orbital energies by *N**0.001 (i.e., *N* milliHartrees); *N* defaults to 100. This option disables automatic archiving. *N*=-1 disables level shifting; NoVShift is equivalent to this setting.

MaxCycle=*N*

Changes the maximum number of SCF cycles permitted to *N*; the default is 64 (or 512 for SCF=DM and SCF=QC).

FullLinear

Specifies that L508 (SCF=QC, SD, or SSD) should do full linear searches at each iteration. By default, a full minimization is done only if the initial microiteration caused the energy to go up.

FinalIteration

FinalIteration performs and NoFinalIteration prevents a final non-extrapolated, non-incremental iteration after an SCF using DIIS or a direct SCF has converged. The default is NoFinalIteration.

IncFock

Forces use of incremental Fock matrix formation. This is the default for direct SCF. NoIncFock prevents the use of incremental Fock matrix formation, and it is the default for conventional SCF.

Pass

For in-core calculations, saves the integrals on disk as well, to avoid recomputing them in Link 1002. Only useful for frequency jobs in conjunction with SCF=InCore. NoPass forces integrals to be recomputed during each in-core phase.

TightLinEq

Use tight convergence in linear equation solution throughout SCF=QC. By default, the convergence criterion is tightened up as the rotation gradient is reduced.

VeryTightLinEq

Use even tighter convergence in the linear equation solutions (microiterations) throughout the QCSCF. This option is sometimes needed for nearly linearly-dependent cases. VTL is a synonym for VeryTightLinEq.

Integral Storage Options

Direct

Requests a direct SCF calculation, in which the two-electron integrals are recomputed as needed. This is the default SCF procedure in Gaussian. This is possible for all available methods, except for MCSCF second derivatives and anything using complex orbitals.

InCore

Insists that the SCF be performed storing the full integral list in memory. This is done automatically in a direct SCF calculation if sufficient memory is available. SCF=InCore is available to force in-core storage or abort the job if not enough is available. NoInCore prohibits the use of the in-core procedure, for both the SCF and CPHF.

Conventional

The two-electron integrals are stored on disk and read-in each SCF iteration. NoDirect is a synonym for Conventional.

Options Related to Convergence and Cutoffs

PtDensity=*N*

Specifies *N* points per Angstrom^2 (N>0) or -*N* tesserae (N<0). The default is 5.

Conver=*N*

Sets the SCF convergence criterion to 10^{-N}. SCF convergence requires both $<10^{-N}$ RMS change in the density matrix and $<10^{-(N-2)}$ maximum change in the density matrix. Note that the energy change is not used to test convergence; however, an SCF 10^{-N} RMS density matrix change typically corresponds to a 10^{-2N} change in energy in atomic units. For GVB and CASSCF calculations, SCF convergence is determined not by change in the density matrix, but rather in terms of the orbital change and energy change, respectively.

VarAcc

Use modest integral accuracy early in direct SCF, switching to full accuracy later on. This is the default for direct SCF, and it can be turned off via NoVarAcc. VarInt is a synonym for VarAcc, and NoVarInt is a synonym for NoVarAcc.

Tight

Use normal, tight convergence in the SCF. This is the default. Synonymous with TightIntegrals.

Big

Turns off optional $O(N^3)$ steps to speed up very large calculations (>5000 basis functions).

MaxNR=*N*

Sets the maximum rotation gradient for a Newton-Raphson step in SCF=QC and SCF=YQC to 10^{-N}. Below this threshold the program switches to the QC SCF procedure (if using SCF=QC), or to the regular SCF procedure (if using SCF=YQC). Above this, scaled steepest descent is used; above 100 times this, steepest descent is used. The default value for N is 2.

Symmetry-Related Options

IDSymm

Symmetrize the density matrix at the first iteration to match the symmetry of the molecule ("initial density symmetrize"). NoIDSymm is the default.

DSymm

Symmetrize the density matrix at every SCF iteration to match the symmetry of the molecule ("density symmetrize"). NoDSymm is the default. DSymm implies IDSymm.

NoSymm

Requests that all orbital symmetry constraints be lifted. It is synonymous with Guess=NoSymm and Symm=NoSCF.

Symm

Retain all symmetry constraints: make the number of occupied orbitals of each symmetry type (abelian irreducible representation) match that of the initial guess. Use this option to retain a specific state of the wavefunction throughout the calculation. It is the default only for GVB calculations.

IntRep

Calls for the SCF procedure to account for integral symmetry by replicating the integrals using the symmetry operations. Allows use of a short integral list even if the wavefunction does not have the full molecular symmetry. Available for L502 (the default for RHF, ROHF and UHF) and L508 (SCF=QC).

FockSymm

Calls for the SCF procedure to account for integral symmetry (use of the petite integral list) by symmetrizing the Fock matrices. This is the default. FSymm is a synonym for FockSymm.

Restart-Related Options

Save

Save the wavefunction on the checkpoint file every iteration, so the SCF can be restarted. This is the default for direct SCF. NoSave suppresses saving the wavefunction.

Restart

Restart the SCF from the checkpoint file. SCF=DM cannot be restarted. SCF=Restart skips steps which are not necessary when restarting an SCF calculation, but which are necessary when reading in a guess from a calculation with a different basis set or at a different geometry. In contrast, if you want to start a new SCF using the restart information from a calculation with a different geometry and/or a different basis set, use Guess=Restart.

5.73 SCRF

This keyword requests that a calculation be performed in the presence of a solvent by placing the solute in a cavity within the solvent reaction field.

The Polarizable Continuum Model (PCM) using the integral equation formalism variant (IEFPCM) is the default SCRF method. This method creates the solute cavity via a set of overlapping spheres. It was initially devised by Tomasi and coworkers and Pascual-Ahuir and coworkers [685–687], and it has been further developed in Gaussian by the Tomasi, Barone and Mennucci groups as well as Gaussian, Inc. researchers and collaborators [349, 352, 353, 360, 366, 367, 410, 688–702]. This model corresponds to SCRF=PCM. See [703] for a review. The model of Chipman [704] is closely related to this method [705].

Gaussian also offers the SMD variation of IEFPCM of Truhlar and workers [706] via the SMD option. This is the recommended choice for computing ΔG of solvation.

Other available models are IPCM, which uses a static isodensity surface for the cavity [707], the Self-Consistent Isodensity PCM (SCIPCM) model [707], and the Onsager model [708–713], which places the solute in a spherical cavity within the solvent reaction field.

In Gaussian 16, we use a continuous surface charge formalism that ensures continuity, smoothness and robustness of the reaction field, which also has continuous derivatives with respect to atomic positions and external perturbing fields [701]. This is achieved by expanding the apparent surface charge that builds up at the solute-solvent interface in terms of spherical Gaussian functions located at each surface element in which the cavity surface is discretized. Discontinuities in the surface derivatives are removed by effectively smoothing the regions where the spheres intersect. This formalism, initially proposed in 1999 by Karplus and York for the conductor screening model [714], never received the attention it deserved. We developed and generalized it within the framework of the PCM family of solvation methods in G09, and it is the default method for building the solute's cavity and computing the reaction field.

The PCM method in Gaussian 16 includes an external iteration procedure whereby the program computes the energy in solution by making the solvent reaction field self-consistent with the solute electrostatic potential (the latter being generated from the computed electron density with the specified model chemistry) [715, 716]. The difference with the standard approach (based on the variational approach or linear response theory) can be illustrated with MP2. The default procedure computes the solvent effect on the SCF density and then applies MP2 perturbation, while the external iteration approach computes the solvent effect self-consistently with respect to the MP2 density. While this technique is of primary interest for studying excited state processes such fluorescence, it can also be used for ground state calculations with theoretical methods that provide gradients: e.g. post-SCF methods. Use the ExternalIteration option to specify this method.

Solvation and Excited States

There are two basic approaches available for modeling excited states in solution:

◊ Computing the lowest excited states in the solvent environment. This approach adds SCRF to a normal excited state calculation such as TD or CIS. This technique employs a linear response formalism by adding the necessary terms to the excited state method equations (thereby including the solvent effects on the excited states) [349, 698]. The geometry of a specific excited state can be optimized in solution with CIS or TD [717].

◊ A single excited state can be modeled via a state-specific approach. In this case, the program computes the energy in solution by making the electrostatic potential generated by the excited state density self consistent with the solvent reaction field [715, 716], using the external iteration technique.

For excited state calculations in solution, there is a distinction between equilibrium and non-equilibrium calculations. The solvent responds in two different ways to changes in the state of the solute: it polarizes its electron distribution, which is a very rapid process, and the solvent molecules reorient themselves (e.g., by a rotation), a much slower process. An equilibrium calculation describes a situation where the solvent had time to fully respond to the solute (in both ways), e.g., a geometry optimization (a process that takes place on the same time scale as molecular motion in the solvent). A non-equilibrium calculation is appropriate for processes which are too rapid for the solvent to have time to fully respond, e.g. a vertical electronic excitation.

Equilibrium solvation is the default for CIS and TD-DFT excited state geometry optimizations. Non-equilibrium is the default for CIS and TD-DFT energies using the default PCM procedure, and equilibrium is the default for calculations using the external iteration approach (SCRF=ExternalIteration). See the examples for the method for computing non-equilibrium external iteration calculations.

For EOM-CCSD calculations in solution, the solvation approaches of Cammi and Caricato are available [360, 366, 367]. The PTED method is available with the PTED option. Other methods discussed in [360] are also available via IOp(9/116).

By default, CASSCF PCM [695] calculations correspond to equilibrium calculations with respect to the solvent reaction field/solute electronic density polarization process. Calculation of non equilibrium solute-solvent interaction involving two different electronic states (e.g. the initial and final states of a vertical transition) can be performed using the NonEq=*type* PCM keyword, in two separate job steps (see *PCM* in the *Input* section).

5.73.1 Input

Keywords and options specifying details for a PCM calculation (i.e., the default SCRF=PCM or SCRF=CPCM) may be specified in an additional blank-line terminated input section provided that the Read option is also specified. Keywords within this section follow general Gaussian input rules. The available keywords are listed in a separate subsection following the examples.

For the Onsager model (SCRF=Dipole), the solute radius in Angstroms and the dielectric constant of the solvent are read as two free-format real numbers on one line from the input stream. A suitable solute radius is computed by a gas-phase molecular volume calculation (in a separate job step); see the discussion of the Volume keyword.

For the IPCM and SCIPCM models, the input consists of a line specifying the dielectric constant of the solvent and an optional isodensity value (the default for the latter is 0.0004).

5.73.2 Options

Specifying the Solvent

Solvent=*item*

Selects the solvent in which the calculation is to be performed. Note that the solvent may also be specified in the input stream in various ways for the different SCRF methods. If unspecified, the solvent defaults to water. *item* is a solvent name chosen from the list at the end of this section.

Selecting the SCRF Method

PCM

Performs a reaction field calculation using the integral equation formalism model (IEFPCM). This is the default. Some details of the formalism and the implementation have changed with respect to Gaussian 03, as described in [701]. IEFPCM is a synonym for PCM.

When PCM is used for an anisotropic or ionic solvent, then items in the PCM input section must be used to select the anisotropic and ionic dielectric models for these types of solvents, using the Read option. The continuous surface charge formalism is also not available with such solvents, and no derivatives can be computed.

CPCM

Performs a PCM calculation using the CPCM polarizable conductor calculation model [410, 691].

Dipole

Performs an Onsager model reaction field calculation.

IPCM

Performs an IPCM model reaction field calculation. Isodensity is a synonym for IPCM.

SCIPCM

Performs an SCIPCM model reaction field calculation, i.e. the SCRF calculation uses a cavity determined self-consistently from an isodensity surface.

SMD Model

SMD

Do an IEFPCM calculation with radii and non-electrostatic terms for Truhlar and coworkers' SMD solvation model [706]. This is the recommended choice for computing ΔG of solvation, which accomplished by performing gas phase and SCRF=SMD calculations for the system of interest and taking the difference the resulting energies. You can define a new solvent for use with SMD by providing additional input via the SCRF=Read option (see "Additional Input" for details).

PCM Non-Equilibrium Solvation

NonEquilibrium=*action*

Save or retrieve data for non-equilibrium solvation. *action* is one of the following:

◊ Save or Write: Save the slow/inertial charges in the checkpoint file (from which they will be retrieved in a subsequent non-equilibrium solvation calculation).

- ◊ Read or Load: Read the slow/inertial charges for use in the current non-equilibrium solvation calculation.
- ◊ CCSave or CCWrite: Save the slow/inertial correlation charges for use in a subsequent non-equilibrium solvation coupled cluster calculation.
- ◊ CCRead or CCLoad: Retrieve the slow/inertial correlation charges for use in the current non-equilibrium solvation coupled cluster calculation.

External Iteration PCM

ExternalIteration

Does a self-consistent PCM calculation performing an external iteration through Link 124. This approach computes the energy in solution by making the solute's electrostatic potential self-consistent with the solvent reaction field [715, 716]. ExternalIteration is available only for energy calculations. SelfConsistent and SC are synonyms for this option.

1stVac

Do the first iteration in an external iteration PCM calculation in solution. 1stVac is equivalent to DoVacuum, which is now deprecated.

1stPCM

Do not do the first iteration in an external iteration PCM calculation in solution. 1stPCM is equivalent to SkipVacuum and NoVacuum, which are now deprecated.

Restart

Restarts a PCM external iteration calculation from the checkpoint file.

IEFPCM and CPCM Other Options

SolventAccessibleSurface

For PCM, use a cavity representing the solvent-accessible surface. Only suitable for single-point calculations, but useful for cases with unusual cavities, snapshots from MD with explicit solvent, etc. SCRF=SAS is a synonym for this option.

AsymmetricIEFPCM

Perform asymmetric isotropic IEFPCM rather than the default of symmetric isotropic IEFPCM. AsymmetricIEFPCM was the default in Gaussian 09.

SCRF=AIEFPCM is a synonym for this option. Not valid with CPCM.

PTED

Use the perturbation theory and density approach (PTED) to PCM-CCSD coupling [360, 366] for an EOM-CCSD calculation in solution. Other schemes discussed in [360] are available via IOp(9/116).

CorrectedLinearResponse

Do a state-specific correction to the energy of a CIS/RPA/TD-DFT SCRF excited state, according to [718]. CorrectedLR is a synonym for this option.

Read

Indicates that a separate section of keywords and options providing calculation parameters should be read from the input stream. This must be specified for anisotropic and ionic solvents.

Checkpoint

Retrieves the SCRF information from the checkpoint file.

Modify

Retrieves the SCRF information from the checkpoint file, and also reads modifications from the input stream.

ONIOMPCM=*k*

Performs an ONIOM calculation in solution [719, 720] according to the scheme selected by the code letter *k*, for which these are the valid values:

A	The reaction field is computed self-consistently using the integrated ONIOM density (available only for energies, and not available for semi-empirical methods).
B	The reaction field is computed for the real-system at the low-level and the corresponding polarization charges are used as external charges in the sub-calculations on the model-systems (available only for energies).
C	The reaction field is computed only for the real-system at the low-level while the sub-calculations on the model-systems are performed assuming zero reaction field (i.e., gas phase). This selection is available for energies, optimizations and frequencies.
X	The reaction field is computed separately in each sub-calculation always using the cavity of the real-system. This is the default if ONIOM and SCRF are specified (available for energies, optimizations and frequencies).

G09Defaults

Sets defaults for PCM solvation back to those used in Gaussian 09.

G03Defaults

Modify the PCM defaults in order to reproduce the results of a Gaussian 03 PCM calculation as closely as possible. Note that perfect agreement is not always possible due to improvements in the program.

Onsager (SCRF=Dipole) Model Options

A0=*val*

Sets the value for the solute radius a_0 in the route section (rather than reading it from the input stream of an SCRF=Dipole calculation). If this option is included, then Solvent or Dielectric must also be included.

Dielectric=*val*

Sets the value for the dielectric constant of the solvent. This option overrides Solvent if both are specified.

IPCM Model Options

GradVne

Use Vne basins for the numerical integration.

GradRho

Use density basins for the numerical integration. The job may fail if non-nuclear attractors are present.

SCIPCM Model Options

UseDensity

Force the use of the density matrix in evaluating the density.

UseMOs

Force the use of MOs in evaluating the density.

GasCavity

Use the gas phase isodensity surface to define the cavity rather than solving for the surface self-consistently. This is mainly a debugging option.

Save/Retrieve PCM Charges

SaveQ

Write the PCM charges on the checkpoint file. SaveQ can be specified together with LoadQ. WriteQ is a synonym for this option.

LoadQ

Read the PCM charges on the checkpoint file. Note that the molecular geometry and the cavity must match exactly to use the saved charges. SaveQ can be specified together with LoadQ. ReadQ is a synonym for this option.

Produce Input for COSMO/RS

COSMORS

Produces the data file used by COSMO/RS and other programs.

5.73.3 Availability

The following table details the availability of the various SCRF=PCM calculation types by theoretical method:

Method	Energy (Ext. Iter.)	Energy	Opt	Freq	3rd Order Props[a]	NMR
MM	no	yes	yes	yes	no	no
AM1, PM3, PM3MM, PM6, PDDG	no	yes	yes	yes	no	no
HF, DFT	yes	yes	yes	yes	yes	yes
MP2	yes	yes[b]	yes[b]	yes[b]	yes[b,c]	yes[b]
MP3, MP4(SDQ), CCSD, QCISD	yes	yes[b]	no	no	no	no
CASSCF	yes	yes	yes	yes[e]	no	no
CIS	yes	yes[d]	yes[d]	yes[d]	no	no
TD	yes	yes[d]	yes[d]	yes[d]	no	no
ZIndo	no	yes	no	no	no	no

[a] For example, Freq=Raman, ROA or VCD;

[b] Computed via SCF MO polarization;

[c] Raman intensities are computed numerically (i.e., as with Freq=NRaman);

[d] Using the linear response approach;

[e] Numerical frequencies only.

CASSCF frequencies with PCM solvation must be done numerically using Freq=Numer.

Restarting SCRF Calculations. SCRF=ExternalIteration and SCRF=IPCM jobs can be restarted from the read-write file by using the Restart option. SCRF=SCIPCM calculations that fail during the SCF iterations should be restarted via the SCF=Restart keyword.

Non-Default Methods. The IPCM model is available for HF, DFT, MP2, MP3, MP4(SDQ), QCISD, CCD, CCSD, CID, and CISD energies only. The SCIPCM model is available for HF and DFT energies and optimizations and numerical frequencies. The Onsager (SCRF=Dipole) model is available for HF, DFT, MP2, MP3, MP4(SDQ), QCISD, CCD, CCSD, CID, and CISD energies, and for HF and DFT optimizations and frequency calculations. However, the Opt Freq keyword combination may not be used in SCRF=Dipole calculations.

5.73.4 Related Keywords

SCF, Volume

5.73.5 Examples

PCM Energy. In general, energy output from the default SCRF method appears in the normal way within the output file. For example, here are the sections of the output file containing the predicted energy from a Hartree-Fock and from an MP2 PCM calculation:

Hartree-Fock SCRF calculation:

```
SCF Done:   E(RHF) =   -99.4687828290      A.U. after     8 cycles
                 Convg   =    0.2586D-08              -V/T =  2.0015
```

MP2 SCRF calculation:

```
E2 =     -0.1192799427D+00   EUMP2 =      -0.99584491345297D+02
```

The predicted energy in solution includes all computed corrections (unlike in Gaussian 03 output).

Additional output lines may appear when various PCM options are included. For example, the following output is produced by an HF SCRF=SMD calculation:

```
SCF Done:  E(RHF) =  -99.4687828290     A.U. after    8 cycles
                 Convg   =    0.2586D-08            -V/T =  2.0015
SMD-CDS (non-electrostatic) energy            (kcal/mol) =       0.54
 (included in total energy above)
```

For external iteration SCRF calculations, the final energy is computed by Link 124, which controls the external iteration, and is reported in a separate output section, which will appear very near the end of the output file, as in the following example:

```
-----------------------------------------------------------------
Self-consistent PCM results
==========================
<psi(f)|   H    |psi(f)>                    (a.u.) =     -99.577537   (A)
<psi(f)|H+V(f)/2|psi(f)>                    (a.u.) =     -99.584002   (B)
(Polarized solute)-Solvent               (kcal/mol) =      -4.06       (C)
-----------------------------------------------------------------
Partition over spheres:
Sphere   on Atom   Surface   Charge   GEl     GCav    GDR
   1       H1       15.27   -0.157   -2.36    0.00    0.00
   2       F2       32.58    0.157   -1.70    0.00    0.00
-----------------------------------------------------------------
```

Predicted energy value for external iteration and state-specific SCRF calculations

```
After PCM corrections, the energy is  -99.5840023899     a.u.
```

```
----------------------------------------------------------------------
```

Line (A) reports the energy computed using the polarized solute wavefunction and the gas phase Hamiltonian, line (B) reports energy computed using the polarized solute wavefunction and the Hamiltonian in solution, line (C) reports the interaction energy between the polarized solute and the solvent, which corresponds to the integral $<\Psi(f)|V(f)/2|\Psi(f)>$ (in kcal/mol), and the final line reports the predicted energy incorporating all PCM corrections.

Fluorescence example: Emission (Fluorescence) from First Excited State ($n \rightarrow \pi^*$) of Acetaldehyde

Here we study the cycle:

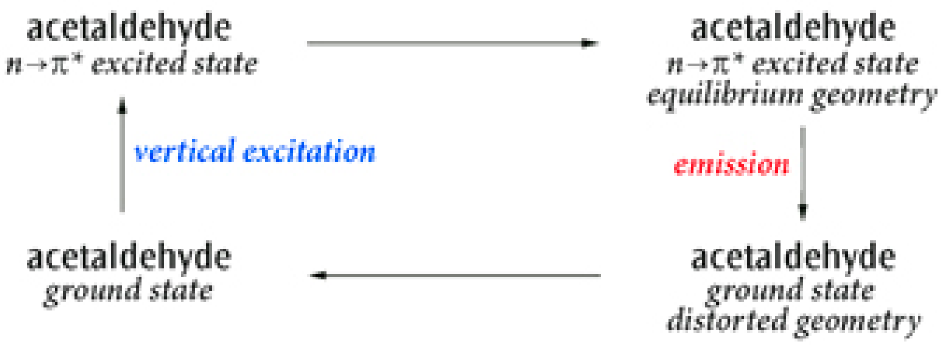


Figure 5.4: Acetaldehyde Excitation and Emission Cycle

The primary process of interest is the emission, but this example shows how to study the complete cycle including the solvent effects.

Step 1: Ground state geometry optimization and frequencies (equilibrium solvation). This is a standard Opt Freq calculation on the ground state including PCM equilibrium solvation.

```
%chk=01-ac
# B3LYP/6-31+G(d,p) Opt Freq SCRF=(Solvent=Ethanol)

Acetaldehyde ground state

0 1
C
C,1,RA
X,2,1.,1,A
O,2,RB,3,A,1,180.,0
X,1,1.,2,90.,3,0.,0
H,1,R1,2,A1,5,0.,0
H,1,R23,2,A23,5,B23,0
H,1,R23,2,A23,5,-B23,0
H,2,R4,1,A4,3,180.,0

RA=1.53643
RB=1.21718
R1=1.08516
R23=1.08688
R4=1.10433
```

```
A=62.1511
A1=110.51212
A23=109.88119
A4=114.26114
B23=120.56468
```

Here is the energy of the ground state optimized geometry in solution:

```
SCF Done:  E(RB3LYP) =  -153.851761719     A.U. after    1 cycles
```

Step 2: Vertical excitation with linear response solvation. This is a TD-DFT calculation of the vertical excitation, therefore at the ground state equilibrium geometry, with the default solvation: linear response, non-equilibrium. We perform a single-point TD-DFT calculation, which defaults to non-equilibrium solvation. The results of this job will be used to identify which state or states are of interest and their ordering. These results give a reasonable description of the solvation of the excited state, but not quite as good as that from a state-specific solvation calculation. In this case, we see that the $n \to \pi^*$ state is the first excited state. Next, we will use the state-specific method to produce a better description of the vertical excitation step.

```
%oldchk=01-ac
%chk=02-ac
# B3LYP/6-31+G(d,p) TD=NStates=6 SCRF=(Solvent=Ethanol) Geom=Check Guess=Read

Acetaldehyde: linear response vertical excited states

0 1
```

The vertical excitation (absorption) to first excited state from the non-equilibrium solvation linear response calculation:

```
Excited State 1: Singlet-A"  4.3767 eV  283.28 nm  f=0.0000  <S**2>=0.000
```

Thus, the ground state to first excited state absorption is at 283.28 nm, computed via the linear-response approach.

Step 3: State-specific solvation of the vertical excitation. This will require two job steps: first the ground state calculation is done, specifying the NonEquilibrium=Save option, in order to store the information about non-equilibrium solvation based on the ground state. Second, the actual state-specific calculation is done, reading in the necessary information for non-equilibrium solvation using NonEquilibrium=Read option, and specifying the checkpoint file from Step 1:

```
%oldchk=01-ac
%chk=03-ac
# B3LYP/6-31+G(d,p) SCRF=(Solvent=Ethanol,NonEquilibrium=Save)
  Geom=Check Guess=Read

Acetaldehyde: prepare for state-specific non-eq solvation
by saving the solvent reaction field from the ground state

0 1

--link1--
%chk=03-ac
```

```
# B3LYP/6-31+G(d,p) TD(NStates=6,Root=1) Geom=Check Guess=Read
  SCRF=(Solvent=Ethanol,ExternalIteration,NonEquilibrium=Read)

Acetaldehyde: read non-eq solvation from ground state and
compute energy of the first excited with the state-specific method

0 1
```

Here is the energy of first excited state – at the ground state optimized geometry – from the non-equilibrium solvation state-specific calculation:

```
After PCM corrections, the energy is  -153.687679826     a.u.
```

Subtracting this energy from the ground state energy (from step 1) gives the ground state to first excited state absorption including the state-specific solvation correction: at 277.69 nm.

Step 4: Relaxation of the excited state geometry. Next, we perform a TD-DFT geometry optimization, with equilibrium, linear response solvation, in order to find the minimum energy point on the excited state potential energy surface. Since this is a TD-DFT optimization, the program defaults to equilibrium solvation. As is typical of such cases, the molecule has a plane of symmetry in the ground state but the symmetry is broken in the excited state, so the ground state geometry is perturbed slightly to break symmetry at the start of the optimization. We retrieve the geometry and other data from the checkpoint file from Step 2:

```
%oldchk=02-ac
%chk=04-ac
# B3LYP/6-31+G(d,p) TD=(Read,NStates=6,Root=1) SCRF=(Solvent=Ethanol)
  Geom=Modify Guess=Read Opt

Acetaldehyde: excited state opt
Modify geometry to break Cs symmetry
since first excited state is A"

0 1

4 1 2 3 10.0
5 1 2 7 -50.0
```

Here are the results for the first excited state after the geometry optimization of first excited state in solution (equilibrium geometry):

```
Excited State 1: Singlet-A  3.2074 eV  386.55 nm  f=0.0013  <S**2>=0.0
This state for optimization and/or second-order correction.
Total Energy, E(TD-HF/TD-KS) =  -153.705918726
```

Step 5: Vibrational frequencies of the excited state structure. Now we run a frequency calculation to verify that the geometry located in step 4 is a minimum. The results could also be used as part of a Franck-Condon calculation if desired. This is a numerical frequency calculation.

```
%oldchk=04-ac
%chk=05-ac
# B3LYP/6-31+G(d,p) TD=(Read,NStates=6,Root=1) Freq
  SCRF=(Solvent=Ethanol) Geom=Check Guess=Read
```

```
Acetaldehyde excited state freq

0 1
```

The frequency calculation is used to confirm that the geometry optimized in Step 4 is a minimum on the excited state potential energy surface.

Step 6: Emission state-specific solvation (part 1). This step does state-specific equilibrium solvation of the excited state at its equilibrium geometry, writing out the solvation data for the next step via the NonEquilibrium=Save option.

```
%oldchk=05-ac
%chk=06-ac
# B3LYP/6-31+G(d,p) TD=(Read,NStates=6,Root=1) Geom=Check Guess=Read
  SCRF=(Solvent=Ethanol,ExternalIteration,NonEquilibrium=Save)

Acetaldehyde emission state-specific solvation
at first excited state optimized geometry

0 1
```

Here is the energy of first excited state – at its optimized geometry – from the equilibrium solvation state-specific calculation:

```
After PCM corrections, the energy is  -153.707148980     a.u.
```

Step 7: Emission to final ground state (part 2). Finally, we compute the ground state energy with non-equibrium solvation, at the excited state geometry and with the static solvation from the excited state.

```
%oldchk=06-ac
%chk=07-ac
# B3LYP/6-31+G(d,p) SCRF=(Solvent=Ethanol,NonEquilibrium=Read)
  Geom=Check Guess=Read

Acetaldehyde: ground state non-equilibrium
at excited state geometry.

0 1
```

Here is the energy of ground state from a non-equilibrium solvation calculation in solution, using the first excited state optimized geometry and the solvent reaction field in equilibrium with the first excited state density):

```
SCF Done:  E(RB3LYP) =  -153.822024722      A.U. after   10 cycles
```

The difference between the energies from steps 6 and 7 gives the vertical emission energy. In this case, the first excited state to ground state emission, including the state-specific solvation correction, is at 396.63 nm.

Steps 1, 2, and 4 would be sufficient to compute the excitation and emission energies in the gas-phase (along with step 5 to confirm the nature of stationary point). They are not sufficient when solvent effects are included because the energies computed in step 4 correspond to the ground state solvent reaction field, while the emission takes place in the reaction field created in response to the excited state charge distribution. This is what is accounted for properly in steps 6 and 7.

If the band shape is to be calculated, then in the gas phase one would simply run a calculation with Freq=(ReadFC,FC,Emission), giving the checkpoint file from step 1 as the main checkpoint file for the job, and providing the name of the checkpoint file from step 5 in the input stream to specify the other state. For the solvated band shape, one must do Freq=(ReadFC,FC,Emission,ReadFCHT) using the checkpoint files for steps 1 and 5, but also providing the state-specific emission energy in the input section for the Franck-Condon calculation.

5.73.6 Additional Input

Additional Input for PCM Calculations

Additional input keywords may be specified for PCM SCRF calculations (including ones using the SMD model). Note that many of the available parameters are designed for expert users; experimenting with modifications to the standard settings is strongly discouraged for production calculations.

These keywords are placed in a separate input section, terminated as usual by a blank line, as in this example which tightens the convergence threshold for the iterative calculation of the PCM polarization charges and sets the electrostatic scaling factor for the cavity sphere radii:

```
# B3LYP/6-31G(d) 5D SCRF(Solvent=Ethanol,Read)

PCM extra input example.

0 1
molecule specification

QConv=10             Input section for PCM keywords
Alpha=1.1
                     Blank line terminates PCM input
```

The PCM input section ends as usual with a blank line.

The available keywords for controlling PCM calculations are listed below, arranged in groups of related items.

Defining Solvent Parameters

The solvent for the PCM calculation is generally specified using the normal Solvent option to the SCRF keyword. You can use the following keywords to override some default values for known solvents.

Eps=*x*	Specifies the static (or zero-frequency) dielectric constant of the solvent.
EpsInf=*x*	Specifies the dynamic (or optical) dielectric constant of the solvent. For SMD calculations, it should be set to the square of the solvent's refractive index at 293 K.
RSolv=*x*	Specifies the solvent radius (in Angstroms). Relevant only when using AddSph or Surface=SAS.

Unspecified parameters default to the values for the solvent specified with the Solvent option (or to water if this option is omitted).

Defining Solvents for SMD Calculations

The items needed to define a solvent differ for the SMD method. Here is an example of an SMD calculation that defined ethanol manually via a PCM input section. Note that the final five items are SMD-specific keywords. Note that all 7 items are required and that all values must be specified as real numbers. For full details about these parameters, see [706]. As of this writing, the vast majority of the solvents in the *Minnesota Solvent Descriptor Database* have solvent keywords defined (see "Solvents").

```
# B3LYP/6-31G(d) 5D SCRF(SMD,Solvent=Generic,Read)

Water in solution with ethanol, defined as generic SMD solvent.

0 1
O
H,1,0.94
H,1,0.94,2,104.5
```

	Input section for PCM keywords
`Eps=24.852`	ε
`EpsInf=1.852593`	n^2
`HbondAcidity=0.37`	α
`HbondBasicity=0.48`	β
`SurfaceTensionAtInterface=31.62`	γ
`CarbonAromaticity=0.`	ϕ
`ElectronegativeHalogenicity=0.`	ψ
	Blank line terminates PCM input

Calculation Method Variations

NonEq=*item*	Compute and save the non-equilibrium reaction field after the completion of an HF, DFT or CASSCF calculation using SCRF(Read), or at the end of any SCRF(ExternalIteration,Read) calculation. NonEq=Write says to save the data in the checkpoint file. Use NonEq=Read to retrieve it from the checkpoint file in a subsequent calculation.
Dis	Computes and includes in the total energy the solute-solvent dispersion interaction energy using the model of Floris and Tomasi [721, 722]. The default is NoDis. This option cannot be used in a SCRF=SMD calculation.
Rep	Includes the solute-solvent repulsion interaction energy in the total energy using the model of Floris and Tomasi [721, 722]. The default is NoRep. This option cannot be used in a SCRF=SMD calculation.
Cav	Includes the solute cavitation energy in the total energy using the model of Pierotti [723]. The default is NoCav. This option cannot be used in a SCRF=SMD calculation.

CavityFieldEffects	Includes the effects of the cavity-field interaction energy (also known as local field effect) in the total energy according to the model of Cammi and co-workers [724]. The default is not to include this effect.
CF=Eps=*x*	Specifies a different value for the static dielectric constant to be used in the cavity-field energy contribution. This is useful to modulate the magnitude of the cavity-field effects.
CF=EpsInf=*x*	Specifies a different value for the dynamic (optical) dielectric constant to be used in the cavity-field energy contribution. This is useful to modulate the magnitude of the cavity-field effects.
FitPot	Performs analysis of the solute-solvent interaction energy in terms of atomic or atomic groups additive contributions. This analysis involves a fitting of atomic charges to the molecular electrostatic potential in solution.
Iterative	Solves the PCM electrostatic problem to calculate polarization charges through an iterative method.
MxIter=*N*	Specifies the maximum number of iterations allowed to the iterative solution of the electrostatic problem. 400 is the default.
QConv=*type*|*N*	Sets the convergence threshold for the iterative calculations of the PCM polarization charges to 10^{-N} or to one of the following predefined types: VeryTight (10^{-12}), Tight (10^{-9}) and Sleazy (10^{-6}). The default is QConv=Tight.
SC=QConv=*x*	Specifies the convergence for the PCM polarization charges during the external iteration procedure.
MaxExtIt=*x*	Specify the maximum number of iterations allowed during the external iteration procedure.

Anisotropic and Ionic Solvents

Anisotropic	Performs a PCM calculation for an anisotropic solvent according to the IEFPCM formalism. The 3-rank symmetric tensor representing the dielectric constant must be specified as the values for these six additional keywords: EPSX, EPSY, EPSZ, EUPHI, EUTHE, and EUPSI (all of them take a parameter: e.g., EPSX=*value*).
Ionic	Performs a PCM calculation for ionic solution according to the IEFPCM formalism. The ionic strength in mol/dm^3 has to be specified as the value to the keyword DISM.

Specifying the Molecular Cavity

By default, the program builds up the cavity using the UFF radii, which places a sphere around each solute atom, with the radii scaled by a factor of 1.1. There are also three United Atom (UA) models available.

The cavity can be extensively modified in the PCM input section: changing sphere parameters and the general cavity topology, adding extra spheres to the cavity built by default, and so on. The whole molecular cavity can be also provided by the user in the input section.

Radii=*model* : Indicates the topological model and/or the set of atomic radii used. Available models and sets are:

◊ UFF: Uses radii from the UFF force field. Hydrogens have individual spheres (explicit hydrogens).

This is the default.

- ◊ UA0: Uses the United Atom Topological Model applied on atomic radii of the UFF force field for heavy atoms. Hydrogens are enclosed in the sphere of the heavy atom to which they are bonded. This was the default in Gaussian 03.
- ◊ UAHF: Uses the United Atom Topological Model applied on radii optimized for the HF/6-31G(d) level of theory.
- ◊ UAKS: Uses the United Atom Topological Model applied on radii optimized for the PBE1PBE/6-31G(d) level of theory.
- ◊ Pauling: Uses the Pauling (actually Merz-Kollman) atomic radii (uses explicit hydrogens).
- ◊ Bondi: Uses the Bondi atomic radii (uses explicit hydrogens).

Surface=*type* : Specify the type of molecular surface representing the solute-solvent boundary. Available options are:

- ◊ VDW: Van der Waals surface. Uses atomic radii (scaled) and skips the generation of "added spheres" to smooth the surface. This is the default.
- ◊ SES: Solvent Excluding Surface. The surface is generated by the atomic or group spheres and by the spheres created automatically to smooth the surface ("added spheres"). This was the default in Gaussian 03.
- ◊ SAS: Solvent Accessible Surface. The radius of the solvent is added to the unscaled radii of atoms and/or atomic groups.

ModifySph : Alters parameters for one or more spheres. The modified spheres can be indicated in the PCM input in lines following this keyword having the following format:

atom radius [alpha]

where *atom* is the atom number or element type.

ExtraSph=*N* : Adds *N* user-defined spheres to the cavity. Parameters of the spheres can be specified in lines following this keyword using the following format:

X Y Z radius [alpha]

where *X*,*Y*,*Z* are the Cartesian coordinates in the standard orientation.

NSph=*N* : The cavity is built just from the *N* spheres provided by the user, specified on lines of the following format:

atom_number radius [alpha] **– or –** *X Y Z radius [alpha]*

where *X*,*Y*,*Z* are the Cartesian coordinates in the standard orientation. Specifying spheres by atom number mimics standard cavity behavior, while specifying Cartesian produces a fixed cavity which does not move with the structure.

PDens=*x* Sets the average density of integration points on the surface, in units of Angstrom^{-2}. 5.0 is the default. Increasing this value results in a finer surface discretization.

Alpha=*scale* Specifies the electrostatic scaling factor by which the sphere radius is multiplied. The default value is 1.1.

SphereOnH=*N* When using a United Atom Topological model, places an individual sphere on the hydrogen at the *Nth* position in the atoms list.

SphereOnAcidicHydrogens When using a United Atom Topological model, puts individual spheres on acidic hydrogens (those bonded to N, O, S, P, Cl and F atoms).

OFac=*x* Specifies the overlap index between two interlocking spheres *Pascual-Ahuir94* for SES added spheres. Decreasing this index results in a smaller number of added spheres. The default value is 0.89.

RMin=*x* Sets the minimum radius in Angstroms for SES added spheres. Increasing this value results in a smaller number of added spheres. The default value is 0.2.

GeomView Create the file *points.off* describing the cavity. This file contains input for the *GeomView* program (see `www.geomview.org`), which can be used to visualize the molecular cavity.

5.73.7 Solvents

The following solvent keywords are accepted with the SCRF=Solvent option. We list the ε values here for convenience, but be aware it is only one of many internal parameters used to define solvents. Thus, simply changing the ε value will not define a new solvent properly.

◊Water: ε=78.3553
◊Acetonitrile: ε=35.688
◊Methanol: ε=32.613
◊Ethanol: ε=24.852
◊IsoQuinoline: ε=11.00
◊Quinoline: ε=9.16
◊Chloroform: ε=4.7113
◊DiethylEther: ε=4.2400
◊Dichloromethane: ε=8.93
◊DiChloroEthane: ε=10.125
◊CarbonTetraChloride: ε=2.2280
◊Benzene: ε=2.2706
◊Toluene: ε=2.3741
◊ChloroBenzene: ε=5.6968
◊NitroMethane: ε=36.562
◊Heptane: ε=1.9113
◊CycloHexane: ε=2.0165
◊Aniline: ε=6.8882
◊Acetone: ε=20.493
◊TetraHydroFuran: ε=7.4257
◊DiMethylSulfoxide: ε=46.826
◊Argon: ε=1.430
◊Krypton: ε=1.519
◊Xenon: ε=1.706
◊n-Octanol: ε=9.8629
◊1,1,1-TriChloroEthane: ε=7.0826
◊1,1,2-TriChloroEthane: ε=7.1937
◊1,2,4-TriMethylBenzene: ε=2.3653
◊1,2-DiBromoEthane: ε=4.9313
◊1,2-EthaneDiol: ε=40.245
◊1,4-Dioxane: ε=2.2099
◊1-Bromo-2-MethylPropane: ε=7.7792
◊1-BromoOctane: ε=5.0244
◊1-BromoPentane: ε=6.269
◊1-BromoPropane: ε=8.0496
◊1-Butanol: ε=17.332
◊1-ChloroHexane: ε=5.9491
◊1-ChloroPentane: ε=6.5022
◊2-Heptanone: ε=11.658
◊2-Hexanone: ε=14.136
◊2-MethoxyEthanol: ε=17.2
◊2-Methyl-1-Propanol: ε=16.777
◊2-Methyl-2-Propanol: ε=12.47
◊2-MethylPentane: ε=1.89
◊2-MethylPyridine: ε=9.9533
◊2-NitroPropane: ε=25.654
◊2-Octanone: ε=9.4678
◊2-Pentanone: ε=15.200
◊2-Propanol: ε=19.264
◊2-Propen-1-ol: ε=19.011
◊3-MethylPyridine: ε=11.645
◊3-Pentanone: ε=16.78
◊4-Heptanone: ε=12.257
◊4-Methyl-2-Pentanone: ε=12.887
◊4-MethylPyridine: ε=11.957
◊5-Nonanone: ε=10.6
◊AceticAcid: ε=6.2528
◊AcetoPhenone: ε=17.44
◊a-ChloroToluene: ε=6.7175
◊Anisole: ε=4.2247
◊Benzaldehyde: ε=18.220
◊BenzoNitrile: ε=25.592
◊BenzylAlcohol: ε=12.457
◊BromoBenzene: ε=5.3954
◊BromoEthane: ε=9.01
◊Bromoform: ε=4.2488
◊Butanal: ε=13.45
◊ButanoicAcid: ε=2.9931
◊Butanone: ε=18.246
◊ButanoNitrile: ε=24.291
◊ButylAmine: ε=4.6178
◊ButylEthanoate: ε=4.9941
◊CarbonDiSulfide: ε=2.6105
◊Cis-1,2-DiMethylCycloHexane: ε=2.06
◊Cis-Decalin: ε=2.2139
◊CycloHexanone: ε=15.619
◊HexanoicAcid: ε=2.6
◊IodoBenzene: ε=4.5470
◊IodoEthane: ε=7.6177
◊IodoMethane: ε=6.8650
◊IsoPropylBenzene: ε=2.3712
◊m-Cresol: ε=12.44
◊Mesitylene: ε=2.2650
◊MethylBenzoate: ε=6.7367
◊MethylButanoate: ε=5.5607
◊MethylCycloHexane: ε=2.024
◊MethylEthanoate: ε=6.8615
◊MethylMethanoate: ε=8.8377
◊MethylPropanoate: ε=6.0777
◊m-Xylene: ε=2.3478
◊n-ButylBenzene: ε=2.36
◊n-Decane: ε=1.9846
◊n-Dodecane: ε=2.0060
◊n-Hexadecane: ε=2.0402
◊n-Hexane: ε=1.8819
◊NitroBenzene: ε=34.809
◊NitroEthane: ε=28.29
◊n-MethylAniline: ε=5.9600
◊n-MethylFormamide-mixture: ε=181.56
◊n,n-DiMethylAcetamide: ε=37.781
◊n,n-DiMethylFormamide: ε=37.219
◊n-Nonane: ε=1.9605
◊n-Octane: ε=1.9406
◊n-Pentadecane: ε=2.0333
◊n-Pentane: ε=1.8371
◊n-Undecane: ε=1.9910
◊o-ChloroToluene: ε=4.6331
◊o-Cresol: ε=6.76
◊o-DiChloroBenzene: ε=9.9949
◊o-NitroToluene: ε=25.669
◊o-Xylene: ε=2.5454
◊Pentanal: ε=10.0
◊PentanoicAcid: ε=2.6924
◊PentylAmine: ε=4.2010

◊1-ChloroPropane: ε=8.3548
◊1-Decanol: ε=7.5305
◊1-FluoroOctane: ε=3.89
◊1-Heptanol: ε=11.321
◊1-Hexanol: ε=12.51
◊1-Hexene: ε=2.0717
◊1-Hexyne: ε=2.615
◊1-IodoButane: ε=6.173
◊1-IodoHexaDecane: ε=3.5338
◊1-IodoPentane: ε=5.6973
◊1-IodoPropane: ε=6.9626
◊1-NitroPropane: ε=23.73
◊1-Nonanol: ε=8.5991
◊1-Pentanol: ε=15.13
◊1-Pentene: ε=1.9905
◊1-Propanol: ε=20.524
◊2,2,2-TriFluoroEthanol: ε=26.726
◊2,2,4-TriMethylPentane: ε=1.9358
◊2,4-DiMethylPentane: ε=1.8939
◊2,4-DiMethylPyridine: ε=9.4176
◊2,6-DiMethylPyridine: ε=7.1735
◊2-BromoPropane: ε=9.3610
◊2-Butanol: ε=15.944
◊2-ChloroButane: ε=8.3930
◊CycloPentane: ε=1.9608
◊CycloPentanol: ε=16.989
◊CycloPentanone: ε=13.58
◊Decalin-mixture: ε=2.196
◊DiBromomEthane: ε=7.2273
◊DiButylEther: ε=3.0473
◊DiEthylAmine: ε=3.5766
◊DiEthylSulfide: ε=5.723
◊DiIodoMethane: ε=5.32
◊DiIsoPropylEther: ε=3.38
◊DiMethylDiSulfide: ε=9.6
◊DiPhenylEther: ε=3.73
◊DiPropylAmine: ε=2.9112
◊e-1,2-DiChloroEthene: ε=2.14
◊e-2-Pentene: ε=2.051
◊EthaneThiol: ε=6.667
◊EthylBenzene: ε=2.4339
◊EthylEthanoate: ε=5.9867
◊EthylMethanoate: ε=8.3310
◊EthylPhenylEther: ε=4.1797
◊FluoroBenzene: ε=5.42
◊Formamide: ε=108.94
◊FormicAcid: ε=51.1
◊PentylEthanoate: ε=4.7297
◊PerFluoroBenzene: ε=2.029
◊p-IsoPropylToluene: ε=2.2322
◊Propanal: ε=18.5
◊PropanoicAcid: ε=3.44
◊PropanoNitrile: ε=29.324
◊PropylAmine: ε=4.9912
◊PropylEthanoate: ε=5.5205
◊p-Xylene: ε=2.2705
◊Pyridine: ε=12.978
◊sec-ButylBenzene: ε=2.3446
◊tert-ButylBenzene: ε=2.3447
◊TetraChloroEthene: ε=2.268
◊TetraHydroThiophene-s,s-dioxide: ε=43.962
◊Tetralin: ε=2.771
◊Thiophene: ε=2.7270
◊Thiophenol: ε=4.2728
◊trans-Decalin: ε=2.1781
◊TriButylPhosphate: ε=8.1781
◊TriChloroEthene: ε=3.422
◊TriEthylAmine: ε=2.3832
◊Xylene-mixture: ε=2.3879
◊z-1,2-DiChloroEthene: ε=9.2

5.74 Semi-Empirical Methods

There are a variety of semi-empirical methods available in Gaussian 16. The AM1 and the PM3 methods have been reimplemented [725–727] to use the standard integral processing infrastructure (rather than using code from the public-domain MOPAC). In addition to increased efficiency, this change also provides analytic gradients and frequencies. PM6 and PDDG are also implemented in this way. The remaining semi-empirical methods use the modified version of MOPAC in Link 402, and they are discussed on their individual pages.

◊ AM1: Requests a semi-empirical calculation using the AM1 Hamiltonian [519, 521, 522, 527, 728–734].

◊ PM3: Requests a semi-empirical calculation using the PM3 Hamiltonian [735, 736]. The parameter for Li has been updated as specified in [734]. PM3MM specifies the PM3 model including the optional molecular mechanics correction for HCON linkages.

◊ PM6: Requests a semi-empirical calculation using the PM6 Hamiltonian [737]. The PDDG variation is also available [340, 738–741].

◊ PM7: Requests a semi-empirical calculation using the PM7 Hamiltonian as modified by Throssel and Frisch for continuous potential energy surfaces [742]. PM7R6 is a synonym. The original PM7 method of Stewart [743] can also be requested with the PM7MOPAC keyword.

Standard parameters for supported atoms are generated automatically by the program unless the NoGenerate option is specified. Additional and/or alternate ones can also be read-in in several ways (see the *options*). Read-in parameters take precedence over internal ones when both are used.

Using PM7 with Third-Row and Higher Elements

For some systems containing these elements, instabilities can exists in the PM7 wavefunction. There may be difficulties getting the wavefunction to converge. This is less common for lighter elements and is most likely to occur for transition metal complexes. In such cases, the first option is to try an alternate SCF algorithm: e.g., SCF=YQC. If this fails, other options are to use a different initial guess or read in the density from a calculation using a different method (see the Guess keyword).

In some cases, the PM7 wavefunction that is found may not be the lowest energy SCF solution. This too is most common for transition metals and so caution is advised when performing calculations on transition metals. A straightforward way of testing the wavefunction is to use the Stable keyword.

5.74.1 Options

Generate

Generate the standard parameters for the specified method. This is the default. NoGenerate says not to generate any standard parameters; all parameters must be read in.

Input

Read parameters from the input stream in Gaussian's format. Any parameter can be specified or overridden. The input section must be terminated by a blank line. Cards is a synonym for Input.

MOPACExternal

Read parameters from the input stream in MOPAC's external format and units. Most but not all parameters can be changed. The input section must be terminated by a blank line.

Both

Read parameters from the input stream, first in Gaussian's format, followed by more parameters in MOPAC's format. Both input sections must be terminated by a blank line.

Checkpoint

Read parameters from the checkpoint file. Chk and Read are synonymous with Checkpoint.

TCheckpoint

Read parameters from the checkpoint file if present; otherwise generate them.

RWF

Read parameters from the read-write file.

Print

Print parameters used for elements in the current job in Gaussian's format. This is the default if parameters are read from the input file. NoPrint says not to print parameters, and it is the default if standard parameters are used.

PrintAll

Print parameters for all elements (in Gaussian's format), even the ones that are not present in the molecule specification.

Zero

Print all parameters including the ones that are zero. The default is NonZero, which says to print only non-zero parameters.

Old

Use the old MOPAC-based code. Second derivatives are done numerically. Applies only to AM1 and

PM3. The default is New, which says to use the new implementation described above.

5.74.2 Specifying Semi-Empirical Parameters

Semi-empirical parameters can be specified in two different formats, Gaussian and MOPAC, via the Input and MOPACExternal options (respectively). We begin with the native Gaussian semi-empirical parameter format, which is very general.

Here's an example in Gaussian format, for FeCH (actual output may wrap differently):

```
Initial section of global parameters.
Method=40 CoreType=2 PM6R6=0.0000124488 PM6R12=0.0000007621
****
H          Parameters for hydrogen.
PQN=1 NValence=1 F0ss=0.5309794634 ZetaOverlap=1.2686410000 U=-0.4133181193
Beta=-0.3069665271 CoreKO=0.9416560046 KON=0,0,0,0.9416560046 EISol=-0.4133181193
EHeat=0.0830298228
GCore=0.0016794859,0.8557539899,3.3750716603 DCore=1,3,1.8737858033,2.2435870000
****
C          Parameters for carbon.
PQN=2,2 NValence=4 F0ss=0.4900713271 F0sp=0.4236511476 F0pp=0.3644399975 F2pp
    =0.1978513243
G1sp=0.0790832988
ZetaOverlap=2.0475580000,1.7028410000
U=-1.8775102825,-1.4676916178
Beta=-0.5653970441,-0.2745883502 DDN=0,1,0.7535642510 DDN=1,1,0.7192361890
CoreKO=1.0202596487 KON=0,0,0,1.0202596487 KON=1,0,1,1.2918442312 KON=0,1,1,1.0202596487
KON=2,1,1,0.7626764584 EISol=-4.2335803497 EHeat=0.2723305520 DipHyp=1.5070417957
GCore=0.0032154961,0.5881175739,2.5208171825 DCore=1,4,0.2878149911,0.2165060000
DCore=2,3,1.6101301385,3.2139710000 DCore=3,3,1.7155258339,16.1800020000
DCore=4,3,2.2293611369,25.0358790000 DCore=5,3,1.5446719761,1.8748590000
DCore=6,5,1.3831173494,0.8135100000,3.1644797074,9.2800000000
****
Fe          Parameters for iron.
PQN=4,4,3 NValence=8 F0ss=0.2931506917 F0sp=0.2861621092 F0pp=0.2797829041
F0sd=0.3417747898 F0pd=0.3378189937 F0dd=0.5580709105 F2pp=0.1567537881 F2pd
    =0.1236661383
F2dd=0.2945882511 F4dd=0.1921227725 G1sp=0.2072870321 G1pd=0.1102204721 G2sd
    =0.0588483485
G3pd=0.0671224585 Rsppd=0.1364112343 Rsdpp=0.1031169651 Rsddd=0.1510228569
ZetaOverlap=1.4791500000,6.0022460000,1.0807470000
Zeta1C=1.4591520000,1.3926140000,2.1619090000
U=-2.5913804076,-2.3138503107,-3.8083983722
Beta=0.2950096563,-0.0413709206,-0.1288993981 DDN=0,1,0.0896587028 DDN=1,1,0.3534210933
DDN=0,2,1.6776352014 DDN=1,2,0.0796789968 DDN=2,2,1.3085519205 CoreKO=1.2720920000
KON=0,0,0,1.7056074374 KON=1,0,1,0.2359511557 KON=0,1,1,1.7056074374 KON
    =2,1,1,0.4977547907
KON=2,0,2,1.6958541322 KON=1,1,2,0.2947417183 KON=0,2,2,0.8959434914 KON
    =2,2,2,1.2449263774
EISol=-15.6859079709 EHeat=0.1582446241 DipHyp=0.1793070893
DCore=1,3,0.4521755746,0.0251950000 DCore=6,3,2.1121277473,0.3668350000
DCore=7,3,1.3232002016,0.1553420000 DCore=8,3,0.9135254945,0.1364220000
```

```
DCore=9,3,2.2726610620,3.6573500000 DCore=15,3,1.3586804751,0.4312910000
DCore=16,3,0.5233514967,0.0334780000 DCore=17,3,0.6507784269,0.0194730000
DCore=19,3,1.0583544172,6.0000000000 DCore=26,3,1.4397774115,1.8468900000
****
```

The following items are specified in the global section:

Method	An integer corresponding to the desired semi-empirical method. This value should correspond to the method specified in the route section as a check. The values are 8 for AM1, 9 for PM3, 10 for PM3MM, 40 for PM6 and 41 for PDDG.
CoreType	Type of core repulsion terms, where 1 means AM1, PM3, or PDDG, and 2 means PM6.
PeptideFC	Force constant for peptide linkages. Only valid for with PM3MM.
RIJScale	R_{ij} scale factor for the AM1 O-H and N-H bonds.
PM6R6	R6 parameter for the PM6 core repulsion.
PM6R12	R12 parameter for the PM6 core repulsion.

The following items specify the parameters for an element:

PQN	Principal quantum numbers for each shell (s, p, d). Determines which basis functions are used on the element.
NValence	Number of valence electrons.
ZetaOverlap	Slater exponents for basis functions used in the calculation of the overlap contribution to the core Hamiltonian.
Zeta1C	Slater exponents for basis functions used in the computation of those one-center two-electron integrals that were not specified explicitly.
F0*, G*, Rs*	Slater-Condon parameters for one-center two-electron integrals. If any of these items are not specified, they are computed from the Zeta1C exponents. When internal parameters are printed, all values are included regardless of whether they were computed from Zeta1C or a specific value. The full list of these parameters is: F0ss, F0sp, F0pp, F0sd, F0pd, F0dd, F2pp, F2pd, F2dd, F4dd, G1sp, G1pd, G2sd, G3pd, Rsppd, Rsdpp and Rsddd.
U	Diagonal core Hamiltonian matrix elements, one per angular momentum.
Beta	Off-diagonal core Hamiltonian parameters, one per angular momentum.
DDN	Point-charge distance parameters for multipole-approximated two-center two-electron integrals. Each instance has the form *L1*,*L2*,*Value* and applies to charge distributions involving one basis function of angular momentum *L1* and one of angular momentum *L2*. If any are needed but are not specified, they are computed from the Zeta1C exponents.

KON	Klopman-Ohno parameters for two-center two-electron integrals. Required but unspecified items are computed by matching the one-center limit to the one-center integrals given by the Slater-Condon parameters and Zeta1C, and using the specified or defaulted DD values. Each instance is of the form *LT*,*L1*,*L2*,*Value* and applies to the *LT* angular momentum component of the product of functions of angular momentum *L1* and *L2*.
CoreKO	Klopman-Ohno parameter used in nuclear attraction terms. If not specified, the 0,0,0 (L=0 SS) parameter is used.
EHeat	Heat of formation of the isolated atom.
EISol	Energy of the isolated atom. If not provided, it is computed from the other parameters and a standard electronic configuration for the atom.
DipHyp	Dipole moment hybridization parameter.
DCore	Core repulsion parameters. Each instance is of the form *El,IType,Value1,Value2*. Each term specifies the core repulsion between the current element and the element *El*. *IType* specifies the bond type: 1 for usual AM1, 2 for AM1 N-H and O-H, 3 usual PM6, 4 PM6 O-H, 5 PM6 CC triple bond, and 6 PM6 Si-O. There will be one or two parameters values, depending on the specific functional form.

MOPAC-style semi-empirical parameter input. If you request PM6=MOPACExternal or AM1=MOPACExternal, then an input section is read giving parameters in the same formation used by the External keyword in MOPAC. This is less general than the native Gaussian format, but includes the most common parameters. Refer to the MOPAC documentation for details. The units expected by Gaussian in this format are those expected by MOPAC, which are a mixture of atomic and other units.

The following table gives the correspondence between MOPAC External labels and the native Gaussian input:

MOPAC	**Gaussian**
USS,UPP,UDD	U
ZS,ZP,ZD	ZetaOverlap
ZSN,ZPN,ZDN	Zeta1C
BetaS,BetaP,BetaD	Beta
GSS,GPP,$\cdots$,FODD,$\cdots$	F0ss,F0pp, etc.†
DD2	DDN=0,1,*Value*
DD3	DDN=1,1,*Value*
DD4	DDN=0,2,*Value*
DD5	DDN=1,2,*Value*
DD6	DDN=2,2,*Value*
PO1	KON=0,0,0,*Value*
PO2	KON=1,0,1,*Value*
PO3	KON=2,1,1,*Value*
PO4	KON=2,0,2,*Value*
PO5	KON=1,1,2,*Value*

PO6	KON=2,2,2,*Value*
PO7	KON=0,1,1,*Value*
PO8	KON=0,2,2,*Value*
PO9	CoreKO
EHeat	EHeat
EISol	EISol
AlpB_NN,XFac_NN	DCore=*NN*,3,*Alpha*,*XFac*

† Note that MOPAC's GSP and GP2 are linear combinations of F0sp and G1sp. Gaussian uses the standard Slater-Condon names and parameter definitions.

Here is example MOPAC External data for Cr, printed by MOPAC using its debug option:

```
PARAMETER VALUES USED IN THE CALCULATION

 NI  TYPE        VALUE     UNIT
 24  USS     -34.86433900  EV         ONE-CENTER ENERGY FOR S
 24  UPP     -26.97861500  EV         ONE-CENTER ENERGY FOR P
 24  UDD     -54.43103600  EV         ONE-CENTER ENERGY FOR D
 24  ZS        3.28346000  AU         ORBITAL EXPONENT  FOR S
 24  ZP        1.02939400  AU         ORBITAL EXPONENT  FOR P
 24  ZD        1.62311900  AU         ORBITAL EXPONENT  FOR D
 24  BETAS    -5.12261500  EV         BETA PARAMETER    FOR S
 24  BETAP     3.92671100  EV         BETA PARAMETER    FOR P
 24  BETAD    -4.23055000  EV         BETA PARAMETER    FOR D
 24  GSS       8.85557242  EV         ONE-CENTER INTEGRAL (SS,SS)
 24  GPP       5.05309383  EV         ONE-CENTER INTEGRAL (PP,PP)
 24  GSP       5.58863066  EV         ONE-CENTER INTEGRAL (SS,PP)
 24  GP2       4.42952965  EV         ONE-CENTER INTEGRAL (PP*,PP*)
 24  HSP       0.64803936  EV         ONE-CENTER INTEGRAL (SP,SP)
 24  ZSN       1.61985300  AU         INTERNAL EXPONENT FOR S - (IJ,KL)
 24  ZPN       0.84826600  AU         INTERNAL EXPONENT FOR P - (IJ,KL)
 24  ZDN       1.40501500  AU         INTERNAL EXPONENT FOR D - (IJ,KL)
 24  F0DD      9.86923654  EV         SLATER-CONDON PARAMETER F0DD
 24  F2DD      5.20966257  EV         SLATER-CONDON PARAMETER F2DD
 24  F4DD      3.39760602  EV         SLATER-CONDON PARAMETER F4DD
 24  F0SD      6.15013600  EV         SLATER-CONDON PARAMETER F0SD
 24  G2SD      2.00030000  EV         SLATER-CONDON PARAMETER G2SD
 24  F0PD      5.63536196  EV         SLATER-CONDON PARAMETER F0PD
 24  F2PD      1.91648791  EV         SLATER-CONDON PARAMETER F2PD
 24  G1PD      1.58022558  EV         SLATER-CONDON PARAMETER G1PD
 24  G3PD      0.96233144  EV         SLATER-CONDON PARAMETER G3PD
 24  DD2       0.28669123  BOHR       CHARGE SEPARATION, SP, L=1
 24  DD3       2.91433601  BOHR       CHARGE SEPARATION, PP, L=2
 24  DD4       1.12394737  BOHR       CHARGE SEPARATION, SD, L=2
 24  DD5       0.81804068  BOHR       CHARGE SEPARATION, PD, L=1
 24  DD6       1.23219554  BOHR       CHARGE SEPARATION, DD, L=2
 24  PO1       1.53639890  BOHR       KLOPMAN-OHNO TERM, SS, L=0
 24  PO2       0.72875078  BOHR       KLOPMAN-OHNO TERM, SP, L=1
 24  PO3       1.96024483  BOHR       KLOPMAN-OHNO TERM, PP, L=2
 24  PO4       0.94950312  BOHR       KLOPMAN-OHNO TERM, SD, L=2
```

```
  24  PO5        1.66105265  BOHR      KLOPMAN-OHNO TERM, PD, L=1
  24  PO6        1.08400979  BOHR      KLOPMAN-OHNO TERM, DD, L=2
  24  PO7        1.53639890  BOHR      KLOPMAN-OHNO TERM, PP, L=0
  24  PO8        1.37859617  BOHR      KLOPMAN-OHNO TERM, DD, L=0
  24  PO9        1.53639890  BOHR      KLOPMAN-OHNO TERM, CORE
  24  CORE       6.00000000  E         CORE CHARGE
  24  EHEAT     95.00000000  KCAL/MOL  HEAT OF FORMATION OF THE ATOM (EXP)
  24  EISOL   -185.72482255  EV        TOTAL ENERGY OF THE ATOM (CALC)
  24 ALPB_24    4.65541900  ALPB factor
  24 XFAC_24   10.31860700  XFAC factor
```

5.74.3 Availability

Energies, optimizations and frequencies.

Parameters are stored in the program for the following elements:

◊ AM1: H, Li-F, Mg-Cl, Cr, Zn, Ge, Br, Sn, I and Hg.

◊ PM3: H, Li-F, Na-Cl, K, Ca, Cr, Zn-Br, Rb, Sr, Cd-I, Cs, Ba and Hg-Bi.

◊ PM6: H-Ba and Lu-Bi.

◊ PM7: H-La and Lu-Bi.

5.74.4 Related Keywords

CNDO, INDO, MNDO, MINDO3

5.74.5 Examples

The energy from these calculations appears in the output file as follows:

```
SCF Done: E(RAM1) = -0.943839275843E-01 A.U. after 6 cycles   AM1
SCF Done: E(RPM3) = -0.850201514485E-01 A.U. after 6 cycles   PM3
SCF Done: E(RPM6) = -0.864068239687E-01 A.U. after 7 cycles   PM6
SCF Done: E(RPM7) = -0.919784216493E-01 A.U. after 7 cycles   PM7
```

The energy printed is the heat of formation as computed by the model. Energies for other semi-empirical methods are reported similarly.

5.75 SP

This calculation type keyword requests a single-point energy calculation. It is the default when no calculation type keyword is specified.

5.75.1 Availability

All methods.

5.75.2 Examples

See the discussions of the various methods keywords for examples of their energy output formats.

5.76 Sparse

The Sparse keyword says to use sparse matrix storage for performance enhancement of large calculations (above around 400 atoms) [744]. The keyword's option allows you to specify the cutoff value for considering matrix elements to be zero.

See the *Program Development-Related Keywords* for options to this keyword.

5.76.1 Options

Loose

Use loose cutoffs: 5.0^{-5}

Medium

Use medium cutoffs: 5.0^{-7}

Tight

Use tight cutoffs: 1.0^{-10}

N

Set the cutoff to 10^{-N}

None

Overrides the default of Sparse for ONIOM layers.

5.77 Stable

This calculation type method requests that the stability of the Hartree-Fock or DFT wavefunction be tested. Gaussian has the ability to test the stability of a single-determinant wavefunction with respect to relaxing various constraints [745, 746](see also [680]). These include:

◊ Allowing an RHF determinant to become UHF.

◊ Allowing orbitals to become complex.

◊ Reducing the symmetry of the orbitals.

Note that analytic frequency calculations are only valid if the wavefunction has no internal instabilities. In examining the results prior to a frequency calculation, it suffices to see if any singlet instabilities exist for restricted wavefunctions or if any instabilities (singlet or triplet) exist for unrestricted wavefunctions. Møller-Plesset calculations are only valid if the wavefunction has no internal instabilities within the constrained symmetry. In examining the results prior to a Møller-Plesset calculation, an internal instability only affects the validity of the results if the pairs of orbitals mixed are of the same spatial symmetry. The validity of restricted Møller-Plesset energies based on wavefunctions which are unstable with respect to becoming UHF is also questionable [747].

By default, only real instabilities (i.e., not complex) are sought. The code which checks for a complex stability (Link 902) is older and less reliable and should not be used unless complex orbitals are of interest.

5.77.1 Options

RExt

Test for external real instability as well as internal instability (the default).

Int

Test for internal instability (a lower determinant with the same constraints) only.

RRHF

Constrain the wavefunction testing or reoptimization to be real, spin-restricted. Synonymous with Singlet.

RUHF

Constrain the wavefunction testing or reoptimization to be real, spin-unrestricted. Synonymous with Triplet.

CRHF

Allow testing for real to complex instabilities in spin-restricted wavefunctions.

CUHF

Allow testing for real to complex instabilities in spin-unrestricted wavefunctions.

Opt

If an instability is found, reoptimize the wavefunction with the appropriate reduction in constraints, repeating stability tests and reoptimizations until a stable wavefunction is found. RepOpt is a synonym for Opt. NoOpt prevents reoptimization and is the default. See also the QCOnly option below.

1Opt

Redo the SCF once if an instability is detected.

Direct

Forces a direct calculation. This is the default.

MO

Forces a stability calculation using transformed two-electron integrals (i.e., in the MO basis).

AO

Forces a calculation using the AO integrals (written to disk), avoiding an integral transformation. The AO basis is seldom an optimal choice, except for small molecules on systems having very limited disk. It is the default when SCF=Conven is also specified.

InCore

Forces an in-core algorithm.

ICDiag

Forces in-core full diagonalization of the matrix formed in memory from transformed integrals. It implies the use of MO integrals.

QCOnly

Suppresses the use of the regular SCF procedure (i.e., non quadratic convergence: Link 502) during later SCF calculations of the Stable=Opt iterations. This is the default.

XQC

Try to use the regular SCF procedure (Link 502) before quadratic convergence SCF (Link 508) for later SCF calculations of the Stable=Opt iterations.

Restart

Restarts the calculation off the checkpoint file. Also implies SCF=Restart.

5.77.2 Availability

HF and DFT methods.

5.77.3 Related Keywords

SCF

5.78 Symmetry

This keyword specifies the uses of molecular symmetry within the calculation. By default, the program attempts to identify the point group of the molecule. If symmetry is in use, the molecule may be rotated to a different coordinate system, called the *standard orientation*, before the calculation is performed. Derivatives are then rotated back to the original (*input*) orientation. Orbitals are printed in the standard orientation. Input for properties and background charge distributions must be specified in the standard orientation.

By default, symmetry is used wherever possible to reduce CPU, disk storage, and I/O requirements. The NoSymmetry keyword prevents molecule reorientation and causes all computations to be performed in the input orientation (although the program still attempts to identify the appropriate point group). Symmetry use can be completely disabled by Symmetry=None; use this option if NoSymm generates an error when identifying the point group.

5.78.1 Options

Int

Int enables and NoInt disables use of integral symmetry (use of the petite list). Synonymous with Int=[No]Symm.

Grad

NoGrad disables and Grad enables use of symmetry in integral derivative evaluation.

SCF

NoSCF disables and SCF enables use of N^3 symmetry in SCF, which is used by default only for GVB calculations. Symm=NoSCF is equivalent to Guess=LowSym and combining all irreducible representations together.

Loose

Tells the program to use looser cutoffs in determining symmetry at the first point. It is designed for use with suboptimal input geometries. Tight says to use the regular criteria at the first point, and it is the default.

Follow

Try to follow point group/orientation during optimization.

PG=*group*

Use no more symmetry than that found in the specified point group.

Axis=[X|Y|Z]

Specify axis to help specify subgroup.

CenterOfCharge

Use the center of atomic charges in determining the standard orientation. This is the default. COC is a synonym for this option.

CenterOfMass

Use the center of atomic masses in determining the standard orientation. COM is a synonym for this option.

None

Do not assign point group and bypass all symmetry processing.

PrintOrientation

Print the standard orientation to high precision.

SaveOrientation

Mark the standard orientation as the input orientation.

On

Turn on symmetry when it would otherwise be off. This can cause wrong answers, so it should only be used if you know what you're doing!

5.78.2 Related Keywords

Int, SCF

5.79 TD

This method keyword requests an excited state calculation using the time-dependent Hartree-Fock or DFT method [717, 748–753]; analytic gradients [717, 753] and frequencies [754–756] are available in Gaussian 16. For a review of using TD-DFT to predict excited state properties, see [757, 758].

Time-dependent DFT calculations can employ the Tamm-Dancoff approximation, via the TDA keyword. TD-DFTB calculations can also be performed [759].

Note that the normalization criteria used is $< X+Y|X-Y >= 1$.

Electronic circular dichroism (ECD) analysis is also performed during these calculations [760–765].

5.79.1 Options

General Options

Singlets

Solve only for singlet excited states. Only effective for closed-shell systems, for which it is the default.

Triplets

Solve only for triplet excited states. Only effective for closed-shell systems.

50-50

Solve for half triplet and half singlet states. Only effective for closed-shell systems.

Root=*N*

Specifies the "state of interest". The default is the first excited state (N=1).

NStates=*M*

Solve for M states (the default is 3). If 50-50 is requested, NStates gives the number of each type of state for which to solve (i.e., the default is 3 singlets *and* 3 triplets).

The keyword Read may also be specified as the parameter to the NStates option. In this case, the number of states to compute is read from the input stream. This features is typically used in EET calculations.

Add=*N*

Read converged states off the checkpoint file and solve for an additional N states. This option implies Read as well.

Read

Reads initial guesses for the states off the checkpoint file. Note that, unlike for SCF, an initial guess for one basis set cannot be used for a different one.

Restart

This option restarts a TD calculation after the last completed geometry. A failed frequency job may be restarted from its checkpoint file by simply repeating the route section of the original job, adding the Restart option to the keyword/option. No other input is required.

EqSolv

Whether to perform equilibrium or non-equilibrium PCM solvation. NonEqSolv is the default except for excited state optimizations and when the excited state density is requested (e.g., with Density=Current or All).

IVOGuess

Force use of IVO guess. This is the default for TD Hartree-Fock. NoIVOGuess forces the use of canonical single excitations for guess, and it is the default for TD-DFT. The HFIVOGuess option forces the use of Hartree-Fock IVOs for the guess, even for TD-DFT.

SOS

Do sum-over states polarizabilities, etc. By default, all excited states are solved for. A list of frequencies at which to do the sums is read in. Zero frequency is always done and need not be in the list.

NonAdiabaticCoupling

Requests that the ground-to-excited-state non-adiabatic coupling be computed [766, 767]. NAC is a synonym for this option. NoNonAdiabaticCoupling and NoNAC suppress this behavior. The default is NoNAC when computing energies or energies+gradients because the extra cost is non-trivial. The default is NAC during frequency calculations where the extra cost is negligible.

Conver=*N*

Sets the convergence calculations to 10^{-N} on the energy and $10^{-(N-2)}$ on the wavefunction. The default is N=4 for single points and N=6 for gradients.

Energy Range Options

An energy range can be specified for CIS and TD excitation energies using the following options to CIS, TD and TDA.

GOccSt=*N*

Generate initial guesses using only active occupied orbitals N and higher.

GOccEnd=*N*

Generate initial guesses: if N>0, use only the first N active occupied orbitals; if N<0, do not use the highest $|N|$ occupieds.

GDEMin=*N*

Generate guesses having estimated excitation energies $\geq N/1000$ eV.

DEMin=*N*

Converge only states having excitation energy $\geq N/1000$ eV; if N=-2, read threshold from input; if N<-2, set the threshold to $|N|/1000$ Hartrees. [768, 769]

IFact=*N*

Specify factor by which the number of states updated during initial iterations is increased. The default for IFact is $Max(4,g)$ where g is the order of the Abelian point group.

WhenReduce=*M*

Reduce to the desired number of states after iteration *M*. The default for WhenReduce is 1 for TD and 2 for TDA. Larger values may be needed if there are many states in the range of interest.

5.79.2 Availability

Energies, gradients and frequencies using Hartree-Fock or a DFT method.

Gradients and frequencies are not available for functionals for which third and fourth derivatives are not implemented: the exchange functionals G96, P86, PKZB, wPBEh and PBEh; the correlation functional PKZB; the hybrid functionals OHSE1PBE and OHSE2PBE.

5.79.3 Related Keywords

CIS, ZIndo, Output

5.79.4 Examples

Here is the key part of the output from a TD excited states calculation:

```
Excitation energies and oscillator strengths:

Excited State 1:  Singlet-A2  4.0147 eV  308.83 nm  f=0.0000  <S**2>=0.000
      8 ->  9         0.70701
This state for optimization and/or second-order correction.
Copying the excited state density for this state as the 1-particle RhoCI
   density.

Excited State 2:  Singlet-B1  9.1612 eV  135.34 nm  f=0.0017  <S**2>=0.000
      6 ->  9         0.70617

Excited State 3:  Singlet-B2  9.5662 eV  129.61 nm  f=0.1563  <S**2>=0.000
      8 -> 10         0.70616
```

The results on each state are summarized, including the spin and spatial symmetry, the excitation energy, the oscillator strength, the S^2, and (on the second line for each state) the largest coefficients in the CI expansion.

The ECD results appear slightly earlier in the output as follows:

```
1/2[<0|r|b>*<b|rxdel|0> + (<0|rxdel|b>*<b|r|0>)*]
Rotatory Strengths (R) in cgs (10**-40 erg-esu-cm/Gauss)
  state          XX          YY          ZZ     R(length)      R(au)
    1        0.0000      0.0000      0.0000      0.0000       0.0000
    2        0.0000      0.0000      0.0000      0.0000       0.0000
    3        0.0000      0.0000      0.0000      0.0000       0.0000
 1/2[<0|del|b>*<b|r|0> + (<0|r|b>*<b|del|0>)*] (Au)
  state          X           Y           Z         Dip. S.   Osc.(frdel)
    1         0.0000      0.0000      0.0000      0.0000       0.0000
    2        -0.0050      0.0000      0.0000      0.0050       0.0033
    3         0.0000     -0.2099      0.0000      0.2099       0.1399
```

5.80 Temperature

Specifies the temperature to be used for thermochemistry analysis (in Kelvin). The value should be specified as an option:

```
# ··· Temperature=300
```

The default is 298.15 K.

5.80.1 Options

Default

Restore the default temperature if a different value was retrieved by Geom=AllCheck.

5.81 Test

This keyword suppresses the automatic creation of an archive entry (formerly intended for the Browse Quantum Chemistry Database System). Its antonym is Archive, which is the default. Note that archive entries may be extracted from Gaussian log files after the fact using the pluck utility.

5.82 TestMO

The cutoffs used in computing and storing integrals and the convergence criteria applied in SCF and CPHF calculations are appropriate for most molecules and basis sets. However, if a nearly linearly dependent basis set is used, very large MO coefficients may occur and, in combination with the finite accuracy of other terms, lead to substantial numerical errors. By default, CPHF and post-SCF calculations are aborted if any MO coefficient is larger than 1000. (Note that this corresponds to a coefficient of 10^{12} for the contribution of an AO integral to an MO integral involving four virtual orbitals.) The NoTestMO keyword suppresses this check. It should be used only after careful thought. TestMO is the default.

5.83 TrackIO

This keyword requests routine-by-routine statistics of I/O and CPU usage.

5.83.1 Related Keywords

#P

5.84 Transformation

This keyword controls the algorithm used for integral transformation, as well as the types of transformed integrals produced.

5.84.1 Options

Integral Transformation Algorithm Options

The default is the smallest set of integrals the method can use.

Direct

Requests that the direct transformation routines be used. Equivalent to L804. Link 804 will select between the in-core, fully direct, and semi-direct methods automatically. This is the default.

InCore

Forces use of the in-core algorithm in Link 804.

Force

Forces an AO-to-MO integral transformation to be done during a single-point SCF calculation.

FullDirect

Forces use of the fully direct (MO integrals in core) method in Link 804.

SemiDirect

Forces use of the semi-direct algorithm in Link 804.

Integral Selection Options

Full

Forces a transformation over all orbitals (i.e., including transformed integrals involving all virtuals). ABCD is a synonym for Full.

IJAB

Produce only <IJ||AB> integrals.

IAJB

Produce <IJ||AB> and <IA||JB> integrals.

IJKL

Produce <IJ||AB>, <IA||JB>, and <IJ||KL> integrals.

IJKA

Produce <IJ||AB>, <IA||JB>, <IJ||KL>, and <IJ||KA> integrals.

IABC

Produce <IJ||AB>, <IA||JB>, <IJ||KL>, <IJ||KA>, and <IA||BC> integrals.

5.85 Units

The Units keyword controls the units used in the Z-matrix for distances and angles and related values, such as step-sizes in numerical differentiation. The defaults are Angstroms and degrees.

5.85.1 Options

Ang

Distances are in Angstroms (this is the default).

AU

Distances are in atomic units (Bohrs).

Deg

Angles are in degrees (the default).

Rad

Angles are in radians.

5.85.2 Restrictions

The Charge, Cube and Massage keywords are not affected by the setting of the Units keyword, and their input is always interpreted in units of Angstroms and degrees.

5.86 Volume

This keyword requests that the molecular volume be computed, defined as the volume inside a contour of 0.001 electrons/Bohr^3 density. The density to be used can be specified with the Density keyword. Since a Monte-Carlo integration is done, the computed volume is only accurate to about two significant figures, but this is sufficient to estimate a radius for use with the Onsager solvent reaction field model. The recommended radius (which is 0.5 Å larger than the radius corresponding to the computed volume) is printed in the output.

Since other, more accurate solvent models are available in Gaussian 16, this keyword has applicability only in preparation for frequency calculations using SCRF=Dipole.

5.86.1 Options

Tight

Requests an increased density of points for more accurate integration. By default, the volume is computed to an accuracy of about 10%. Use of this option is recommended if the computed molecular volume is needed more quantitatively.

5.86.2 Availability

Hartree-Fock, all DFT methods, CIS, MP2, MP3, MP4(SDQ), CID, CISD, CCD, CCSD, and QCISD.

5.86.3 Related Keywords

SCRF=Dipole

5.87 W1 Methods

These method keywords request the W1 methods of Martin [107, 770]:

◊ The W1U keyword specifies the W1 method modified to use UCCSD instead of ROCCSD for open shell systems [771] (the ROCCSD method is that of Handy, Pople and coworkers [543]).

◊ W1BD requests a related method which replaces coupled cluster with BD [771]. This method is both more expensive and more accurate than CBS-QB3 and G3 (true for all W1 methods); we recommend using W1BD when you require very accurate energies.

◊ W1RO is the W1 method described in [107] with a slightly improved scalar relativistic correction as described in [771].

5.87.1 Options

SP

Perform only a single-point energy evaluation using the specified compound model chemistry. No zero-point or thermal energies are included.

NoOpt

Perform the frequencies and single-point energy calculation for the specified model chemistry at the input geometry. Freq=TProjected is implied. This option is not meaningful or accepted for methods such as G1, which use different geometries for the frequencies and the single-point steps. StartFreq is a synonym for NoOpt.

ReadAmplitudes

Reads the converged amplitudes from the checkpoint file (if present). Note that the new calculation can use a different basis set, method (if applicable), etc. than the original one.

SaveAmplitudes

Saves the converged amplitudes in the checkpoint file for use in a subsequent calculation (e.g., using a larger basis set). Using this option results in a very large checkpoint file, but also may significantly speed up later calculations.

The ReadAmplitudes option is the default for all W1 methods. SaveAmplitudes is also the default for W1BD.

Restart

Restart an incomplete W1 calculation.

ReadIsotopes

This option allows you to specify alternatives to the default temperature, pressure, frequency scale factor and/or isotopes – 298.15 K, 1 atmosphere, no scaling, and the most abundant isotopes (respectively). It is useful when you want to rerun an analysis using different parameters from the data in a checkpoint file. Be aware, however, that all of these can be specified in the route section (Temperature, Pressure and Scale keywords) and molecule specification (the Iso parameter), as in this example:

```
#T Method/6-31G(d) JobType Temperature=300.0 ···

...

0 1
C(Iso=13)
...
```

ReadIsotopes input has the following format:

temp pressure [scale]	*Values must be real numbers.*
isotope mass for atom 1	
isotope mass for atom 2	
...	
isotope mass for atom n	

Where *temp*, *pressure*, and *scale* are the desired temperature, pressure, and an optional scale factor for frequency data when used for thermochemical analysis (the default is unscaled). The remaining lines hold the isotope masses for the various atoms in the molecule, arranged in the same order as they appeared in the molecule specification section. If integers are used to specify the atomic masses, the program will automatically use the corresponding actual exact isotopic mass (e.g., 18 specifies ^{18}O, and Gaussian uses the value 17.99916).

5.87.2 Examples

Calculation Summary Output. After all of the output for the component job steps, Gaussian prints a table of results for these methods. Here is the key part of the output from a W1BD calculation:

Results before spin correction.

```
Temperature=          298.150000 Pressure=                1.000000
E(ZPE)=                 0.016919 E(Thermal)=              0.019783
W1BD  (0 K)=          -39.139927 W1BD  Energy=          -39.137063
W1BD  Enthalpy=       -39.136119 W1BD  Free Energy=     -39.158277

W1U spin correction:
G.P.F. Wood, L. Radom, G.A. Petersson, E.C. Barnes, M.J. Frisch
and J.A. Montgomery, Jr., JCP 125, 94106 (2006).

DE(Spin)=                         -0.000051

W1Bsc Electronic Energy           -39.156897
```

Predicted energy.

Spin-corrected results.

```
Temperature=          298.150000 Pressure=                1.000000
E(ZPE)=                 0.016919 E(Thermal)=              0.019783
W1Bsc(0 K)=           -39.139978 W1Bsc Energy=          -39.137114
W1Bsc Enthalpy=       -39.136170 W1Bsc Free Energy=     -39.158328
```

The predicted energy is given between the ordinary and spin-corrected thermochemistry analysis tables. The energy labels thus have the following meanings (spin-corrected W1BD is used as an example):

W1Bsc (0 K)	Zero-point-corrected electronic energy: $E_0 = E_{elec} + ZPE$
W1Bsc Energy	Thermal-corrected energy: $E = E_0 + E_{trans} + E_{rot} + E_{vib}$
W1Bsc Enthalpy	Enthalpy computed using the spin-corrected W1BD predicted energy: H = E + RT
W1Bsc Free Energy	Gibbs Free Energy computed using the spin-corrected W1BD predicted energy: G = H - TS

5.88 Window Keyword and Frozen Core Options

These options specify which inner orbitals are frozen in post-SCF calculations. Gaussian 16 added some additional options to the ones previously available in the program [772]. See also BD for BD-specific choices.

The Window keyword performs an analogous function but takes its input as parameters in the route section rather than from the input stream. It requests that L801 perform the setup for a frozen-core calculation during jobs (e.g., single-point SCF) which would otherwise not set up a window.

5.88.1 Options

Frozen Core Options to Post SCF Methods Keywords

FC

This specifies a “frozen core” calculation, and it implies that inner-shells are excluded from the correlation calculation. This is the default calculation mode. Note that FC, Full, RW, and Window are mutually exclusive. It is equivalent to FreezeG2 for the 6-31G and 6-311G basis sets and to FreezeNobleGasCore for all other basis sets, except that the outer s and p core orbitals of 3rd row and later alkali and alkaline earth atoms are not frozen (in accord with the G2/G3/G4 conventions).

FreezeNobleGasCore

In post-SCF calculations, the largest noble gas core is frozen. FrzNGC is a synonym for this option.

FreezeInnerNobleGasCore

In post-SCF calculations, the next to largest noble gas core is frozen. That is, the outermost core orbitals are retained. FrzINGC and FC1 are synonyms for this option.

FreezeG2

Freeze orbitals according to the G2 convention: d orbitals of main group elements are frozen, but the outer sp core of 3rd row and later alkali and alkaline earth elements are kept in the valence.

FreezeG3

Freeze orbitals according to the G3 convention.

FreezeG4

Freeze orbitals according to the G4 convention.

Full

This specifies that all electrons be included in a correlation calculation.

RW

The "read window" option means that specific information about which orbitals are retained in the post-SCF calculation will be given in the input file. ReadWindow is a synonym for RW.

The required input section consists of a line specifying the starting and ending orbitals to be retained, followed by a blank line. A value of zero indicates the first or last orbital, depending on where it is used. If the value for the first orbital is negative (-*m*), then the highest *m* orbitals are retained; if the value for the last orbital is negative (-*n*), then the highest *n* orbitals are frozen. If *m* is positive and *n* is omitted, *n* defaults to 0. If *m* is negative and *n* is omitted, then the highest |*m*| occupied and lowest |*m*| virtual orbitals are retained.

Here are some examples for a calculation on C_4H_4:

0,0	Equivalent to Full.
5,0	Freezes the 4 core orbitals and keeps all virtual orbitals (equivalent to FC if the basis has a single zeta core).
5,-4	Freezes the four core orbitals and the highest four virtual orbitals. This is the appropriate frozen core for a basis with a double-zeta core.
6,22	Retains orbitals 6 through 22 in the post-SCF phase. For example, since C_4H_4 has 28 electrons, if this is a closed-shell calculation, there will be 14 occupied orbitals, 5 of which will be frozen, so the post-SCF calculation will involve 9 occupied orbitals (orbitals 6-14) and 8 virtual orbitals (orbitals 15-22).
-6	Retains orbitals 9 through 20.

ChkWindow

The window read in during a previous job is recovered from the checkpoint file.

ListWindow

Reads a list of orbitals to freeze from the input stream, terminated by a blank line. Two lists are read for unrestricted calculations. A range of orbitals can be specified, e.g.: 2 7-10 14.

The Window Keyword

Window=(*m*[,*n*])

Window=*parameter*

This keyword performs the same function as the ReadWindow option, but takes its input as parameters in the route section rather than from the input stream. It is typically used with Output=(Matrix, MO2ElectronIntegrals).

The Window keyword accepts one or two integers as parameters (as with the corresponding option). It also accepts the following parameters, which have the same meanings as the correspondingly-named options: FC, FreezeNobleGasCore, FreezeInnerNobleGasCore, FreezeG2, FreezeG3, FreezeG4, and Full.

5.89 ZIndo

This method keyword requests an excited state energy calculation using the ZIndo/S method [773–781]. Note that ZIndo calculations must not specify a basis set keyword.

5.89.1 Options

Singlets

Solve only for singlet excited states. Only effective for closed-shell systems, for which it is the default.

Triplets

Solve only for triplet excited states. Only effective for closed-shell systems.

50-50

Solve for half triplet and half singlet states. Only effective for closed-shell systems.

Root=*N*

Specifies the “state of interest”. The default is the first excited state (N=1).

NStates=*M*

Solve for M states (the default is 3). If 50-50 is requested, NStates gives the number of each type of state for which to solve (i.e., the default is 3 singlets *and* 3 triplets).

Add=*N*

Read converged states off the checkpoint file and solve for an additional N states.

Window=(*m*[,*n*])

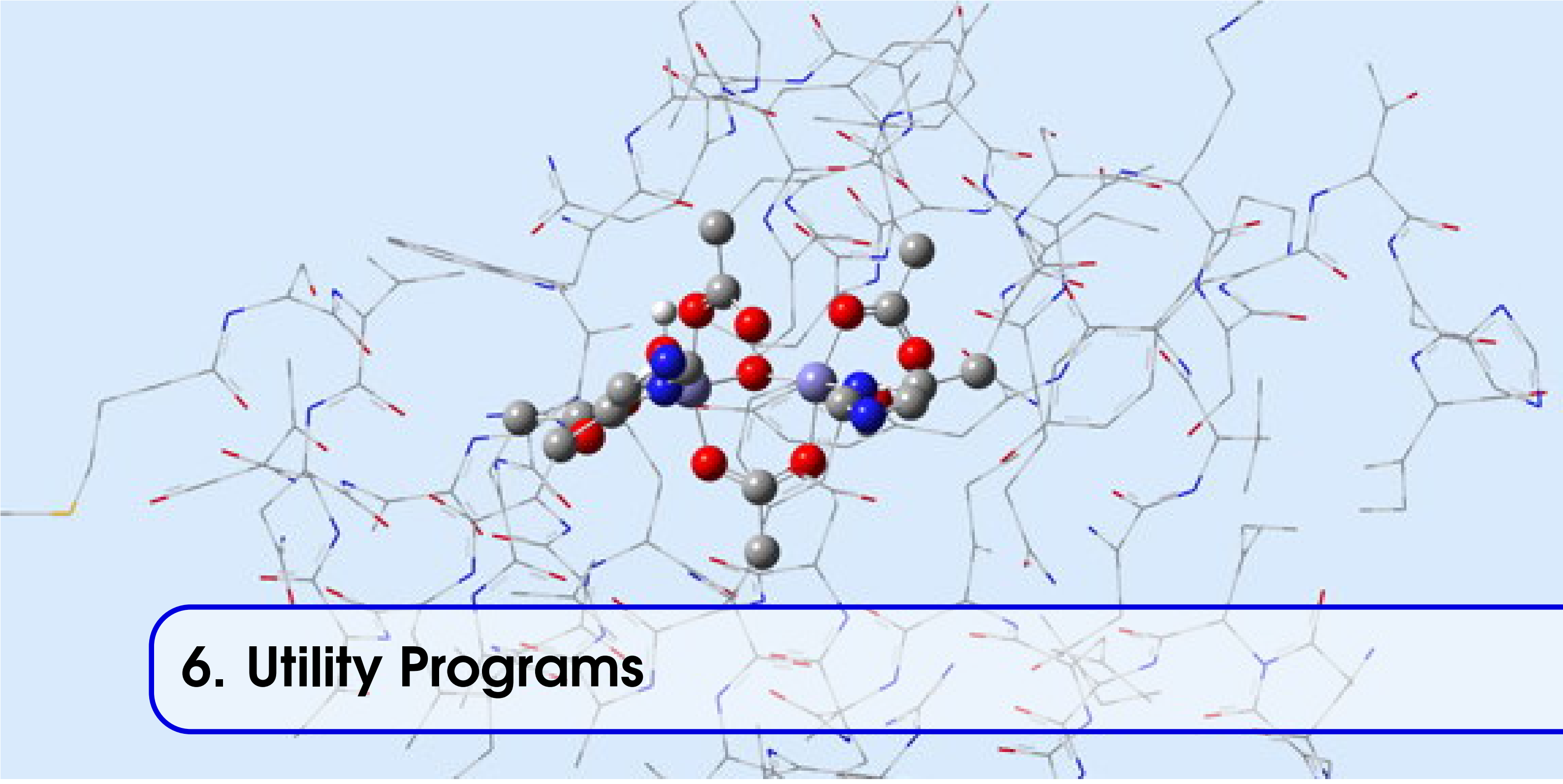
The two values specify the starting and ending orbitals to be used. The default is to use all orbitals. A value of zero indicates the first or last orbital, depending on where it is used. If the value for the first orbital is negative (-m), then the highest m orbitals are retained; the value for the last orbital is negative (-n), then the highest n orbitals are frozen. If m is positive and n is omitted, n defaults to 0. If m is negative and n is omitted, then the highest $|m|$ occupied and lowest $|m|$ virtual orbitals are retained.

5.89.2 Availability

Energies only. The Density keyword is ignored for ZIndo calculations.

5.89.3 Related Keywords

CIS, TD



6. Utility Programs

Most utilities are available for both UNIX and Windows versions of Gaussian. However, be sure to consult the release notes accompanying the program for information pertaining to specific operating systems.

The GAUSS_MEMDEF environment variable may be used to increase the memory available to utilities which do not offer such an option themselves. Its value should be set to the desired amount of memory. The default unit is words; the specified value can also be followed by a suffix indicating other units (e.g., GB for gigabytes).

The following lists the available utilities and their functions (*starred* items are included on the Gaussian 16W **Utilities** menu):

◊ c8616

Converts checkpoint files from previous program versions to Gaussian 16 format.

◊ chkchk*

Displays the route and title sections from a checkpoint file.

◊ cubegen*

Standalone cube generation utility.

◊ cubman*

Manipulates Gaussian-produced cubes of electron density and electrostatic potential (allowing them to be added, subtracted, and so on).

◊ formchk*

Converts a binary checkpoint file into an ASCII form suitable for use with visualization programs and for moving checkpoint files between different types of computer systems.

◊ freqchk*

Prints frequency and thermochemistry data from a checkpoint file. Alternate isotopes, temperature, pressure and scale factor can be specified for the thermochemistry analysis.

◊ freqmem

Determines memory requirements for frequency calculations.

◊ gauopt

Performs optimizations of variables other than molecular coordinates.

◊ ghelp

On-line help for Gaussian.

◊ mm

Standalone molecular mechanics program.

◊ newzmat*

Conversion between a variety of molecular geometry specification formats.

◊ testrt*

Route section syntax checker and non-standard route generation.

◊ unfchk*

Convert a formatted checkpoint file back to its binary form (e.g., after moving it from a different type of computer system).

6.1 c8616

The c8616 utility converts checkpoint files from Gaussian 86 through Gaussian 09 to Gaussian 16 format. It takes the name of the checkpoint file as its argument and transforms it in place so that the reformatted file has the same name as the original one. For example, the following command converts the checkpoint file *taxol.chk* in the Gaussian scratch directory to Gaussian 16 format:

$ c8616 $GAUSS_SCRDIR/taxol.chk

6.2 chkchk

The chkchk utility displays the route and title sections from a checkpoint file, and indicates other information that is present within it. It is useful for determining the contents of random checkpoint files whose purpose has been forgotten and whose names are non-descriptive. It takes the name of the checkpoint file as its argument. Here is an example of its use:

$ chkchk important

```
Checkpoint file important.chk:
Title:  Opt freq for pentaprismane
Route:  #P opt freq b3lyp/6-311+g(2d,p)
Version:  ES64L-G16RevB.01
NAtoms=      4 ICharg=     0 Multip=     1
Atomic coordinates present.
AO basis is present.
MO coefficients present.
Cartesian force constants present.
This checkpoint is from a job which completed successfully.
Internal force constants may be present.
An OT bucket is present. The stored data is from a(n) Optimization job.
  Total number of points stored =         1.
  Number of statistics per point=         2.
```

The -p option may be used to print additional information from the checkpoint file, including the molecular coordinates, the basis set in Gen format, and the molecular orbitals in the correct format for Guess=Cards. This information may be redirected to a file for later use as input to Gaussian or another program.

Here is an example:

```
$ chkchk -p myfile >newjob.gjf
$ cat newjob.gjf
 Checkpoint file myfile.chk:
 Title:  DMABN: Opt freq from saddle point found by first opt: acetonitrile
 Route:  #N Geom=AllCheck Guess=TCheck SCRF=Check GenChk
         B3LYP/6-311+G(2d,p) Freq
 Atomic coordinates (Angstroms):   Molecular structure
    0    1
  N   2.258424299166     -0.051967437933     -0.108610457679
  C   0.911965883667     -0.011528021623     -0.025315883831
  ...

 AO basis set (Overlap normalization):   Basis set in format for Gen keyword
 Atom N1 Shell 1 S  6  bf 1 -  1   4.267803417278   -0.098204225521   -0.205244020181

      0.6293480000D+04  0.1969788147D-02
      0.9490440000D+03  0.1496128592D-01
      ...
 Atom N1 Shell 2 SP 3  bf 2 -  5   4.267803417278   -0.098204225521   -0.205244020181

      0.3063310000D+02  0.1119060795D+00  0.3831191864D-01
      ...

(5D15.8)     MOs in format for Guess=Cards
    1 Alpha MO OE=-0.15587059D+02
 0.55760173D+00 0.46815568D+00-0.15193125D-03 0.20920429D-03 0.73161643D-03
 0.58391570D-02-0.61620240D-04-0.80819280D-04-0.30786697D-03-0.40418785D-02
 ...

  357 Alpha MO OE= 0.37351112D+02
 0.81138955D-01-0.87165085D-01 0.52591067D-03 0.25976528D-03-0.27428643D-03
 0.35549516D-01-0.44238173D-02-0.13061602D-02 0.11962956D-02-0.14464014D+00

 Input for Opt=FCCards (Energy,Forces,Force Constants): Forces for use with Opt=FCCards
 -0.4554698501440964D+03
  0.00000485  0.00000348  0.00000040 -0.00000441  0.00000421 -0.00000731
  0.00000048  0.00000292 -0.00000191 -0.00000112  0.00000014 -0.00000354
  ...

 Internal force constants may be present.
 An OT bucket is present. The stored data is from a(n) Optimization job.
   Total number of points stored =         1.
   Number of statistics per point=         2.
```

Note that some editing of the generated input file may be required prior to use.

6.3 cubegen

Gaussian includes a standalone utility for generating cubes from the data in a formatted checkpoint file (equivalent to the previous Cube keyword). The utility is named cubegen, and it has the following syntax:

cubegen *nprocs kind fchkfile cubefile npts format cubefile2*

The parameters, which are not case-sensitive, have the following meanings:

nprocs

Number of shared memory processors used for electrostatic potential calculations. A value of 0 is equivalent to 1 (it is the default). Note that this parameter must be included if other parameters are specified. Previously this parameter was used to specify the amount of memory to allocate. The GAUSS_MEMDEF environment variable should be used instead.

kind

A keyword specifying the type of cube to generate:

MO=*n*	Molecular orbital *n*. The keywords Homo, Lumo, All, OccA (all alpha occupied), OccB (all beta occupied), Valence (all valence orbitals) and Virtuals (all virtual orbitals) may also be used in place of a specific orbital number. There is no default for *n*, and an error will occur if it is omitted. AMO and BMO can be similarly used to select only alpha or beta orbitals (respectively). For open shell systems, Homo selects both alpha and beta orbitals.
Density=*type*	Total density of the specified type. The *type* keyword is one of the single density selection options that are valid with the Density keyword: SCF, MP2, CC, CI, and so on (note that Current is not supported). The fdensity, falpha and fbeta forms request the use of full instead of frozen core densities. The default is SCF.
Spin=*type*	Spin density (difference between α and β densities) of the specified type.
Alpha=*type*	Alpha spin density of the specified type.
Beta=*type*	Beta spin density of the specified type.
Potential=*type*	Electrostatic potential using the density of the specified type.
Gradient	Compute the density and gradient.
Laplacian	Compute the Laplacian of the density ($\nabla^2\rho$).
NormGradient	Compute the norm of the density gradient at each point.
CurrentDensity=*I*	Magnitude of the magnetically-induced (GIAO) current density, where *I* is the applied magnetic field direction (X, Y or Z).
ShieldingDensity=*IJN*	Magnetic shielding density. *I* is the direction of the applied magnetic field, *J* is the direction of the induced field (X, Y or Z), and *N* is the number of the nucleus for which the shielding density (GIAO) is to be calculated.

fchkfile

Name of the formatted checkpoint file. cubegen will prompt for this filename if it is not specified.

cubefile

Name of the output cube file; test.cube is the default if it is not explicitly specified (i.e., specifying the name of the checkpoint file does not change the default cube filename).

npts

Number of points per side in the cube. A value of 0 selects the default value of 80^3 points distributed evenly over a rectangular grid generated automatically by the program (not necessarily a cube). Positive values of *npts* similarly specify the number of points per “side”; e.g., 100 specified a grid of 1,000,000 (100^3) points.

The values -2, -3 and -4 correspond to the keywords Coarse, Medium and Fine and to values of 3 points/Bohr, 6 points/Bohr and 12 points/Bohr (respectively). Negative values of *npts* $\leq$-5 specify spacing of *npts*$*10^{-3}$ Angstroms between points in the grid.

A value of -1 says to read the cube specification from the input stream or from a second cube file (see below), according to the following format:

IFlag, X_0, Y_0, Z_0	*Output unit number and initial point.*
N_1, X_1, Y_1, Z_1	*Number of points and step-size in the X-direction.*
N_2, X_2, Y_2, Z_2	*Number of points and step-size in the Y-direction.*
N_3, X_3, Y_3, Z_3	*Number of points and step-size in the Z-direction.*

IFlag is the output unit number. If *IFlag* is less than 0, then a formatted file will be produced; otherwise, an unformatted file will be written.

If N_1<0 the input cube coordinates are assumed to be in Bohr, otherwise, they are interpreted as Angstroms. $|N_1|$ is used as the number of X-direction points in any case; N_2 and N_3 specify the number of points in the Y and Z directions, respectively. Note that the three axes are used exactly as specified; they are not orthogonalized, so the grid need not be rectangular.

The value -5 says to read in an arbitrary list of points from standard input. If you enter this input by hand, terminate the input with an end-of-file (i.e., Ctrl-D under Unix). Alternatively, you can redirect standard input to a file containing the list of points (do not place a blank line or Ctrl-D at the end of the file).

format

Format of formatted output files: h means include header (this is the default); n means don’t include header. This parameter is ignored when unformatted cube files are produced.

cubefile2

If specified, the size for the generated cube is taken from this file. This option is useful when creating cubes for later arithmetic operations such as difference densities.

In order for the cube dimensions to be taken from the specified cube file, the npts parameters must be -1, and the specified file must have been created with a header.

If no parameters are specified, cubegen will prompt for fchkfile and run the following:

cubegen 1 density=scf test.cube 80 h

This generates a file named test.cube (with header) containing the SCF density in a rectangular grid of 803 points.

6.3.1 Output File Formats

All values in the cube file are in atomic units, regardless of the input units.

Densities, the norm of the density gradient, the Laplacian of the density, and the potential are scalars (i.e. one value per point, *NVal*=1). A gradient cube contains the density plus the vector gradient of the density, so it has four values per point (*NVal*=4): i.e. the value of the density plus the X, Y, and Z components of its gradient.

Cube files have one row per record (i.e., N_1*N_2 records each of length N_3*$NVal$). For formatted output, each row is written out in format (6E13.5). In this case, if N_3*$NVal$ is not a multiple of six, then there may be blank space in some lines.

For example, schematically, cube files will look like this:

NAtoms, X-Origin, Y-Origin, Z-Origin, NVal	*NVal is the # of values/point*
N1, X1, Y1, Z1	*# of increments in the slowest running direction*
N2, X2, Y2, Z2	
N3, X3, Y3, Z3	*# of increments in the fastest running direction*
IA1, Chg1, X1, Y1, Z1	*Atomic number, charge, and coordinates of the first atom*
...	
IAn, Chgn, Xn, Yn, Zn	*Atomic number, charge, and coordinates of the last atom*
*(N1*N2) records, each length N3*NVal*	*Values of the density at each point in the grid*

Note that a separate write is used for each record.

For molecular orbital output, *NAtoms* will be less than zero, and an additional record follows the data for the final atom (in format 10I5 if the file is formatted):

NMO, (MO(I),I=1,NMO)   *Number of MOs and their numbers*

If N_{MO} orbitals were evaluated, then each record is N_{MO}*N_3 long and has the values for all orbitals at each point together (in this case N_{MO}=$NVal$).

6.3.2 Reading Cube Files with Fortran Programs

If one wishes to read the values of a cube file back into an array dimensioned X($NVal$,N_3,N_2,N_1), code like the following Fortran loop may be used:

```
Do 10 I1 = 1, N1
Do 10 I2 = 1, N2
   Read(n,'(6E13.5)') ((X(IVal,I3,I2,I1),IVal=1,NVal)I3=1,N3)
10 Continue
```

where *n* is the unit number corresponding to the cube file.

6.4 cubman

The cubman program manipulates cubes of values of electron density and electrostatic potential as produced by Gaussian. The program prompts for an operation to perform and then the names of the necessary files. The possible operations and their associated subcommands are:

add

Add two cubes to produce a new one.

copy

Copy a cube, possibly converting it from formatted to unformatted or vice versa.

diff

Compute properties of the difference between two cubes, without writing out a new cube.

prop

Computes the properties of a single cube.

subtract

Subtracts two cubes to produce a new cube.

scale

Scale a cube by a constant factor, producing a new cube.

square

Multiply a cube by itself to produce a new cube.

sprod

Take the scalar product of vectors in two cubes, producing a new cube of scalars.

vprod

Take the vector product of 3-vectors in two cubes, producing a new cube of 3-vectors.

magnitude

Compute the magnitude of vectors in a cube, producing a new cube of scalars.

normalize

Normalize the vectors in a cube, producing a new cube of vectors.

select

Select among the values per point and produce a new cube.

toxyz

Convert a cube into a simple list: $X,Y,Z,Value,Value,\cdots$ format.

All operation subcommands can be abbreviated to the shortest unique form.

6.5 Example

Here are some annotated sample runs with cubman (user input is shown in boldface type, and output has been condensed slightly due to space considerations):

```
$ cubman
Action [Add, Copy, Difference, Properties, SUbtract, SCale, SQuare]? p
Input file? b.cube
Is it formatted [no,yes,old]? y
Opened special file b.cube.
Input file titles:
First excited state of propellane                  Title line from the job
CI Total Density                                   Contents of cube file

 SumAP= 13.39263 SumAN= .00000 SumA= 13.39263      Statistics about cube contents
 CAMax=  3.35320 XYZ= .18898 -1.32280 .000004
 CAMin=   .00000 XYZ= -9999.00000 -9999.00000 -9999.00000

 DipAE=  -.8245357658     .7624198057     .1127178115
 DipAN=  -.0000060000    -.0000060000     .0000000000
 DipA=   -.8245417658     .7624138057     .1127178115

$ cubman
Action [Add, Copy, Difference, Properties, SUbtract, SCale, SQuare]? su
First input? b.cube
```

```
Is it formatted [no,yes,old]?  y
Opened special file b.cube.
Second input? a.cube
Is it formatted [no,yes,old]?  y
Opened special file a.cube.
Output file? c.cube                                    File to hold the new cube
Should it be formatted [no,yes,old]?  y
Opened special file c.cube.
Input file titles:
First excited state of propellane                     Title from first file
CI Total Density                                      Contents of first cube
Input file titles:
Propellane HF/6-31G*                                  Title from second file
SCF Total Density                                     Contents of second cube
Output file titles:                                   Composite title used for new file
First excited state of propellane || Propellane HF/6-31G*
CI Total Density - SCF Total Density                  Difference to be computed

 SumAP= 13.39263 SumAN= .00000 SumA= 13.39263         Statistics for first cube
 CAMax=  3.35320 XYZ=  .18898 -1.32280 .000004
 CAMin=   .00000 XYZ= -9999.00000 -9999.00000 -9999.00000

 SumBP= 13.38168 SumBN= .00000 SumB= 13.38168         Statistics for second cube
 CBMax=  3.39683 CBMin= .00000

 SumOP=   .63453 SumON=-.62358 SumO=   .01094         Statistics for output cube
 COMax=   .49089 COMin=-.39885

 DipAE=  -.8245357658     .7624198057     .1127178115
 DipAN=  -.0000060000    -.0000060000     .0000000000
 DipA=   -.8245417658     .7624138057     .1127178115

 DipBE=  -.8306292172     .5490287046     .1243830393
 DipBN=  -.0000060000    -.0000060000     .0000000000
 DipB=   -.8306352172     .5490227046     .1243830393

 DipOE=   .0060934514     .2133911011    -.0116652278
 DipON=  -.0000060000    -.0000060000     .0000000000
 DipO=    .0060874514     .2133851011    -.0116652278
```

In the output, the input cubes are denoted as A and B, and the output cube is designated by O. Other code letters are N for "negative values" or for "nuclear", depending on the context, P for "positive values", E for "electronic", C for "charge", Dip for "dipole", Sum for "summarization", Max for "maximum", and Min for "minimum." Thus, SumAN is the sum over the first input cube, taking the negative values only, and DipON is the nuclear contribution to the dipole moment for the output cube. Similarly, CBMax is the maximum charge for the second input cube, and SumO is the sum of the values in the output cube, including both positive and negative values.

6.6 formchk

formchk converts the data in a Gaussian checkpoint file into formatted forms that are suitable for input into a variety of visualization software. By default, formchk creates a text version of the checkpoint file known as the “formatted checkpoint file.” formchk can also generate “matrix element files” via various options. (see *Options*)

formchk has the following syntax:

formchk [*options*] *chkpt-file* [*formatted-file*]

where *chkpt-file* is the name of the binary checkpoint file to be formatted and *formatted-file* is the name for the resultant output file. If the name of the formatted file is omitted, it defaults to the base name of the checkpoint file with .fchk.

For example, the following command will produce the formatted checkpoint file propell.fchk from the checkpoint file propell.chk:

$ formchk propell.chk propell.fchk

The conventional extension for formatted checkpoint files is .fchk on Unix systems and other computers supporting variable-length extensions.

Note that formatted checkpoint files can be used as a data exchange format between computer platforms. Use formchk on the originating computer and unfchk on the target computer to create a binary checkpoint file. Always include the -3 option when moving files between different types of computers (see *Options*).

6.6.1 Options

By default formchk produces a formatted checkpoint file that is backwardly compatible with all previous versions of Gaussian.

-3

Produce a version 3 formatted checkpoint file, including all features supported in Gaussian 16. This is also the format used by Gaussian 09.

-2

Produce a version 2 formatted checkpoint file. This was the version used with Gaussian 03.

-c

Causes the molecular mechanics atom types to appear in the formatted checkpoint file as strings rather than integers.

-r

Restore data for the general case in an ONIOM checkpoint file (i.e., the low-level on the real system). This is useful if the checkpoint file is from an ONIOM job which died in the middle of an ONIOM calculation but one wants to put the real system structure and other information in the formatted checkpoint file, e.g., to read the structure into GaussView in order to modify it.

In addition to creating formatted checkpoint files, formchk can also generate matrix element files via the following options:

-matrix

Generates a Fortran unformatted matrix element file from the specified checkpoint file. The default extension for the output file is .dat.

-rawmatrix

Writes a raw binary matrix element file from the specified checkpoint file. The default extension for the output file is .dat.

-i4

Writes the Fortran unformatted matrix element file using 4-byte integers in the output file even if Gaussian is using 8-byte integers (i.e., on a 64-bit machine). formchk -matrix -i4 is needed if the file is to be read using the example Fortran and Python code in the gauopen directory because they are compiled with 4-byte integers. If the file is to be read by some other program, whether -i4 is needed depends on the size of integers in that program.

-files="(*list*)"

When combined with the -matrix or -rawmatrix option, includes the contents of the specified internal Gaussian file(s) within the generated matrix element file. For example, the following option:

$ formchk -files="(123,(456,offset=1,integer=27))" -matrix fox7_dimer.chk fox7_dimer.dat

will cause the contents of internal file 123 (assumed by default to be real values), the 27 integers in internal file 456 starting after the first (8-byte) word, and any real values following the integers, to be included within the matrix element file.

6.7 freqchk

The freqchk utility is used to retrieve frequency and thermochemistry data from a checkpoint file, with optional specification of an alternate temperature, pressure, scale factor, and/or isotope substitutions.

The full syntax of the freqchk command is:

$ freqchk *checkpoint-file* [*options*] [*answers to prompts*]

If the checkpoint file name does not include an extension, .fchk is assumed; both formatted and unformatted checkpoint files are accepted.

6.7.1 Options

The following options are supported:

-o *filename*

Send output to the specified file. By default, output is displayed to the screen.

-nfd

Skip the diagonalization of the full $(3N_{atoms})^2$ matrix (as with Freq=NoDiagFull).

-sel

Select normal modes for inclusion. The utility will prompt for required information as usual and then pause to allow you to enter the desired modes (without prompting). You can specify mode numbers and/or atom lists as for Freq=ReadNormalModes. See the example below.

-read

Read normal modes from the checkpoint file rather than computing them.

-save

Save normal modes; only works with binary checkpoint files. When combined with -sel, only the selected modes will be in the checkpoint for a subsequent -read operation.

-np=*N*

Use N processors to compute the frequency analysis.

6.7.2 Examples

freqchk can prompt for all other information that it requires. The following annotated sessions illustrate its use in this mode (user input is set in boldface type):

```
$ freqchk
 Checkpoint file? solvent.chk
 Write Hyperchem files? n
 Temperature (K)? [0=>298.15] 0                    Zero must be entered; return doesn't work
 Pressure (Atm)? [0=>1 atm] 0
 Scale factor for frequencies during thermochemistry? [0=>1/1.12] 0
 Do you want the principal isotope masses? [Y]:          Return accepts defaults
 Isotopes for each atom are printed
 Full mass-weighted force constant matrix:
 Low frequencies -- -948.3077     .0008     .0020     .0026
 ...
 Normal Gaussian frequency output follows ...
                       1                    2
                      ?A                   ?A
 Frequencies --   1885.3939            3853.5773
 Red. masses --      1.0920               1.0366
 Frc consts  --      2.2871               9.0697
 IR Inten    --     17.3416              21.5997
 Raman Activ --      7.8442              67.0384
 Depolar     --       .7428                .2248
 Atom AN      X      Y      Z        X      Y      Z      Normal modes
   1   8    .06    .00    .04      .04    .00    .02
   2   1   -.70    .00    .03      .01    .00   -.71
   ...
 -------------------
 - Thermochemistry -
 -------------------
 Temperature   298.150 Kelvin.  Pressure   1.00000 Atm.
 Thermochemistry will use frequencies scaled by  .8929.
 ...
 Zero-point vibrational energy      53494.5 (Joules/Mol)
                                   12.78550 (Kcal/Mol)
 VIBRATIONAL TEMPERATURES:  2422.01  4950.36  5495.38 (KELVIN)
Zero-point and thermal corrections:
 Zero-point correction=                .020375 (Hartree/Particle)
 Thermal corr to Energy=               .023210
 Thermal corr to Enthalpy=             .024154
 Thermal corr to Gibbs Free Energy=    .045589
 E=thermal energy; CV=constant volume molar heat capacity; S=entropy
                 E             CV                S
             KCAL/MOL   CAL/MOL-KELVIN   CAL/MOL-KELVIN
 TOTAL         14.564          6.001            45.114
 ELECTRONIC      .000           .000              .000
 TRANSLATIONAL   .889          2.981            34.609
 ROTATIONAL      .889          2.981            10.500
 VIBRATIONAL   12.787           .039              .005
 Partition functions
```

```
                              Q              LOG10(Q)           LN(Q)
 TOTAL BOT          .561443D-01            -1.250695        -2.880127
 TOTAL V=0          .132155D+09             8.121085        18.699192
 VIB (BOT)          .424961D-09            -9.371650       -21.579023
 VIB (V=0)          .100030D+01              .000129          .000297
 ELECTRONIC         .100000D+01              .000000          .000000
 TRANSLATIONAL      .300436D+07             6.477751        14.915574
 ROTATIONAL         .439749D+02             1.643204         3.783618
```

```
$ freqchk solvent.chk                          Checkpoint filename can be placed on the command line
 Write Hyperchem files? n
 Temperature (K)? [0=>298.15] 300                                   Alternate temperature.
 Pressure (Atm)? [0=>1 atm] 1.5                                     Alternate pressure.
 Scale factor for freqs during thermochem? [0=>1/1.12] 1            No scaling.
 Do you want to use the principal isotope masses? [Y]: n
 For each atom, give the integer mass number.
 In each case, the default is the principal isotope.
 Atom number 1, atomic number 8: [16]                        Return accepts default.
 Atom number 2, atomic number 1: [1] 2                       Specify isotope masses as integers.
 Frequency output follows ···
```

Frequency output follows, reflecting the values specified above. Note that if scaling is specified, only the thermochemistry data reflects it; the frequencies themselves are not scaled.

An additional prompt sometimes appears in a freqchk session:

```
Project out gradient direction? [N]
```

This prompt appears when the forces are significantly non-zero. A possible reason for the forces being non-zero is that the frequencies were done at a point along the IRC, so projecting out the gradient direction may be useful. If the forces are non-zero in other circumstances, the structure is not a stationary point, and its geometry should be optimized in order to obtain a meaningful vibrational analysis. However, if you want to look at the frequencies, for example, to see if the starting point for an optimization has the correct curvature, then the direction of the gradient should not be projected out.

As an alternative to interactive mode, you can specify all freqchk input on the command line, as in this example, which performs the same operation as the final interactive session above:

```
$ freqchk solvent.chk N 300 1.5 1 N N
```

You will be prompted for the isotopes if the second-to-last parameter is N. The final parameter is the answer to the gradient direction prompt should it appear; if this parameter is omitted and the prompt is relevant, the utility will prompt you.

Selecting Modes. The following command and input selects modes 1-5 and any others involving nitrogen atoms, sending the output to the file ala_freq.out:

```
$ freqchk ala.chk -sel -o ala_freq.out N 0 0 0 N N
1-5 atoms=N
                                    Input ends with a blank line.
$ more ala_freq.out                 Frequency data is ready for viewing.
...
```

6.8 freqmem

The freqmem utility takes parameters for a frequency calculation and determines the amount of memory required to complete all steps in one pass for maximum efficiency. All parameters must be provided on the command line, using the following syntax:

freqmem *natoms nbasis* r|u *function*

The arguments are:

natoms

Number of atoms in the molecule.

nbasis

Number of basis functions for this system under the desired basis set.

r|u

A one-letter code indicating an RHF (closed shell) or UHF (open shell) calculation, as appropriate.

function

A letter indicating the highest angular momentum basis function in the basis set: i.e., d for d functions, f for f functions, h for h functions, and so on. This parameter may also be a string indicating the types of basis functions used in the chosen basis set: sp, spd or spdf (as was used in earlier versions of this utility).

6.8.1 Example

This examples estimates the memory resources required for RHF/STO-3G frequencies on taxol (113 atoms):

$ freqmem 113 361 r p

```
 RHF direct frequencies with sp functions:
 One pass requires 44.80 megawords.
```

The output indicates that the program will require about 360 MB of memory to complete the frequency calculation in a single pass.

If the amount of memory specified by freqmem is not available, a frequency calculation can still be completed using multiple passes. Use the %Mem Link 0 command to specify the amount of available memory. Setting this parameter to one half or one third of the amount of memory recommended by freqmem is often a good choice.

The number of basis functions used in a Gaussian calculation is printed out early in the output file. It may also be calculated by setting up an input file for the job in question and including the %KJob=301 Link 0 command, which tells the program to terminate as soon as Link 301 is reached (which is almost immediately). The number of basis functions used for the molecule with the specified basis set may then be retrieved from the log file with a command like this one:

$ grep "basis func" *name*.log

```
     361 basis functions        1083 primitive gaussians
```

6.9 gauopt

The gauopt utility performs an optimization by repeatedly executing Gaussian. In this way, it can optimize any parameter in the input stream, including general or massaged basis functions. It operates by repeatedly creating subprocesses running Gaussian. The gauopt utility is typically used to optimize parameters such as basis functions for which there is no standard optimization method implemented within Gaussian. It is invoked by its command verb, gauopt, and takes its input from standard input.

Input for gauopt consists of a template file, in which certain fields are replaced with variables whose values are to be optimized. The template file is used to construct an actual Gaussian input file containing the current values of the variables for each energy evaluation. The energy is then computed at each step automatically by running a Gaussian single point calculation. The format for the first line of the template is:
NVar, MaxIt, SaveFlag, Conv, ConvV
using a format `2I3, L2, D9.2`. The fields are defined as follows:

NVar

The number of variables.

MaxIt

The maximum number of optimization cycles to perform.

T|F

A logical flag indicating whether the intermediate Gaussian output files are to be saved. These are named fork.com, fork.log, fork.rwf, and so on. They are deleted by default, but can be saved as an aid in debugging the template input.

Conv

Convergence on the RMS change in the variables. A fairly tight default is provided if this parameter is set to 0.0.

ConvV

Convergence on the energy, which defaults to 1 milliHartree when the parameter is set to 0.0.

The next line of the template file has one or more pairs of values using the following syntax:
Value C|V *Repeated NVar times (no internal spaces)*
where *Value* is the value for the variable, and the second value is a one-character flag which can be set to C to constrain the variable (i.e., not optimize it during the current run) or to V if the variable is to be optimized. This line uses a format of F14.9, A1 for each pair of values.

The remainder of the template file contains a Gaussian input file template. Each field in the input file where a previously-defined variable should be inserted should contain:
<*n x.y*>
indicating that the n^{th} variable should be inserted at that point using format Fx.y. If n is preceded by a minus sign, then -1 times the specified variable will be inserted there.

6.9.1 Example

An example will help make all of these concepts clearer. The following gauopt template file optimizes the scale factors in the STO-2G expansion of a minimal basis set for water:

```
  3 10 T      0.00       0.00
   7.660000000V    2.250000000V    1.240000000V
```

```
# RHF/Gen Test

Water RHF/STO-2G basis with optimized scale factors

0,1
O
H,1,r
H,1,r,2,a

r 0.96
a 104.5

1 0
sto 1s 2 <1 12.10>
sto 2sp 2 <2 12.10>
****
2 0
sto 1s 2 <3 12.10>
****
3 0
sto 1s 2 <3 12.10>
****
```

The scale factors on the two hydrogens are made equal by using the same gauopt variable in more than one place; of course, this same effect could also have been accomplished by specifying that the same basis was to be used on every hydrogen atom.

6.10 ghelp

The ghelp utility is a hierarchical help facility for Gaussian. Typing ghelp alone will display general information and a list of topics for which help is available. The form ghelp topics will display just the list of topics.

Information about Gaussian keywords and options is available using the format:
ghelp route *keyword* [*option*]

Information about Gaussian utilities may be accessed using either the utility name as the primary topic or via the topic *utilities*. Information about internal option *m* in overlay *n* (IOp(*n*/*m*)) may be obtained using the following command on Unix systems (note the quotation marks):
$ ghelp "ov*n* iop(*m*)"

6.11 mm

Standalone molecular mechanics program. This program reads a Gaussian input file from standard input and writes a new input file with the (possibly optimized) structure to standard output. The desired force field must be selected via the -Dreiding, -UFF, -Amber or -Param option (see Options for the latter). The type of job to run is specified with the -Force, -Freq, -Opt, and -Micro command line options; the default is an energy calculation. -Micro optimizes only the atoms that are in the low ONIOM layer (i.e., the real system), in order to preoptimize the MM portion of the molecule.

6.11.1 Options

-Param *N*

Use force field *N* (same as IOp(1/64)=*N* within Gaussian).

-ReadParam

Read in additional parameters. Internally-stored parameters have priority over read-in parameters.

-ReplaceParam

Read in additional parameters. Read-in parameters take priority over the internally-stored parameters.

-OptCyc *N*

Specify the maximum number of optimization cycles to *N*.

-ReadCon

Read connectivity information from the input file (i.e., the input file uses Geom=Connect).

-Test *N*

Set the debugging flag to *N* (higher numbers result in more debugging output).

-TRScale *num*

Use scaling scheme *num* for rigid translations/rotation: 0=no scaling (the default); 1 says to scale the *N* atoms in a rigid block by 1/*N*; 2 says to scale the *N* atoms in a rigid block by 1/SQRT(*N*), and a negative value scales by |*N*|/1000.

-External

Read input and write output in the formats used by the External keyword.

6.12 newzmat

The newzmat utility was designed primarily for converting molecule specifications between a variety of standard formats. It can also perform many related functions, such as extracting molecule specifications from Gaussian checkpoint files. Its full set of capabilities includes the following:

- ◊ newzmat can convert molecule specifications between a variety of data file formats. This includes generating a Z-matrix (and hence input for Gaussian) from the files produced by other programs and also converting between the file formats of any of these programs. The newzmat keyword can thus be used to produce Gaussian input from the data files of many popular graphics and mechanics packages, allowing them to act as graphical input front-ends to Gaussian. The resulting data files have the proper symmetry constraints for efficient computation (if applicable).
- ◊ newzmat can also generate Gaussian 16 checkpoint files from other data files, and (more importantly) generate the data files from checkpoint files. This capability can be used to extract data for display with a visualization package.
- ◊ newzmat can retrieve intermediate structures from a checkpoint file from (or during) a geometry optimization, for reuse or display.

Known Difficulties with newzmat

- ◊ The symmetry averaging process, which guesses the intended symmetry given coordinates which are only approximately symmetric, does not always achieve the intended symmetry. It will take coordinates printed in a Gaussian output file to 6 digits and restore symmetry, and it will usually work given coordinates from molecular mechanics provided that the mechanics optimization was converged reasonably far. In generating coordinates with MacroModel, for example, it is sometimes necessary to do a final full Newton-Raphson step after the normal minimization.

◊ newzmat computes the nuclear repulsion energy of the initial read-in structure and of the final structure as a consistency check. If these disagree, a warning is printed. Substantial disagreement indicates a failure of the program.

6.12.1 Syntax

newzmat has the following general syntax:

newzmat *option(s) input-file output-file*

where *option(s)* is one or more options, specifying the desired operations, *input-file* is the file containing the structure to be converted (or retrieved), and *output-file* is the file in which to place the new molecule specification (or Gaussian input). A slightly variant syntax is used when merging information from two files (see below).

If the output filename is omitted, it is given the same base name as the input file, along with a conventional extension denoting its file type. In general, extensions can be omitted from file specifications provided that extension conventions are followed. The default extensions are listed in the following table:

Extension	Description	Option Form
.bgf	Biograf internal data file	bgf
.cac	CaChe molecule file	cache
.chk	Gaussian 09 checkpoint file	chk
.com	Gaussian input file (Z-matrix mol. spec.)	zmat
.com	Gaussian input file (Cart. coords. mol. spec.)	cart
.con	QUIPU system data file	con
.dat	Model/XModel/MM2 data file	model
.dat	MacroModel data file (may be formatted or unformatted)	mmodel, ummodel
.ent	Brookhaven data file (≡PDB)	ent
.com	Fractional coords. for crystal structures (requires exactly 3 trans. ops.)	fract
.inp	MOPAC input file	mopac
.pdb	Protein Data Bank format (≡Brookhaven)	pdb
.ppp	Some PPP program (output only)	ppp
.xyz	Unadorned Cartesian coordinates	xyz
.zin	Ancient version of ZIndo	zindo

6.12.2 Options

The options specifying the formats of the input and output molecule specifications are formed from the string -i or -o (respectively), followed immediately by the appropriate option form string from the preceding table corresponding to the desired molecule specification format (no spaces intervene). For example, -ipdb indicates that the input molecule specification is in PDB format and that the extension .pdb should be applied to the input filename if no extension is specified. Similarly, -oxyz specifies an output format of Cartesian coordinates along with a default extension of .xyz for the output filename. The default input and output options are -izmat and -ozmat. Note that -izmat and -icart are synonyms, and either one of them can read a Gaussian input file containing any molecule specification format: Z-matrix, Cartesian coordinates, or mixed internal and Cartesian coordinates.

Selecting An Output Format

In order to communicate with a non-supported visualization system, the first choice of format to try is the PDB file. This format includes the connectivity information and is widely supported. Note that some software packages use the .ent extension, rather than .pdb; the -ient and -oent options select the former, while -ipdb and -opdb select the latter. Another commonly used alternative is the Mopac file format.

The following options further specify the input for newzmat:

-ngeom *N*

Use geometry from N^{th} structure on the checkpoint file. This functions in the same manner as Geom=(NGeom=*N*).

-ot *list*

Use geometries from the listed structures on the checkpoint file. Lists can include multiple structure numbers (separated by commas) and ranges of structure numbers. For example, -ot 3,7 extracts structures from steps 3 and 7, and -ot 2-5 extracts all structures ranging from steps 2 through 5.

-step *N*

Use the structure from step *N* of the geometry optimization data in a Gaussian 16 checkpoint file (valid only for the -ichk input option).

This option is not available for optimizations in redundant internal coordinates (the default coordinate system). Instead, retrieve the structure from the checkpoint file in a subsequent job by using a route section containing Geom=(Check,Step=*N*).

-ubohr

Input distances in input file are specified in Bohr (the default is Angstroms).

-urad

Input angles in input stream are specified in radians (the default is degrees).

The following options retrieve changes from the checkpoint file and apply them as the MM charges for both regular atoms and link atoms. These options specify which kind of charges to retrieve:

-qmul

Mulliken charges.

-qesp

ESP-fit charges.

-qaim

AIM charges.

-qnpa

NPA charges.

-qapt

APT charges.

The following options further specify the output file format:

-mof1

Use macromodel format 1 (only valid with -ommodel).

-mof2

Use macromodel format 2 (this is default if -ommodel is specified).

-optprompt

Prompt for which parameters should be optimized; used when setting up a molecule specification destined for a geometry optimization and -ozmat is specified (or no output option is included). By default, all parameters not fixed by symmetry are optimized.

-prompt

Prompt for route section and title section lines and for the charge and multiplicity when using -ozmat (or no output option is specified). Gaussian input files produced by newzmat set up HF/6-31G(d) single point energy calculations by default.

6.12.3 Examples

The following command reads the molecule specification from the PDB file water.pdb and writes a Gaussian input file, including the equivalent Z-matrix, to the file h2o.com:

```
$ newzmat -ipdb water h2o                -ozmat is the default, so it can be omitted.
 Charge and multiplicity [0,1]?          A return accepts the default values shown.
```

newzmat prompts for the charge and multiplicity for the Z-matrix since these items cannot be determined from the PDB file.

The following command reads the molecule specification from the Gaussian 09 checkpoint file job-11234.chk and writes the PDB file propell.pdb:

```
$ newzmat -ichk -opdb job-11234 propell
```

The following command reads the molecule specification from step 5 of the optimization from the checkpoint file newopt.chk and produces the Mopac file step5.inp:

```
$ newzmat -ichk -omopac -step 5 newopt step5
```

The following command reads the molecule specification from the Mopac file newsalt.inp and writes a Gaussian input file including the equivalent Z-matrix to the file newsalt.com, prompting for the route and title sections and the charge and spin multiplicity for the molecule:

```
$ newzmat -imopac -prompt newsalt
Percent or Route card? # B3LYP/6-31G(d,p) Opt
Route card?            End route section with a blank line.
Titles? Optimization of caffeine at B3LYP/6-31G**
Titles?                End title section with a blank line.
Charge and Multiplicity? 0,1
```

Merging Information From Two Files Using Newzmat

newzmat can create an input file where data from two different files have been combined. This feature is helpful for setting up ONIOM calcuations in which one wants to apply a custom setup (ONIOM layers, MM atom types, MM atom charges, etc.) from one file to another structure on a different file. It allows you to specify the custom setup only once, and then later apply that information to other structures.

The -t and -s options request and control the creation of the merged input file. A merge command has the following general form:

$ newzmat -i*type* [**-o***type*] **-t***format* **-s***itemn files*

The command requires that you specify the locations of the existing input file (input), the new input file you are creating (output), and the template file (template). The order of the input, output and template files on the command line varies depending on whether the input and/or template file(s) are Gaussian checkpoint files:

Is checkpoint file:

input?	**template?**	*files* **arguments**
n	n	*input output* ignore *template*
n	y	*input output template*
y	n	ignore *output input template*
y	y	*input output template*

In the preceding table, “ignore” is a placeholder.

-t requires a file type argument like -i and -o, and it accepts the same values.

The -s option, which can be specified multiple times, indicates the various information for the new input file. It accepts the following keywords, each of which is followed by either 1 or 2, which indicates that the item should be taken from the input or template file (respectively):

XYZ*n*	Geometry (coordinates, nuclear charges, etc.).
MMT*n*	MM types.
MMC*n*	MM charges. If copied, also copied to link atoms if there is ONIOM data present.
PDB*n*	PDB information (including secondary structure).
Con*n*	Connectivity.
Oni*n*	ONIOM layer and link atom data.
Mic*n*	MicOpt (freezing/optimizing atoms and rigid block info).

The default for all items is the input file (i.e., file 1).

Here is an example. We have a Gaussian input file, allsetup.gjf, containing the MM atom type, MM charges, PDB information, connectivity, ONIOM layers, and so on. We can apply all of these settings to the structure in the PDB file new.pdb with the following command:

$ newzmat -icart -tpdb -sXYZ2 -sCon2 allsetup.gjf newinput.gjf ignore new.pdb

This command takes only the geometry and connectivity from file 2 – the template file new.pdb – and everything else (MM atom types, MM charges, PDB information ONIOM data and optimize/freeze atom data) from file 1, the input file allsetup.gjf. The output file, newinput.gjf, will be the result of this merge.

The other options to newzmat are concerned with generating connectivity information, with the use of standard geometrical parameters, and with the determination and use of molecular symmetry. A complete connectivity table can be used to generate Z-matrix specifications suitable for inclusion of symmetry constraints. Such a table is also required for output of the data files for the molecular mechanics programs. If one of the input formats which includes full connectivity is used (e.g., MacroModel data files), the connectivity that it provides is used.

However, when Z-matrix or MOPAC format input is provided, only the connectivity information which is implied by the internal coordinate specification is available. Thus if a new Z-matrix which incorporates the molecular symmetry is to be generated, the remaining connectivity information must be generated. When Cartesian coordinates are read in, naturally, no connectivity information is provided, so the default is to generate the table using the internally stored atomic radii. In addition, when used to generate input structures, the mechanics programs may not generate suitable bond distances and often produce coordinates which are close to but not exactly symmetric. Options control how each of these cases is handled.

-allbonded

In generating new connectivity information, assume all atoms are bonded.

-bmodel

Use standard model B bond lengths along with internal values in determining bond distances.

-density *N*

Generate natural orbitals for density number *N*. This option is only useful if you are generating a CaChe file. *N* should be set to 0 for HF, to 2 for MP2, to 6 for CI, and to 7 for QCISD or CCD.

-fudge

Fudge bond distances to make sure they are reasonable, using internal values. This is the default for model input and is not applicable elsewhere.

-gencon

Generate connectivity information using internal radii.

-getfile

Insist on filename specifications for all arguments, making standard input and output unacceptable.

-lsymm

Use loose cutoffs for determining symmetry. This option implies -symav.

-mdensity *M*

Subtract generalized density *M* from that specified with -density to make a difference density, which is then converted to natural orbitals.

-nofudge

Do not fudge bond distances. This is the default and only choice for all cases except model input.

-nogetfile

Cancels -getfile.

-noround

Turns off rounding of Z-matrix parameters.

-nosymav

Turns off averaging of input coordinates.

-nosymm

Turns off all use of symmetry.

-round

Rounds Z-matrix parameters to 0.01 Åand 1 degree.

-symav

Average input coordinates using approximate symmetry operations to achieve exact symmetry.

-symm

Assign molecular symmetry.

-tsymm

Use tight cutoffs for determining symmetry. The option is the default.

-rebuildzmat

Build a new Z-matrix rather than using the read-in one (as would be the default for Z-matrix or MOPAC input). This option implies -gencon, and the option may be abbreviated as -redoz.

6.13 testrt

testrt is a utility which takes a standard Gaussian route as input and produces the equivalent non-standard route. The route is usually specified on the command line (enclosed in quotation marks):

$ testrt "# rhf/sto-3g"

If it is not included on the command line, testrt will prompt for the route to be tested.

If the specified route is valid, testrt will print out the non-standard route corresponding to it. If syntax errors are present, then error messages will be displayed. Thus, testrt can be used to verify the syntactic correctness of route sections even by users who understanding nothing of non-standard routes.

6.13.1 Example

Here are some example runs of testrt:

$ testrt "# ccsd(modredun)/6-31G* scf=driect"

```
 -----------------------------------
 # ccsd(modredun)/6-31G* scf=driect
 -----------------------------------
 QPErr —- A syntax error was detected in the input line.
 # ccsd(modredundant)
     '
```

ModRedundant is a valid option, but not for CCSD.

$ testrt "mp4 nmr"

NMR is not available for MP4.

```
 -------
 mp4 nmr
 -------
 NMR only for HF, DFT, and MP2.
```

$ testrt

```
 Please type in the route specification, terminated with a blank line:
```

# HF/6-31G(d) Opt=QST2

End of testrt input.

```
 ----------------------
 # HF/6-31G(d) Opt=QST2
 ----------------------
 1/5=1,18=20,27=202,38=1/1,3;
 2/9=110,12=2,17=6,18=5,40=1/2;
...
```

As the first example indicates, only the first error within the route section is flagged. The second example illustrates the error message from an invalid combination of keywords. The final example shows the output from a successful route test.

Note that testrt cannot detect keyword usage errors; it checks only the syntax of the given route section. Thus, it will not warn you that including the MP2 keyword twice within the route section will have unexpected results (running an MP4 job).

testrt's output can be redirected to a file by standard UNIX output redirection:

$ testrt "# rhf/sto-3g" >*output-file*

6.14 unfchk

This utility is the opposite number to formchk. It converts a formatted checkpoint file to a binary checkpoint file, in a format appropriate to the local computer system:

$ unfchk

```
 Formatted Checkpoint file? water
 Read formatted file water.fchk
 Write checkpoint file water.chk
```

The utility applies the extension .fch to the specified filename on Windows systems and the extensions .fchk on other computer systems.

The formatted checkpoint filename can also be given on the command line:

$ unfchk job222.chk

```
 Read formatted file "job222.fchk"
 Write checkpoint file "job222.chk"
 LenFI=          2580 MDV=     104857600 DoIMCk=T
```

Note that formatted checkpoint files can be used as a data exchange format between computer platforms. Use formchk on the originating computer and unfchk on the target computer to create a binary checkpoint file.

unfchk can also convert a matrix element file to a binary checkpoint file (see *Options*)

6.14.1 Options

In addition to formatted checkpoint files, unfchk can work with matrix element files and create binary checkpoint files containing their data.

-matrix ***file1 file2***

Reads (fortran unformatted) matrix element file *file1.dat* and writes checkpoint file *file2.chk*

-rawmatrix ***file1 file2***

Reads (raw binary) matrix element file *file1.dat* and writes checkpoint file *file2.chk*

The following options specify the format of the matrix element file. They may be needed, depending on how the file was written by formchk, Gaussian, or whatever program created it.

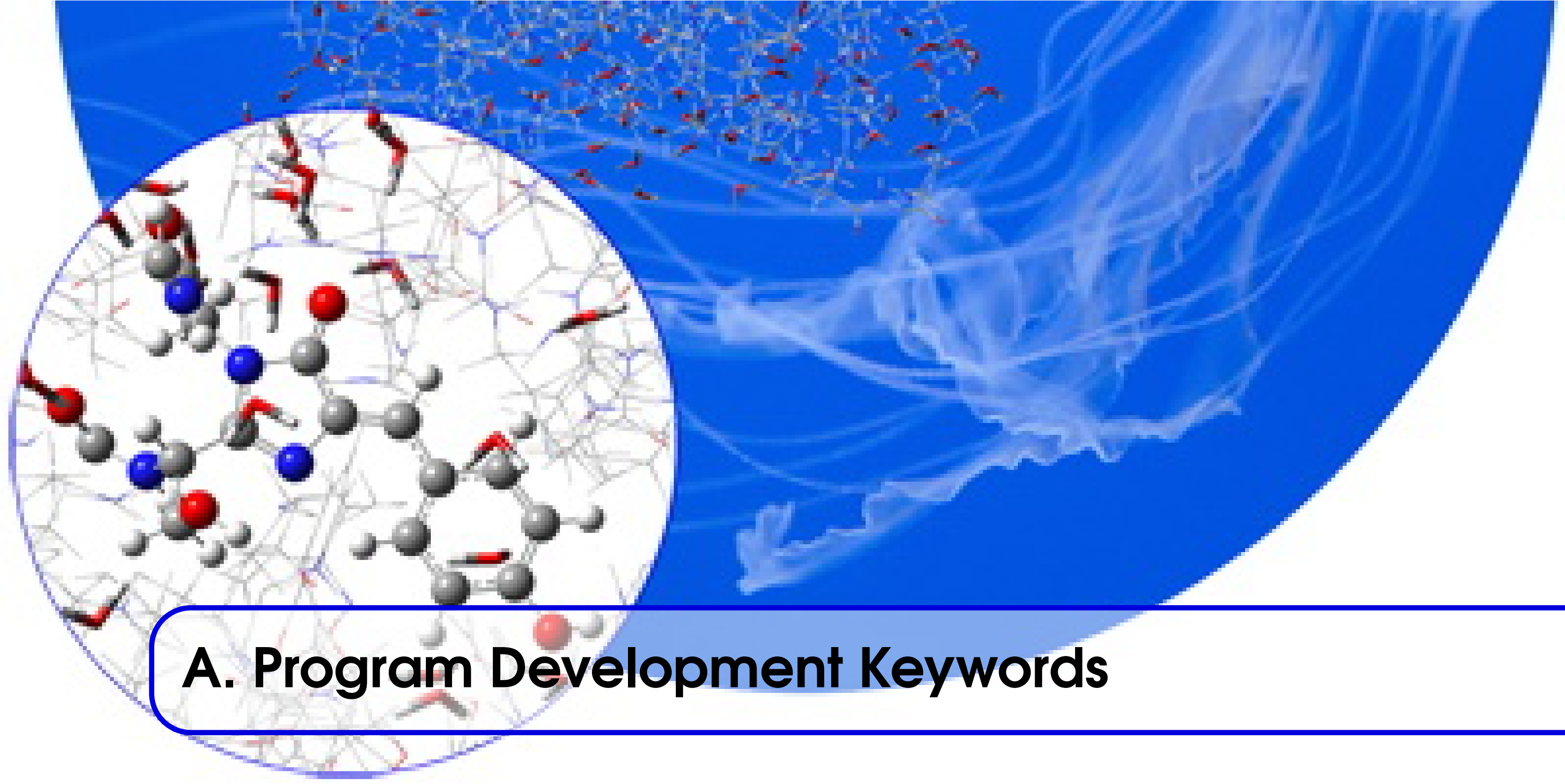
-i4

Use this option when the Fortran unformatted matrix element file uses 4-byte integers.

-i2

Use this option when the Fortran unformatted matrix element file was written using 2-byte integers, for the two electron integral labels (and 4-bytes for most other integers).

Part Three: Appendix



A. Program Development Keywords

This section documents keywords and options useful for developers who are extending and/or interfacing to Gaussian 16. It also discusses non-standard routes and the determination of the standard orientation.

A.1 Keywords

The keywords and options described here are useful for developing new methods and other debugging purposes, but are not recommended for production level calculations.

A.1.1 General Job Restart

We discuss here the general use of Restart, designed for debugging. See the Restart section for production use. This keyword restarts a calculation by reusing the read-write file. The form Restart L1 reuses the read-write file but generates a new route.

Restarts using the original route can specify the occurrence of a particular link and whether to clean up or retain overlay and link-volatile files using the following syntax:

#P Restart [**L***n*[(*m*)]] [**Clean|KeepOverlay|KeepAll**]

When all parameters are specified, the job restarts at the *m*th occurrence of Link *n*. Clean requests that all routine and overlay volatile files be removed by Link1, KeepOverlay requests that overlay-volatile files be retained but not link-volatile ones, and KeepAll retains everything. The default is to KeepAll if the read-write file is set up for an intra-link restart and Clean otherwise.

A.1.2 IOp Setting Keywords

IOp1=*keyword*

This keyword controls various details of the operating system interface. The options are standard, but not all are implemented (or even relevant!) in every version.

FileIODump	Dump FileIO tables at the end of each link. FDump is a synonym for this keyword.

TimeStamp	Turn on time stamping. TStamp is a synonym for this keyword.
FileIOPrint	Turn on additional debug print in FileIO.
Synch	Currently a no-op (appears in a few test jobs).
NoDFTJ	Turn off use of the pure Coulomb term for non-hybrid DFT (seldom useful in production jobs).
AbelianOnly	Force the use of only abelian symmetry (seldom useful in production jobs).
NoPackSort	Turn off packing addresses into 32-bits during sorts, even if the address space being sorted is $< 2^{31}$ (seldom useful in production jobs).

IOp2

This option sets the maximum amount of memory which will be dynamically allocated. MDV and Core are synonyms for IOp2.

IOp33

This sets the standard debug print option as specified.For example, the following sets IOp(33) to 3 in all invocations of overlay 2, and IOp(33) to 1 in all invocations of overlay 7:

```
IOp33(2=3,7=1)
```

The *Gaussian 16 IOps Reference* also documents all internal options (IOps).

A.2 Options

CPHF

The following options are used for debugging:

KeepMicro	Keep all EE centers in CPHF, even for Opt=CalcFC or Opt=CalcAll with non-quadratic microiterations, where atoms that are not used in internal coordinates need not be included in the CPHF.
NoReuse	Do not reuse the electric field CPHF solution in the 2^{nd} (nuclear) CPHF during frequency calculations. The default is ReUse.
XY	Treat real and imaginary perturbations together. The opposite is NoXY, which does them separately. The default is to treat them separately if nuclear perturbations are also being done, but to treat them together if there are only electromagnetic perturbations.
ZVector	Use the Z-Vector method [225–227] for post-SCF gradients. Allowed and the default if Hartree-Fock 2nd derivatives are not also requested. The NoZVector keyword says to use the full $3 \times N_{Atoms}$ CPHF for post-SCF gradients.

FMM

The following options are available for debugging:

LMax=*N*	Specifies the maximum order multipole. The default is 25.
Levels=*N*	Specifies the number of levels to use in the FMM. The default is 8 for molecules and is adjusted dynamically for PBC.

Tolerance=N	Specifies the accuracy level as 10^{-N}. The default values for N are 11 except for pass 0 of the SCF where it is 7.
JBoxLen=N	Sets the minimum box length (size) to N/1000 Bohrs when doing J. By default, N is 2.5. The maximum of JBoxLen and KBoxLen is used if J and K are done at the same time. BoxLen is a synonym for JBoxLen.
KBoxLen=N	Sets the minimum box length (size) to N/1000 Bohrs when doing K. By default, N is 0.75. The maximum of KBoxLen and JBoxLen is used if K and J are done at the same time.
AllNearField	Turn on all near-field in FMM.
NoParallelCPHF	Forbid parallel execution in FMM during the CPHF phase. NoParCPHF is a synonym for this option.

Integral

The following options are used for debugging:

CNDO	Do calculation in main code using CNDO/2 integrals.
INDO	Do calculation in main code using INDO/2 integrals.
ZIndo1	Do calculation in main code using ZIndo/1 integrals.
ZIndoS	Do calculation in main code using ZIndo/S integrals.
DPRISM	Use the PRISM algorithm [782] for spdf integral derivatives. This is the default.
Rys1E	Evaluate one-electron integrals using the Rys method [783–785], instead of the default method. This is necessary on machines with very limited memory.
Rys2E	If writing two-electron integrals, use Rys method (L314) [783–786]. This is slower than the default method, but may be needed for small memory machines and is chosen by default if regular (non-Raffenetti) integrals are requested (by the NoRaff option).
DSRys	Use scalar Rys integral derivative code. Can combine with Berny for df only using Rys.
Berny	Use Berny sp integral derivative and second derivative code (L702).
Pass	Pass specifies that the integrals be stored in memory via disk, and NoPass disables this. Synonymous with SCF=[No]Pass, which is the recommended usage.
NoJEngine	Forbid use of special Coulomb code.
NoSP	Do not use the special sp integral program (L311) when writing integrals to disk.
RevDagSam	Reverse choice of diagonal sampling in Prism.
NoSchwartz	Do not use Schwartz integral estimates (only use the heuristic set). Schwartz says to use the Schwartz integral estimates in addition to the heuristic set. The default is to use both.
RevRepFock	Reverse choice of Scat20 vs. replicated Fock matrices.
NoDFTCut	Turn off extra DFT cutoffs.
SplitSP	Split AO S=P shells into separate S and P shells. NoSplitSP is the default.
SplitSPDF	Split AO S=P=D and S=P=D=F shells into S=P, D, and F. NoSplitSPDF is the default.

SplitDBFSPDF	Split density S=P=D and S=P=D=F into S=P, D, and F. NoSplitDBFSPDF is the default.
NoGather	Forbid use of gather/scatter digestion, even when processing small numbers of density matrices. Splatter is a synonym for this option.
ForceNuc	Do nuclear-electron Coulomb with electron-electron.
SepJK	Do J and K in HF/hybrid DFT separately for testing.
Seq2E	Set up for parallel 2 electron integral evaluation but then do not run in parallel (for debugging).
SeqXC	Set up for parallel 2 electron integral evaluation but then do not run in parallel (for debugging).
SeqLinda	Cause Linda workers to run sequentially. Currently just makes the Linda workers other than the master run simultaneously but before the master.
BigAtoms	Make all atom sizes large in XC quadrature.
BigShells	Make all shell sizes large in XC quadrature.
NoSymAtGrid	Do not use (Abelian) symmetry to reduce grid points on symmetry-unique atoms.
LinMIO	Convert to linear storage in FoFCou for testing.
RevDistanceMatrix	Reverse choice of whether to precompute distance matrix during numerical quadrature. The default is to precompute for molecules but not for PBC.
NoDynParallel	Turn off dynamic work allocation.

Sparse

The following options are used for debugging:

Loose	Sets the cutoff to $5 * 10^{-5}$.
Medium	Sets the cutoff to $5 * 10^{-7}$. This is the default for semi-empirical methods.
Tight	Sets the cutoff to $1 * 10^{-10}$. This is the default for DFT methods.
N	Sets the cutoff to $1 * 10^{-N}$.

A.2.1 Changing Link Invocation and Ordering

ExtraLinks

This requests that additional links be executed. They are added to all instances of their overlay after the regular links. For example, ExtraLinks=L9997 will cause each instance of overlay 99 to include links 9999 (by default) and 9997, in that order.

ExtraOverlays

This command requests that extra overlay cards be read in non-standard route format and inserted into the standard route immediately before the final (overlay 99) card.

Skip

Skip initial overlay cards in the route. Skip=OvNNN skip until first occurrence of overlay NNN. Skip=M skip first M cards.

Use=L*nnn*

This specifies alternate routes through the program. The following options are available:

L123	Use L123 instead of L115 for IRC. This is the default for IRC, except for IRCMax jobs.
L402	Use old link 402 code for semi-empirical.
L503	Use link 503 for SCF.
L506	Use link 506 for ROHF.

A.3 The Standard Orientation

Before a calculation is performed, a molecule can be reoriented to a different coordinate system, called the standard orientation, with the use of molecular symmetry. In geometry optimizations, reorientation occurs at every step; the program then checks if the standard orientation of a molecule has flipped by 180 degrees during an optimization and avoids the flip. This avoids jumps when animating optimizations, IRCs, etc. in GaussView and improves SCF convergence.

This section describes the goals, factors to consider, and various rules for positioning axes for the standard orientation of molecules.

A.3.1 Selection Goals

The goals for selecting conventions for standard orientation are:

- ◊ To simplify the 3×3 transformation matrices by reorienting the molecule.
- ◊ Two Z-matrices differing in values of internal coordinates but identical in integer quantities (such as occurs on subsequent points of a geometry optimization) should produce the same standard orientation.
- ◊ Two different Z-matrices for the same molecule should produce the same coordinates, except for a possible renumbering of atoms.
- ◊ Maximizing the number of molecular orbital coefficients which are zero by symmetry.

A.3.2 General Considerations

The factors that should be considered for standard orientation are:

- ◊ A right-handed coordinate system is used throughout the calculation.
- ◊ The molecule is translated so that its center of charge is at the origin.
- ◊ Atoms are not reordered relative to their order upon input.
- ◊ The Cartesian axes are considered to increase in priority in the order $X < Y < Z$.

A.3.3 Rules for Positioning an Axis

Criteria for rotating and aligning an axis are listed below. If rotation is required to meet one of these criteria, it should be a 180 degree rotation about the X, Y, or Z axis, defined as follows:

X	Rotate about X
Y	Rotate about Y
Z	Rotate about Z

An axis of rotation or a principal axis of charge can be aligned with a Cartesian axis in one of two ways – either parallel or antiparallel, depending on the successive application of the following tests until a definite result is achieved:

- ◊ The sum of the coordinates of the atoms of the highest atomic number on the axis must be positive.

- ◊ The third moment of charge must be positive.
- ◊ The sum of the projections of the atomic coordinates onto the reference axis must be positive.
- ◊ The first atom with a non-zero projection on the reference axis must have a positive projection on that axis.

A.3.4 Rules for Positioning Principal Axes of Charge

In the absence of any other rules, the principal axis corresponding to the largest principal moment of charge must be aligned with the highest priority Cartesian axis available. Individual point groups have specific considerations:

C_s	The molecular plane must be made coincident with the XY plane. Note that although this convention conflicts with Mulliken's suggestion, it is consistent with the character tables of Cotton and Herzberg. The molecule is then rotated about the Z axis according to the rules given below for C_n molecules.
C_{2v}	The molecular plane is placed in the YZ plane, following Mulliken's recommendation for planar C_{2v} molecules. The following tests are successively applied for non-planar molecules: (1) The mirror plane with the most atoms is put in the YZ plane; (2) The mirror plane with the most non-hydrogen atoms is put in the YZ plane; (3) The mirror plane with the lowest numbered atom is made coincident with YZ. Finally, the axes of charge rules are applied (as described above).
Planar, D_{2h}	Following Mulliken's recommendation, the molecular plane is placed in the YZ plane. The molecule is rotated about the X axis so that the Z axis can pass through either the greater number of atoms, or, if this is not decisive, the greater number of bonds.
C_n	Follow the rules for general symmetric top molecules.
C_i	Translate but do not reorient.
C_1	Translate but do not reorient.

A.3.5 Special Rules for Symmetric Top Molecules

Symmetric top molecules are distinguished by having two of three moments of inertia equal. The third moment can thus be uniquely identified as the reference axis and the point group is analyzed by considering circular sets of atoms.

The following rules are applied for symmetric top molecules:

- ◊ The unique axis is aligned with the Z axis.
- ◊ A circular-set of atoms is composed of atoms lying in a plane which have the same atomic number, and are equidistant from a reference axis perpendicular to the plane. Atoms on the reference axis are not included in any circular-set. A circular-set of atoms is generated by a proper rotation axis.
- ◊ The key atom in a symmetric top molecule is the atom with the lowest number in the key circular-set. The following tests are carried out successively to find the key circular-set:
  - Which set is nearest the XY plane?
  - Which set has a positive projection on the Z axis?
  - Which set is nearest the Z axis?

- Which set is comprised of atoms with the lowest atomic number?

◊ The orientation is then chosen for the specific point group:

- C_n, C_{nh}, S_n: The molecule is rotated about the Z axis to maximize the number of pairs of heavy atoms parallel to the Y axis. If no such arrangement is satisfactory, then the key atom is placed in the YZ plane to give it a positive Y coordinate.
- D_n, D_{nh}: One of the C_2 axes is made coincident with the Y Cartesian axis. The tests described below are used to decide which C_2 axis is so positioned.
- D_{nd}, C_{nv}: One of the vertical planes is made coincident with the YZ Cartesian plane. The tests below are used to decide which plane is so positioned.

◊ The following are tests for selecting among axes for the D_n, D_{nh}, D_{nd}, and C_{nv} molecules:

- Maximize the projection of the key atom on the Y axis.
- If two orientations give the maximum projection on the Y axis, select one with the maximum projection on the X axis.
- For molecules contained in the XY plane, the standard axis orientation rules (see above) are applied to the X axis to complete the orientation specification.

A.3.6 Special Rules for Spherical Top Molecules

Spherical top molecules are distinguished by having their equal moments of inertia and can be characterized by identifying spherical sets of atoms.

A spherical-set of atoms is composed of atoms which are equidistant from the origin and have the same atomic number. Spherical-sets should be ordered in terms of increasing distance from the origin and of increasing atomic number at any one distance. The key atom is the lowest numbered atom in the first spherical-set.

Although not generally the case, it is possible, with appropriate geometric constraints, to have D_{2d}, D_{2h}, or D_2 molecules that are symmetric tops. Such molecules have three perpendicular two-fold axes that are aligned with the X, Y, and Z axes in accordance with the rules given above.

A.4 Non-Standard Routes

If a combination of options or links is required which is drastically different than a standard route, then a complete sequence of overlays and links with associated options can be read in. The job-type input section begins with the line:

```
# NonStd
```

This is followed by one line for each desired overlay, in execution order, giving the overlay number, a slash, the desired options, another slash, the list of links to be executed, and finally a semicolon:

Ov/Opt=val,Opt=val,···/Link,Link,···;

For example:

```
7/5=3,7=4/2,3,16;
```

specifies a run through the links 702, 703, and 716 (in this order), with option 5 set equal to 3 and option 7 equal to 4 in each of the links. If all options have their default value, the line would be

```
7//2,3,16;
```

A further feature of the route specification is the jump number. This is given in parentheses at the end of the link list, just before the semicolon. It indicates which overlay line is executed after completion of the current overlay. If it is omitted, the default value is +0, indicating that the program will proceed to the next line in the list (skipping no lines). If the jump number is set to -4, on the other hand, as in

```
7//2,3,16(-4);
```

then execution will continue with the overlay specified four route lines back (not counting the current line).

This feature permits loops to be built into the route and is useful for optimization runs. An argument to the program chaining routine can override the jump. This is used during geometry optimizations to loop over a sequence of overlay lines until the optimization has been completed, at which point the line following the end of the loop is executed.

Note that non-standard routes are not generally created from scratch but rather are built by printing out and modifying the sequence produced by the standard route most similar to that desired. This can be accomplished most easily with the testrt utility.

A Simple Route Example. The standard route:

```
# RHF/STO-3G
```

causes the following non-standard route to be generated:

```
1/38=1/1;
2/12=2,17=6,18=5,40=1/2;
3/6=3,11=1,16=1,25=1,30=1,116=1/1,2,3;
4//1;
5/5=2,38=5/2;
6/7=2,8=2,9=2,10=2,28=1/1;
99/5=1,9=1/99;
```

The resulting sequence of programs is illustrated in Figure A.1.

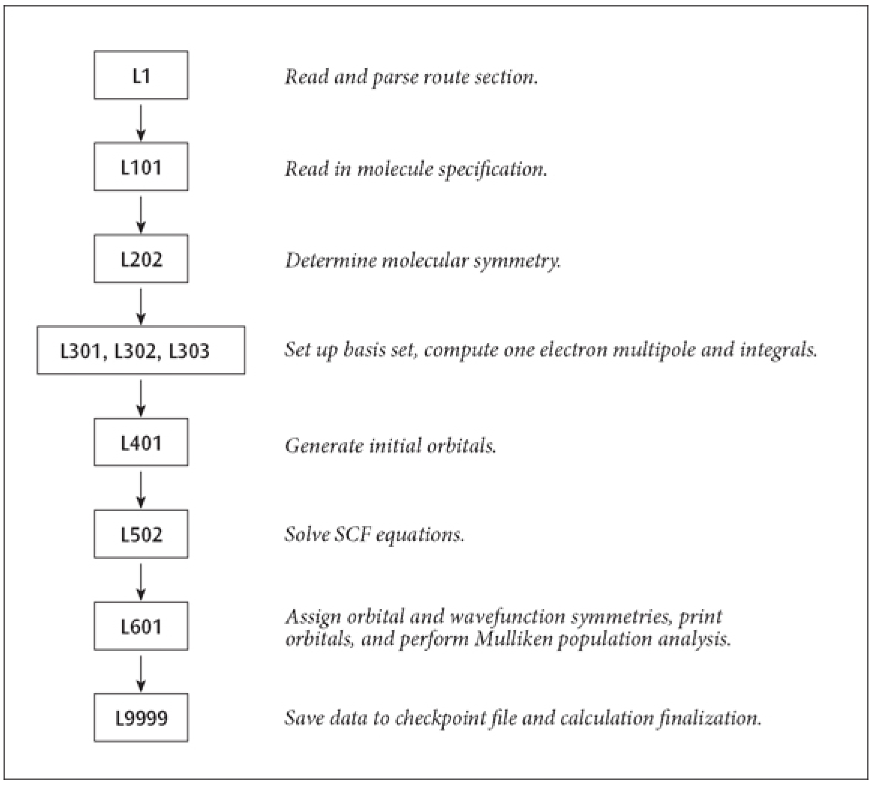

L1
Read and parse route section.
L101
Read in molecule specification.
L202
Determine molecular symmetry.
L301, L302, L303
Set up basis set, compute one electron multipole and integrals.
L401
Generate initial orbitals.
L502
Solve SCF equations.
L601
Assign orbital and wavefunction symmetries, print orbitals, and perform Mulliken population analysis.
L9999
Save data to checkpoint file and calculation finalization.

Figure A.1: A Simple Route Sequence

The basic sequence of program execution is identical to that found in any *ab initio* program, except that Link 1 (reading and interpreting the route section) precedes the actual calculation, and that Link 9999 (writing to the checkpoint file) follows it. Similarly, an MP4 single point has integral transformation (links 801 and 804) and the MP calculation (link 913) inserted before the population analysis (Link 601) and Link 9999. Link 9999 automatically terminates the job step when it completes.

A Route Involving Loops. The standard route:
`# RHF/STO-3G Opt` produces the following on-standard route:

```
 1  1/18=20,19=15,38=1/1,3;
 2  2/9=110,12=2,17=6,18=5,40=1/2
 3  3/6=3,11=1,16=1,25=1,30=1,71=1,116=1/1,2,3;
 4  4//1;
 5  5/5=2,38=5/2;
 6  6/7=2,8=2,9=2,10=2,28=1/1;
 7  7//1,2,3,16;
 8  1/18=20,19=15/3(2);
 9  2/9=110/2;
10  99//99;
11  2/9=110/2;
12  3/6=3,11=1,16=1,25=1,30=1,71=1,116=1/1,2,3;
13  4/5=5,16=3/1;
14  5/5=2,38=5/2;
15  7//1,2,3,16;
16  1/18=20,19=15/3(-5);
17  2/9=110/2;
18  6/7=2,8=2,9=2,10=2,19=2,28=1/1;
19  99/9=1/99;
```

The resulting sequence of program execution is illustrated in Figure A.2.

Several considerations complicate this route:

◊ The first point of the optimization must be handled separately from later steps, since several actions must be performed only once. These include reading the initial molecule specification and generating the initial orbitals.

◊ There must be a loop over geometries, with the optimization program (in this case the Berny optimizer, Link 103) deciding whether another geometry was required or the structure has been optimized.

◊ If a converged geometry is supplied, the program should calculate the gradients once, recognize that the structure is optimized, and quit.

◊ Population analysis and orbital printing are done by default only at the first and last points, not at the relatively uninteresting intermediate geometries.

The first point has been dealt with by having two basic sequences of integrals, guess, SCF, and integral derivatives in the route. The first sequence includes Link 101 (to read the initial geometry), Link 103 (which does its own initialization), and has options set to tell Link 401 to generate an initial guess. The second sequence uses geometries produced in Link 103 in the course of the optimization, and has options set to tell Link 401 to retrieve the wavefunction from the previous geometry as the initial guess for the next.

The forward jump on the eighth line has the effect that if Link 103 exits normally (without taking any spe-

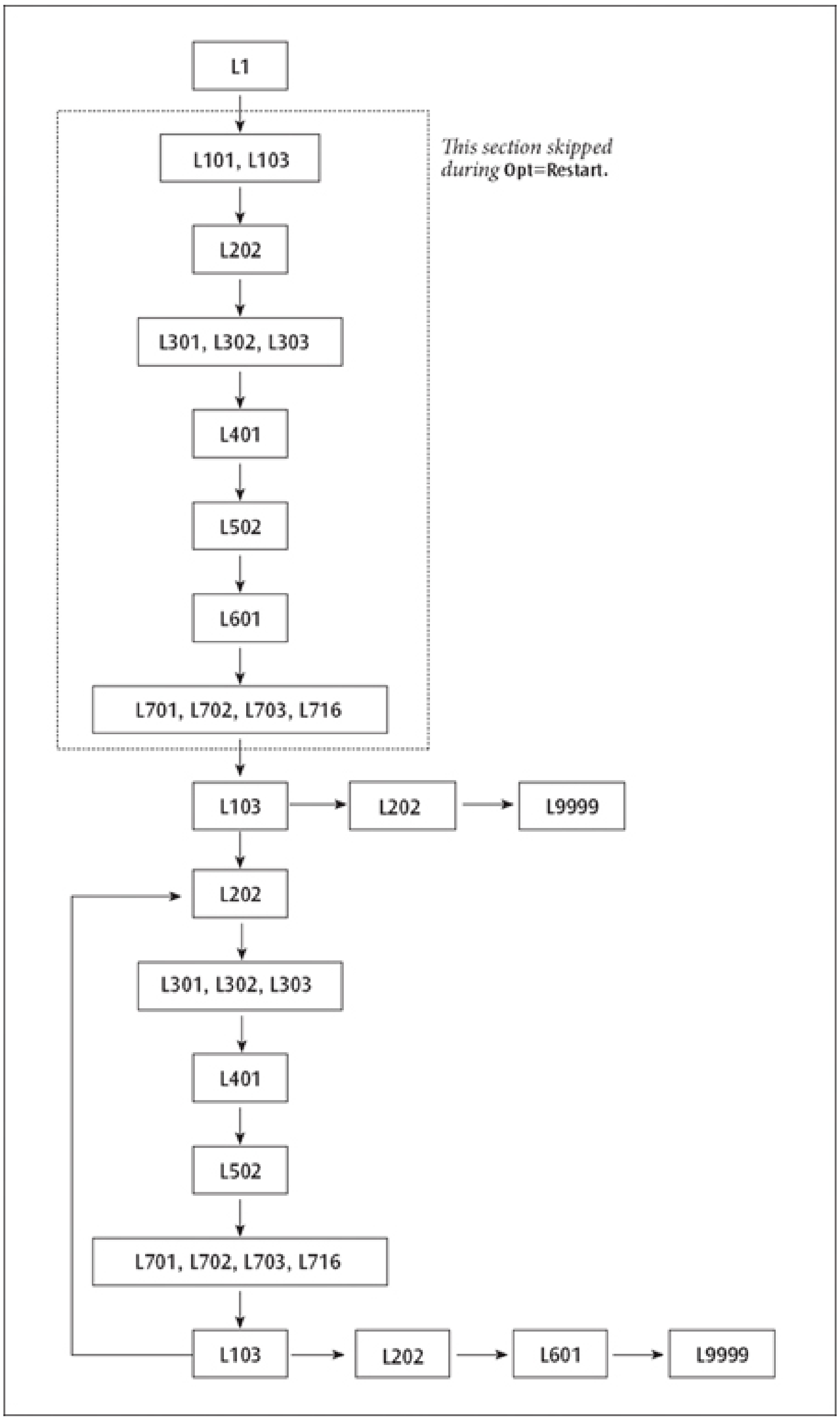


Figure A.2: A Route Involving Loops

cial action), the following lines (invoking Links 202 and 9999) are skipped. Normally, in this second invocation of Link 103, the initial gradient will be examined and a new structure chosen. The next link to be executed will be Link 202, which processes the new geometry, followed by the rest of the second energy+gradient sequence, which constitutes the main optimization loop. If the second invocation of Link 103 finds that the geometry is converged, it exits with a flag which suppresses the jump, causing Links 202, 601 and 9999 to be invoked by the following lines and the job to complete.

Lines 11-16 form the main optimization loop. This evaluates the integrals, wavefunction, and gradient for the second and subsequent points in the optimization. It concludes with Link 103. If the geometry is still not converged, Link 103 chooses a new geometry and exits normally, causing the backward jump on line 16 to be executed, and the next line processed to be line 11, beginning a new cycle. If Link 103 finds that the geometry has converged, it exits and suppresses the jump, causing the concluding lines (17-19) to be processed.

The final instance of Link 601 prints the final multipole moments as well as the orbitals and population analysis if so requested. Finally, Link 9999 generates the archive entry and terminates the job step.

MP and CI optimizations have the transformation and correlation overlays (8 and 9) and the post-SCF gradient overlays (11 and 10, in that order) inserted before overlay 7. The same two-phase route structure is used for numerical differentiation to produce frequencies or polarizabilities.

The route for Opt=Restart is basically just the main loop from the original optimization, with the special lines for the first step omitted. The second invocation of Link 103 is kept and does the actual restarting.

A.5 RWF Numbers

The following is a list of read-write files. Those that are permanently on the checkpoint file are marked with the letter **P**, and those that are temporarily on the checkpoint file are marked with the letter **T**. **T** files are saved for use in restarting an optimization or numerical frequency run, but are deleted when the job step completes successfully.

Type	**RWF**	**Description**
P	501	Gen array.
P	502	/LABEL/–Title and atomic orbital labels.
	503	Connectivity information (MxBond,0),NBond(NAtoms),IBond(MxBond,NAtoms),RBond(MxBond, NAtoms), where arrays are rounded to a multiple of IntPWP.
	504	Dipole derivative matrices (NTT,3,NAt3).
P	505	Array of copies of /Gen/ from potential surface scan.
P	506	Saved basis set information before massage, uncontraction, etc.
P	507	ZMAT/ and /ZSUBST/.
P	508	/IBF/ Integral Bugger Format.
	509	Incomplete integral buffer.
T	510	/FPINFO/ Fletcher-Powell optimization program data.
P	511	/GRDNT/ energy, First and second derivatives over variables, NVAR.
P	512	Pseudo-potential information.
P	513	/DIBF/ integral derivative buffer format.
	514	Overlap matrix, optionally followed by absolute overlap and absolute overlap over primitives.
	515	Core-Hamiltonian. There are four matrices here: H(α), the α core Hamiltonian; H(β), the β core Hamiltonian; G'(α), the α G' contribution to Fock matrix; G'(β), the β G' contribution to Fock matrix. H(α) and H(β) differ only if Fermi contact integrals have been added. The G' matrices are for perturbations which are really quadratic in the density (and hence have a factor of 1/2 in their contribution to the energy as compared to the true one-electron terms) but which are computed externally to the SCF.

	516	Kinetic energy and modifications to the α and β core Hamiltonian. These include ECP terms, Douglas-Kroll-Hess corrections, multipole perturbations and Fermi contact perturbations. The latter are used for calculations in which the nuclear and electronic Coulomb terms are computed together, such as the Harris functional and PBC calculations. For semi-empirical, holds the core Hamiltonian without nuclear attraction terms for use in the initial guess.
	517	Fermi contact integrals.
	518	Multipole integrals, in the order X,Y,Z,XX,YY,ZZ,XY,XZ,YZ,XXX,YYY,ZZZ,XYY,XXY,XXZ, XZZ,YZZ,YYZ,XYZ,XXXX,YYYY,ZZZZ, XXXY,XXXZ,YYYX,YYYZ,ZZZX,ZZZY,XXYY, XXZZ,YYZZ,XXYZ,YYXZ,ZZXY.
T	519	Common /OptEn/–optimization control for link 109.
T	520	Electronic state: count and packed string (1+9 integers).
P	521	Electronic state: count and packed string (1+9 integers).
P	522	Eigenvalues, alpha and if necessary, beta.
	523	Symmetry assignments.
P	524	MO coefficients, real alpha.
P	525	(no longer used)
P	526	MO coefficients, real beta.
P	527	(no longer used)
T	528	SCF density matrix, real alpha.
T	529	(no longer used)
T	530	SCF density matrix, real beta.
T	531	(no longer used)
T	532	SCF density matrix, real total.
T	533	(no longer used)
T	534	SCF density matrix, real spin.
	535	(no longer used)
	536	Fock matrix, real alpha.
	537	Fock matrix, imaginary alpha.
	538	Fock matrix, real beta.
	539	Fock matrix, imaginary beta.
	540	Molecular alpha-beta overlap (U), real.
	541	Molecular alpha-beta overlap (U), imaginary.
T	542	Pseudo-potential information.
T	543	Pseudo-potential information.
T	544	Pseudo-potential information.
P	545	/ORB/ – window information.
	546	Bucket entry points.
	547	Eigenvalues (double precision with window: always alpha and beta, even in RHF case).
P	548	MO coefficients (double precision with window, alpha and if necessary beta). Complex if necessary.
	549	Molecular orbital alpha-beta overlap, double precision with window.

T	550	Potential surface scan common block.
T	551	Symmetry operaiton info (permutations, transformation matrices, etc.)
P	552	Character strings containing the stoichiometric formula and framework group designation.
T	553	Temporary storage of common/gen/ during FP optimizations.
T	554	Alternate starting MO coefficients, from L918 to L503, real alpha. Also MO coefficients in S-1/2 basis for L509 and rotation angles from L914 to L508.
	555	Alternate starting MO coefficients, from L918 to L503, imaginary alpha.
T	556	Alternate starting MO coefficients, from L918 to L503, real beta. Also MO coefficients in S-1/2 basis for L509 and rotation angles from L914 to L508.
	557	Alternate starting MO coefficients, from L918 to L503, imaginary beta.
	558	Saved HF 2nd derivative information for G1, G2, etc.
	559	Common /MAP/.
	560	Core-Hamiltonian (a. o. basis) with 2 j – k part of deleted orbitals added in. (i.e. frozen core).
P	561	External point charges or SCIPCM informations.
P	562	Symmetry operations and character table in full point group.
T	563	Integer symmetry assignments (α).
T	564	Integer symmetry assignments (β).
T	565	Lists of symmetry equivqlent shells and basis functions.
T	566	Unused in G16.
T	567	GVB pair information (currently dimensioned for 100 paired orbitals).
P	568	Saved hamiltonian information from L504 and L506.
P	569	Saved read-in window.
P	570	Saved amplitudes (IAS1,IAS2,IAD1,IAD2,IAD3; only IAS1 and IAD2 for closed-shell).
	571	Energy weighted density matrix.
	572	Dipole-velocity integrals <Phi|Del|Phi'>, X, Y, and Z, followed by R $\times$ Del integrals (R $\times$ X, R $\times$ Y, R $\times$ Z).
	573	More SCIPCM information.
T	574	/MSINFO/ Murtaugh-Sargent program data.
T	575	/OPTGRD/ Gradient optimization program data for L103, L115, and L509.
T	576	/TESTS/ Control constants in L105.
T	577	Symmetry adapted basis function data.
T	578	A logical vector indicating which MO's are occupied.
T	579	NEQATM (NATOMS*NOP2) for symmetry.
T	580	NEQBAS (NBASIS*NOP2+NBas6D*NOp2) for symmetry.
T	581	NSABF (NBASIS*NOP2) for symmetry. Followed by matching integer character table, always (8,8).
T	582	MAPROT (3*NBASIS) for symmetry.
T	583	MAPPER (NATOMS) for symmetry.
P	584	FXYZ (3*NATOMS) cartesian forces. During PSCF gradient runs, there will be two arrays here: first the PSCF gradient, then the HF only component (needed for PSCF with HF 2nd deriv).
P	585	FFXYZ (NAT3TT) cartesian force constants (lower triangle).

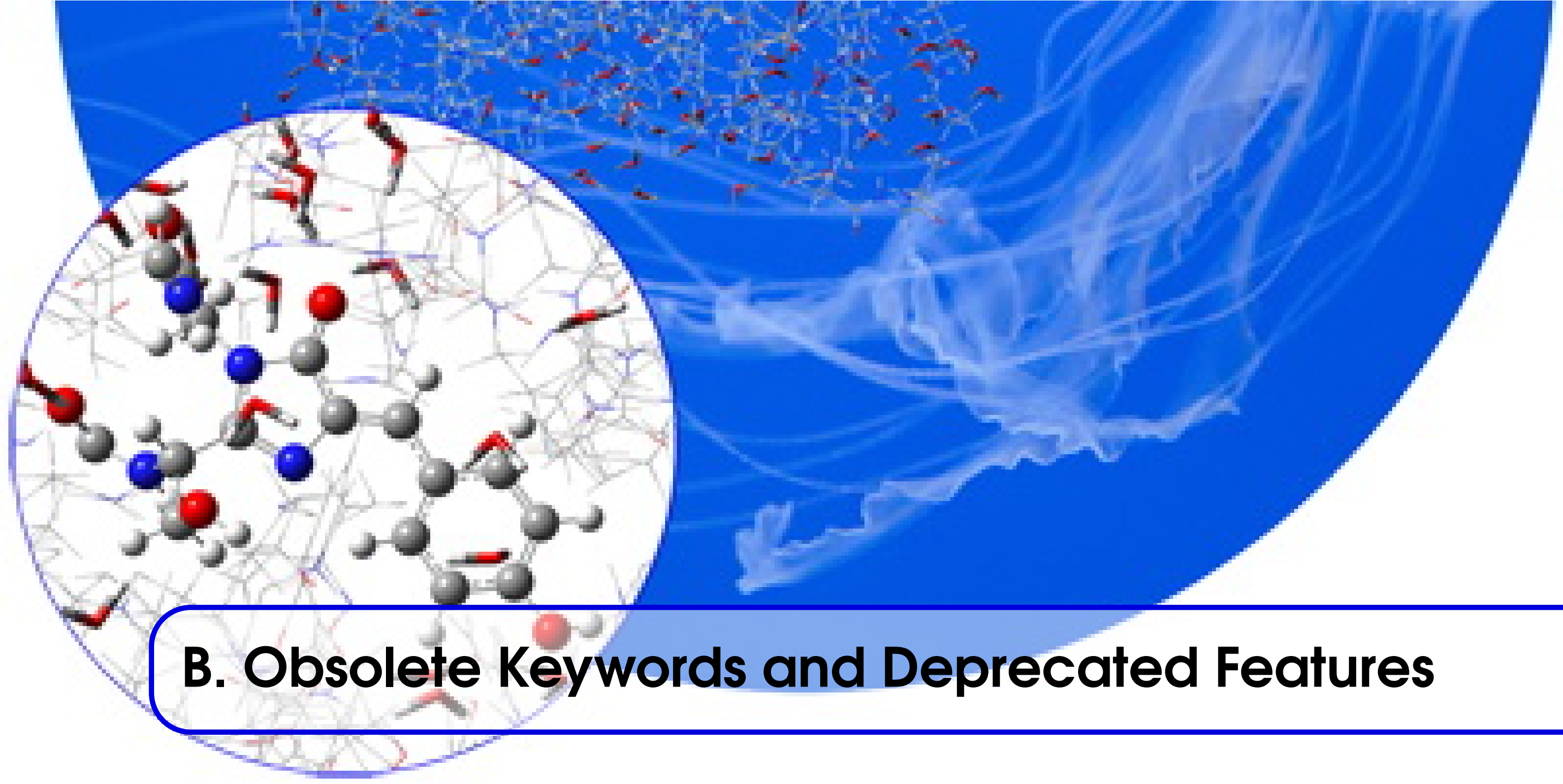
T	586	Info for L106, L110, and L111.
T	587	L107 (LST) data.
	588	Sx over cartesians in the ao basis.
	589	Hx over cartesians in the ao basis.
	590	F(x) over cartesians in the ao basis (all α, followed by all β for UHF) (without CPHF terms).
	591	U1(A,I) – MO coefficient derivatives with respect to electric field and nuclear coordinates.
	592	Electric field and nuclear P1 (AO basis).
	593	Electric field and nuclear W1 (AO basis).
	594	Electric field and nuclear S1 (MO basis).
	595	Magnetic field U1(A,I) – Del(X,Y,Z) then R $\times$ (X,Y,Z), 6 α followed by 6 β.
	596	Full MO Fock derivatives in the MO basis, including CPHF terms.
P	597	Configuration changes for Guess=Alter.
	598	User Name.
	599	Density basis set info: NDBFn, NVar, U0, DenBfn(4,NDBfn), ITypDB(NDBfn), Var(NVar), IJAnDB(NDBfn), IVar(4,NDBfn).
	600	Saved data for intra-link restart.
P	601	Saved structures, and possibly forces and force constants along reaction path. All structures, then all forces, then all force constants.
	602	Post-SCF two-particle density matrix.
P	603	Density Matrices at various levels of theory.
T	604	common /drt1/ from drt program $\cdots$ misc integer ci stuff, followed by variable dimension drt arrays.
P	605	Atomic charges from Mulliken Populations, ESP fits, etc. Bitmap followed by 0 or more NAtoms arrays. Bits 0/1/2/3/4 Mulliken/ESP-fit/Bader/NPA/APT.
	606	SCF orbital symmetries in Abelian point group. Alpha and, if necessary, beta, full set followed by windowed set.
	607	Window'd orbital symmetries like rw 606 (always alpha and beta).
	608	IBF for sorted integrals (normally on SAO unit).
	609	Bit map for sorted integrals (normally on SAO unit).
	610	Sorted AO integrals (normally on SAO unit).
	611	NTT maps for sorted integrals (normally on SAO unit).
	612	Some 1E generators for direct CI matrix element generation.
	613	Some more 1E generators for direct CI matrix element generation.
	614	Configuration information for CAS-MP2.
	615 - 616	Used for CAS-MP2.
	617	Spin-orbit integrals.
P	618	Nuclear coordinate third derivatives.
P	619	Electric field derivatives: 1 WP word bit map, dipole, dipole derivative, polarizability, dipole 2nd derivatives, polarizability derivatives, hyperpolarizability.
	620	Magnetic field derivatives for GIAOs.

	621	Susceptiblity and chemical shift tensors.
	622	Partial overlap derivatives (<Mu|dNu/da>, NBasis*NBasis*NAt3).
P	623	Born-Oppenheimer wavefunction derivatives (<Phi|d2Phi/dadb> for electronic Phi and a,b nuclear, NAt3TT).
	624	Unused in G16.
	625	Expansion vectors and AY products from CPHF, in the order Y α, AY α, Y β, AY β.
	626	MCSCF MO 1PDM (NTT).
	627	MCSCF MO Lagrangian (NTT).
	628	MCSCF MO 2PDM (NTT,NTT) or NVTTTT.
	629	AO 2PDM (shell order).
T	630	MCSCF information.
	631	Post-SCF Lagrangian (TA, then TB if UHF).
	632	O*V*3*NAtoms, followed by O*V*NVar d2E/d(V,O)d(XYZ,Atom).
P	633	Excited-state CI densities.
T	634	SCF Restart information (alpha, then possibly beta MOs).
P	635	CIS and CASSCF CI coefficients and restart information.
	636	NBO analysis information.
	637	Natural orbitals generated by link 601.
	640	MCSCF data or CIS AO Tx's for 2nd derivatives.
	641	MCSCF data for 2nd derivatives.
	642	MCSCF data for 2nd derivatives.
	643	MCSCF data for 2nd derivatives.
	644	MCSCF data for 2nd derivatives.
	645	MCSCF data for 2nd derivatives.
	646	MCSCF data for 2nd derivatives.
	647	MCSCF data for 2nd derivatives.
	648	MCSCF data for 2nd derivatives.
	649	Eigenvalue derivatives (non-canonical form even if done canonically).
	650	2PDM derivatives, (LenTQ,NDeriv,ShellQuartet) order.
	651	Full U's, canonical or non-canonical as requested.
	652	Generalized density derivatives for the current method (NTT,NDeriv,IOpCl+1).
	653	Lagrangian derivatives for the current method (NTT,NDeriv,IOpCl+1).
	654	Gx(Gamma).
	655	G(Gamma).
	656	Non-symmetric S1 and S2 parts of Lagrangian for MP2 or CIS second derivatives.
	657	t*Ix and t*Ix/D matrices from L811 for L1112.
	658	L(x) from L1111.
	659	MO correlated W for correlated frequencies.
	660	2nd order CPHF results: Pia,xy, Sxy, Fxy (complete) all in MO basis, PSF α then PSF β if UHF.
	661	Computed electric field from L602.
	662	Points for electrostatic evaluation.

T	663	Saved information for L117 and L124.
	664	Spin projection data.
P	665	Redundant coordinate information.
	666	(no longer used)
	667	CIS AO Fock matrix.
	668	CIS Gx(T) matrices.
	669	Saved /ZMat/ and /ZSubst/ during redundant optimzations.
P	670	New format basis set data (compressed /B/).
P	671	New optimization (L103/L104) data.
P	672	Unused in G16.
	673	Global optimization data.
	674	ONIOM internal data.
	675	Saved files for LS during ONIOM.
	676	Saved files for MS during ONIOM.
	677	Saved files for LM during ONIOM.
	678	Saved files for HS during ONIOM.
	679	Saved files for MM during ONIOM.
	680	Saved files for LL during ONIOM.
	681	Saved files for HM during ONIOM.
	682	Saved files for ML during ONIOM.
	683	Saved files for HL during ONIOM.
	684	SABF information for DBFS: equivalent to files 577 and 581 for AOs.
	685	Cholesky U, or transformation to surviving basis functions.
	686	Cholesky U-1.
	687	Molecular mechanics parameters.
	688	Density in orthogonal basis (α spin) for ADMP or sparse SCF.
	691	Saved initial files during ONIOM (gridpoint 17, hence 674+17=691).
	694	Permutation applied to MOs for post-SCF symmetry.
	695	Magnetic properties.
	696	Saved magnetic field density derivatives.
	698	Saved initial structure during geometry optimization, in standard orientation, also used for constraints with the force constants following the structure.
	699	Density in orthogonal basis (β spin) for ADMP or sparse SCF.
	700	Saved /Mol/ for ONIOM.
	701	Saved Trajectory/IRC/Optimization history.
	702	Fit density for Coulomb.
	703	Fit density for Coulomb.
	704	Saved XC contribution to electric field F(xa) for polar derivatives.
P	710	Basic PCM information.
P	711	Other PCM data.
P	712	Non equilibrium data for PCM.

	713	Saved information for RFO with ONIOM microiterations.
	714	Saved model system information for ONIOM microiterations.
	715	Saved rigid fragment information for ONIOM microiterations.
T	716	Saved copy of basis set data for counterpoise.
T	717	Saved copy of ECP data for counterpoise.
T	718	Saved copy of fitting basis for counterpoise.
	719	Saved DiNa information.
P	720	Saved DiNa information.
	721	Frequency-dependent properties.
	722	Derivatives of frequency-dependent properties.
	723	Density fitting matrices (metrics).
	724	Density fitting basis (same format as /B/).
	725	DBF symmetry information (NEqDBF(NDBF,NOp2),NEqDB6(NDBF6D,NOp2)).
	726	DBF shell symmetry information (NEqDBS(NDBShl,NOpAll)).
	727	F(x)(P-Pfit) for density fitting second derivatives.
	728	PBC cell replication information.
	729	Alternate new guess during optimizations.
	730	Counterpoise input specification.
	731	Counterpoise intermediate data.
	732	Basis set for finite nuclei.
	733	PBC Cell scalars and integer cell indices.
	734	State-specific input parameters for SAC-CI.
	735	Excitation lables of SAC and SAC-CI.
	736	Eigenvalues and eigenvectors of SAC and SAC-CI.
	737	H matrices and their indices of non-zero elements used for SAC/SAC-CI.
	738	Saved atomic parameters for DFTB/EHTSC.
	739	Temporary storage for imaginary core Hamiltonian perturbations.
	740	Orbital information for SAC/SAC-CI gradients and PES by GSUM.
	741	MOD Orbital information for SAC gradients.
	742	Saved quadrature grid.
	743	Alpha Fock matrices in orthonormal basis for ADMP, also alpha HF Fock matrix for non-HF post-SCF.
	744	Beta Fock matrices in orthonormal basis for ADMP, also beta HF Fock matrix for non-HF post-SCF.
	745	K-integration mesh information.
	746	Eigenvalues and orbitals at all k-points.
	747	Information for external low-level calculations for ONIOM.
	748	TS vector information for ONIOM TS optimizations.
	749	Conical intersection information for ONIOM.
	750	Not used in G16.
	751	Temporary storage for SO ECP integrals.

	752	Pseudo-canonical MO Fock matrix for ROMP and ROCC.
	753	Data for FD polar derivatives.
	754	Saved PCM charge derivatives.
	755	PCM inverse matrices.
	756	Charge information for ONIOM.
	757	MO:MO embedding charge data for L924.
	758	Derivatives of embedding charges, when computed explicitly.
	759	Basis set info for density embedding.
P	760	Full set of pseudocanonical orbitals for RO.
	761	Charges from external PCM iterations (both L117 and L124).
	762	Saved weights for non-symmetric Mulliken analysis.
	763	File for FC/HT integrals.
	764	File for FC/HT integrals.
P	765	Saved normal modes.
	766	Saved QuadMac vectors (temporary).
	767	CIS coefficients reordered by symmetry.
	768	Semi-empirical parameters.
	769	Saved MOs during numerical differentiation.
P	770	Saved ground-to-excited state energies and transition moments.
	771	EOM iteration information.
	772	Symmetry operations and character table in Abelian point group.
	989	Multi-step job information (1000 reals and 2000 integers).
	990	KJob info in some implementations.
	991	Holds file names, ID's and save flags.
	992	Used for link substitution information in some implementations.
	993	COMMON /INFO/
	994	COMMON /PHYCON/
	995	COMMON /MUNIT/
	996	COMMON /IOP/
P	997	COMMON /MOL/
P	998	COMMON /ILSW/
	999	Overlay data.



B. Obsolete Keywords and Deprecated Features

B.1 Obsolete

The following table lists obsolete keywords used by previous versions of Gaussian.

Obsolete Keyword	*Replacement Keyword & Option*
Alter	Guess=Alter
BD-T	BD(T)
BeckeHalfandHalf	BHandH
Camp-King	SCF=Camp-King
CCSD-T	CCSD(T)
CubeDensity	cubegen *utility*
GridDensity	cubegen *utility*
Cube	cubegen *utility*
DIIS	SCF=Fermi
Direct	SCF=Direct (*the default*)
Formchk	formchk *utility*
Guess=Restart	SCF=Restart
MP2=Stingy *and* VeryStingy	*none* (*options are a no-op*)
NoDIIS	SCF=NoDIIS
NoExtrap	SCF=NoExtrap
NoRaff	Int=NoRaff
OldConstants	Constants=1979
Opt=AddRedundant	Opt=ModRedundant
OptCyc=*n*	Opt(MaxCyc=*n*)
PlotDensity	cubegen *utility*
Prop=Grid	cubegen *utility*

QCID	CCD
QCISD-T	QCISD(T)
QCSCF	SCF=QC
Raff	Int=NoRaff
Save	*none* (Save *is a no-op*)
SCFCon=*n*	SCF(Conver=*n*)
SCFCyc=*n*	SCF(MaxCyc=*n*)
SCFDM	SCF=QC
SCFQC	SCF=QC
SCRF=Checkpoint	Field=EChk
VShift[=*n*]	SCF(VShift[=*n*])
%NProcLinda	%LindaWorkers
-L-	-W-
%NProc	%CPU
-N-	-C-
chkmove *utility*	formchk *and* unfchk

B.2 Deprecated

NoFMM

This keyword prevents the FMM facility from being used even when it would improve performance. It was required in some circumstances when running in parallel on a cluster or LAN with Linda in Gaussian 03. The associated problems have been fixed, and it is no longer needed.

CCD+STCCD

Specifies a coupled cluster calculation using double substitutions and evaluation of the contribution of single and triple excitations through fourth order using the CCD wavefunction. It is superseded by CCSD(T).

CBS-Q, CBS-Lq

Request the CBS-Q [180] and CBS-q [178] methods (i.e., Lq for "little q"). These are superceded by CBS-QB3.

CBS-QB3O

Uses the original parametrization [110] of CBS-QB3. It is obsolete and is included for backward compatibility only.

CBS-4O

Requests the original parametrization [180] of CBS-4. It is obsolete and is included for backward compatibility only.

Geom=Coord

Indicates that the geometry specification is in Cartesian coordinates. Cartesian coordinates can be included in molecule specifications without any special options being necessary.

Geom=OldRedundant

Use the Gaussian 94 redundant internal coordinate generator.

Geom=ModLargeRedundant

Uses the minimal setup for Opt=Big. It may not be used for periodic boundary calculations.

Int=Raff

Applies only to SCF=Conventional. Raff requests that the Raffenetti format [787] for the two-electron integrals be used. NoRaff demands that the regular integral format be used. It also suppresses the use of Raffenetti integrals during direct CPHF. This affects conventional SCF and both conventional and direct frequency calculations.

Int=BWeights

Use the weighting scheme of Becke for numerical integration.

ReUse

Use an existing integral file. Both the integral file and checkpoint file must have been preserved from a previous calculation. Only allowed for single point calculations and Polar=Restart.

WriteD2E

Forces the integral derivative file to be written in HF frequency calculations. Useful only in debugging new derivative code.

LST, LSTCyc

Requests that an initial guess for a transition structure be generated using Linear Synchronous Transit [788]. The LST procedure locates a maximum along a path connecting two structures and thus provides a guess for the transition structure connecting them.

Note that an LST calculation does not actually locate a proper transition state. The LST method has been superseded by Opt=QST2.

Massage

The Massage keyword requests that the molecule specification and basis set data be modified after it is generated. This keyword is deprecated in favor of ExtraBasis, Charge, Counterpoise and other keywords.

Opt=EnOnly

Requests an optimization using a pseudo-Newton-Raphson method with a fixed Hessian and numerical differentiation of energies to produce gradients. This option requires that the Hessian be read in via ReadFC or RCFC. It can be used to locate transition structures and higher saddle points. Requires the molecule be specified as a Z-matrix. The default for energy-only methods is Opt=(EnOnly,EF).

Opt=FP

Requests the Fletcher-Powell optimization algorithm [789], which does not require analytic gradients. The maximum number of variables allowed for a Fletcher-Powell optimization is 30. Requires the molecule be specified as a Z-matrix.

Opt=Grad

Requests a gradient optimization, using the default method unless another option is specified. This is the default whenever analytic gradients are available and is invalid otherwise.

Opt=MNDOFC

Requests that the MNDO (or AM1, if possible) force constants be computed and used to start the (presumably *ab initio*) optimization. We recommend performing a PM6 Freq calculation followed by Opt=RCFC instead of this option.

Opt=MS

Specifies the Murtaugh-Sargent optimization algorithm [790]. The Murtaugh-Sargent optimization method is an obsolete alternative, and is retained in Gaussian only for backwards compatibility. The maximum number of variables allowed for a Murtaugh-Sargent optimization is 50. Requires the molecule be specified as a Z-matrix.

Opt=UnitFC

Requests that a unit matrix be used instead of the usual valence force field guess for the Hessian.

Opt=GDIIS

Specifies the use of the modified GDIIS algorithm [791–793]. The default GEDIIS algorithm is always better.

Opt=Big

Requests the optimization to be done using the fast equation solving methods [794] for the coordinate transformations and the Newton-Raphson or RFO step. This method avoids the matrix diagonalizations. Consequently, the eigenvector following methods (Opt=TS) cannot be used in conjunction with it. This option is unreliable and not recommended.

Output=PolyAtom

This requests output of an integral file in one variant of the format originated for the PolyAtom integrals program. The format produced by default is that used by the Caltech MQM programs, but the code in Link 9999 is easily modified to produce other variations on the same theme.

Output=Trans

Write an MO coefficient file in Caltech (Tran2P5) format. This is only of interest to users of the Caltech programs.

SCF=Sleazy

Requested the loose SCF convergence criteria appropriate for single points; equivalent to SCF=(Conver=4,VarInt, NoFinal,Direct). SinglePoint is a synonym for Sleazy. It is never recommended for production quality calculations.

SCF=VerySleazy

Reduced cutoffs even further; uses Int=CoarseGrid and single-point integral accuracy during iterations, followed by a single iteration with the usual single point grid (MediumGrid). Not recommended for production quality calculations.

SCRF=DPCM

Uses the polarizable dielectric model [685, 686, 688], which corresponds to the Gaussian 98 SCRF=PCM option except for some minor implementation details [700]. This model is no longer recommended for general use. The default SCRF method is IEFPCM.

SCRF=Numer

Force numerical SCRF rather than analytic. This keyword is required for multipole orders beyond Dipole. This option implies the use of spherical cavities, which are not recommended. No gradients are available for this option.

SCRF=Dipole

The options Dipole, Quadrupole, Octopole, and Hexadecapole specify the order of multipole to use in the SCRF calculation. All but Dipole require that the Numer option be specified as well.

SCRF=Cards

Begin the SCRF=Numer calculation with a previously computed reaction field read from the input stream, immediately after the line specifying the dielectric constant and radius (three free-format reals).

%SCR

Used to specify the location of the **.SCR** scratch file.

Stable=Symm

Retain symmetry restrictions. NoSymm relaxes symmetry restrictions and is the default.

Transformation=Old2PDM

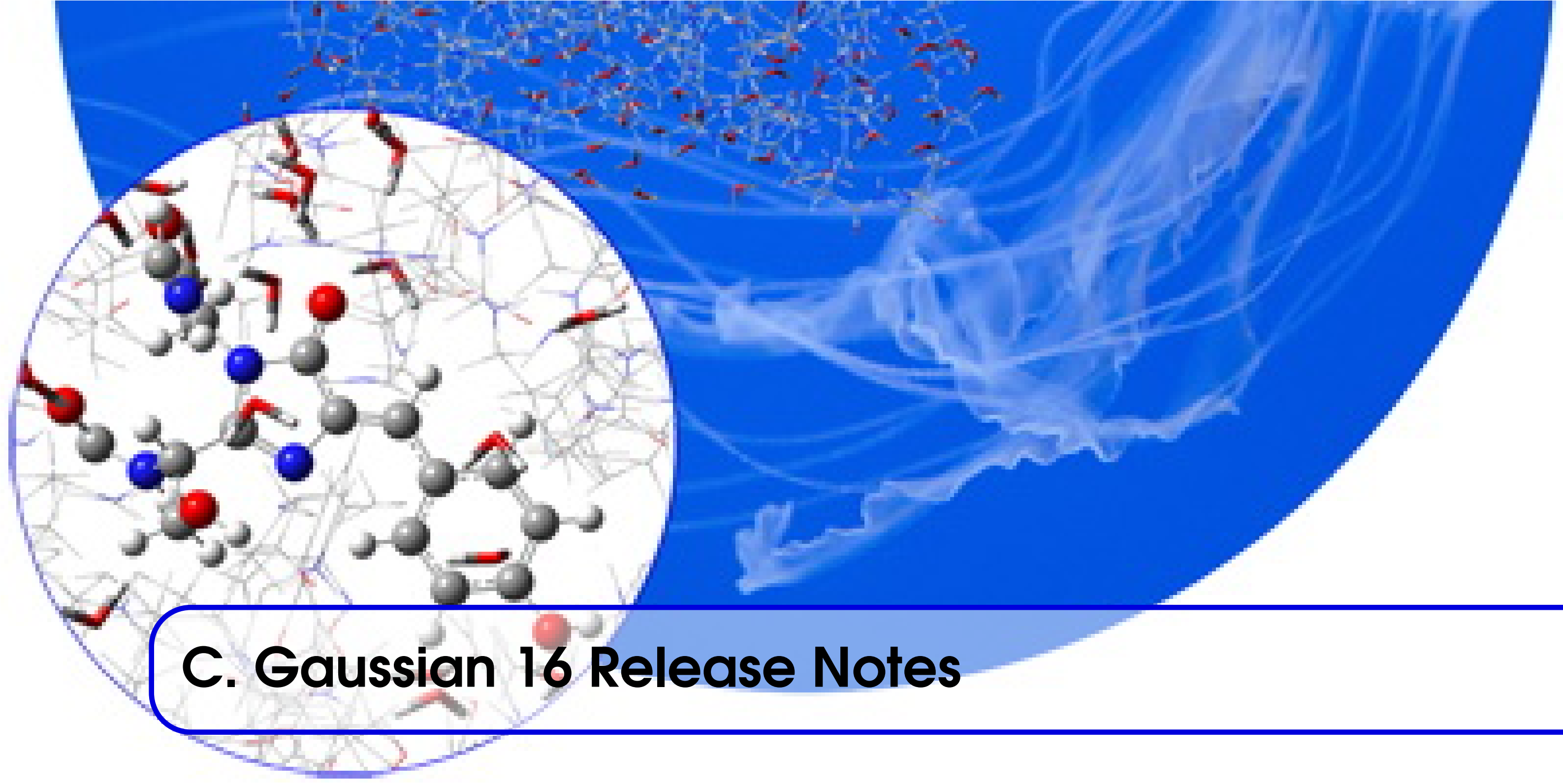
Forces the old-fashioned process of the 2PDM in post-SCF gradients (sorted in L1111 and then processed in L702 and L703). This is slow, but it reduces memory requirements. This option cannot be used for frozen core calculations.

Transformation=New2PDM

Causes the 2PDM to be generated, used, and discarded by L1111 in post-SCF gradient calculations.

Transformation=Conventional

Requests that the original transformation method based on externally stored integrals be used.



C. Gaussian 16 Release Notes

Features and changes introduced in Rev. B.01 are indicated by [**REV B**].

C.1 New Features

C.1.1 New Modeling Capabilities

- ◊ [**REV B**] Static Raman intensities can be computed for excited states at the CIS and TD levels of theory. TD Freq=Raman computes the polarizability by numerical differentiation with respect to an electric field, so the cost of Freq=Raman for these methods is 7x that of the frequencies without Raman intensities.
- ◊ TD-DFT analytic second derivatives for predicting vibrational frequencies/IR and Raman spectra and performing transition state optimizations and IRC calculations for excited states.
- ◊ EOMCC analytic gradients for performing geometry optimizations.
- ◊ Anharmonic vibrational analysis for VCD and ROA spectra: see Freq=Anharmonic.
- ◊ Vibronic spectra and intensities: see Freq=FCHT and related options.
- ◊ Resonance Raman spectra: see Freq=ReadFCHT.
- ◊ New DFT functionals: M08HX, MN15, MN15L.
- ◊ New double-hybrid methods: DSDPBEP86, PBE0DH and PBEQIDH.
- ◊ PM7 semi-empirical method.
- ◊ Ciofini excited state charge transfer diagnostic: see Pop=DCT.
- ◊ The EOMCC solvation interaction models of Caricato: see SCRF=PTED.
- ◊ Generalized internal coordinates, a facility which allows arbitrary redundant internal coordinates to be defined and used for optimization constraints and other purposes. See Geom=GIC and GIC Info.

C.1.2 Performance Enhancements

- ◊ NVIDIA K40, K80 and P100 (Pascal) GPUs are supported under Linux for Hartree-Fock and DFT calculations; P100 support is new with [**REV B**] which also provides performance improvements for all GPU types. See Using GPUs for details on GPU support and usage.

- ◊ Parallel performance on larger numbers of processors has been improved. See the Parallel Performance tab for information about how to get optimal performance on multiple CPUs and clusters.
- ◊ **[REV B]** Dynamic allocation of tasks among Linda workers is now the default, improving parallel efficiency.
- ◊ Gaussian 16 uses an optimized memory algorithm to avoid I/O during CCSD iterations.
- ◊ There are several enhancements to the GEDIIS optimization algorithm.
- ◊ CASSCF improvements for active spaces $\geq$ (10,10) increase performance and make active spaces of up to 16 orbitals feasible (depending on the molecular system).
- ◊ Significant speedup of the core correlation energies for W1 compound model.
- ◊ Gaussian 16 incorporates algorithmic improvements for significant speedup of the diagonal, second-order self-energy approximation (D2) component of composite electron propagator (CEP) methods as described in [385]. See EPT.

C.1.3 Usage Enhancements

- ◊ **[REV B]** The ChkChk utility now reports the job status (whether the job completed normally, failed, is in progress, etc.)
- ◊ **[REV B]** The optional parameters in the input line for an atom can now specify the radius to use when finite (non-point) nuclei are used. The radius is specified as a floating point value in atomic units using the RadNuclear=*val* item. For example:

```
C(RadNucl=0.001) 0.0 0.0 3.0
```

- ◊ Tools for interfacing Gaussian with other programs, both in compiled languages such as Fortran and C and with interpreted languages such as Python and Perl. Refer to *Interfacing to Gaussian 16* for details. **[REV B]** adds many additional quantities have been added to the matrix element file, including atomic populations, one-electron and property operator matrices and the non-adiabatic coupling vector. The new items are the labeled sections QUADRUPOLE INTEGRALS, OCTOPOLE INTEGRALS, HEXADECAPOLE INTEGRALS, [MULLIKEN,ESP,AIM,NPA,MBS] CHARGES, DIP VEL INTEGRALS, R X DEL INTEGRALS, OVERLAP DERIVATIVES, CORE HAMILTONIAN DERIVATIVES, F(X), DENSITY DERIVATIVES, FOCK DERIVATIVES, ALPHA UX, BETA UX, ALPHA MO DERIVATIVES, BETA MO DERIVATIVES, [Alpha,Beta] [SCF,MP2,MP3,MP4,CI Rho(1),CI,CC] DENSITY and TRANS MO COEFFICIENTS and the scalars 63-64.
- ◊ Parameters specified in Link 0 (%) input lines and/or in a *Default.Route* file can now also be specified via either command-line arguments or environment variables. **[REV B]** introduces command-line options to specify input and/or data using a checkpoint or matrix element file (the equivalent of the %OldChk or %OldMatrix Link 0 commands for input). See the *Equivalencies* tab for details.
- ◊ You can now compute the force constants at every nth step of a geometry optimization: see Opt=Recalc.
- ◊ **[REV B]** DFTB parameters are now read in Link 301 before the basis set is constructed, so that the presence or absence of d functions for an element can be taken from the parameter file.
- ◊ **[REV B]** There are now command-line options to specify input and/or data to/from checkpoint or matrix element files. See the *Equivalencies* tab or the *command line options* page for details.

C.2 Functional Changes

C.2.1 Changes from Gaussian 16 Rev. A.03

◊ There have been minor modifications to the procedure for building from source code, which is documented at `http://gaussian.com/g16/g16src_install.pdf`.

C.2.2 Changes from Gaussian 09

Calculation Defaults

The following calculation defaults are different in Gaussian 16:

◊ Integral accuracy is 10^{-12} rather than 10^{-10} in Gaussian 09.

◊ The default DFT grid for general use is UltraFine rather than FineGrid in G09; the default grid for CPHF is SG1 rather than CoarseGrid. See the discussion of the Integral keyword for details.

◊ SCRF defaults to the symmetric form of IEFPCM [702] (not present in Gaussian 09) rather than the non-symmetric version.

◊ Physical constants use the 2010 values rather than the 2006 values in Gaussian 09.

The first two items were changed to ensure accuracy in several new calculation types (e.g., TD-DFT frequencies, anharmonic ROA). For these reasons, Integral=(UltraFine,Acc2E=12) was made the default. Using these settings generally improve the reliability of calculations involving numerical integration, e.g., DFT optimizations in solution. There is a modest increase in the CPU requirements for these options compared to the Gaussian 09 defaults of Integral=(FineGrid,Acc2E=10).

The G09Defaults keyword sets all four of these defaults back to the Gaussian 09 values. It is provided for compatibility with previous calculations, but the new defaults are strongly recommended for new studies.

Default Memory Use

Gaussian 16 defaults memory usage to %Mem=100MW (800MB). Even larger values are appropriate for calculations on larger molecules and when using many processors; refer to the *Parallel Jobs* tab for details.

TD-DFT Frequencies

TDDFT frequency calculations compute second derivatives analytically by default, since these are much faster than the numerical derivatives (the only choice in Gaussian 09).

C.3 Using GPUs

See *Sec.* 4.6.

C.4 Parallel Usage and Performance Notes

C.4.1 Shared-memory parallelism

Memory allocation. Calculations involving larger molecules and basis sets benefit from larger memory allocations. 4 GB or more per processor is recommended for calculations involving 50 or more atoms and/or 500 or more basis functions. The freqmem utility estimates the optimal memory size per thread for ground-state frequency calculations, and the same value is reasonable for excited-state frequencies and is more than sufficient for ground and excited state optimizations.

The amount of memory allowed should rise with the number of processors: if 4 GB is reasonable for one processor, then the same job using 8 CPUs would run well in 32 GB. Of course, there may be limitations to smaller values imposed by the particular hardware, but scaling memory linearly with number of CPUs should be the goal. In particular, increasing only the number of CPUs with fixed memory size is unlikely to lead to good performance when using large numbers of processors.

For large frequency calculations and for large CCSD and EOM-CCSD energies, it is also desirable to leave enough memory to buffer the large disk files involved. Therefore, a Gaussian job should only be given 50-70% of the total memory on the system. For example, on a machine with a total of 128 GB, one should typically give 64-80 GB to a job which was using all the CPUs, and leave the remaining memory for the operating system to use as disk cache.

Pinning threads to CPUs under Linux. Efficiency is lost when threads are moved from one CPU to another, thereby invalidating the cache and causing other overhead. On most machines, Gaussian can tie threads to specific CPUs, and this is the recommended mode of operation, especially when using larger numbers of processors. The %CPU Link 0 line specifies the numbers of specific CPUs to be used. Thus, on a machine with one 8-core chip, one should use %CPU=07 rather than %NProc=8 because the former ties the first thread to CPU 0, the next to CPU 1, etc.

On some older Intel processors (Nehalem and before), there is not enough memory bandwidth to keep all the CPUs on a chip busy, and it is often preferable to use half the CPUs, each with twice as much memory as if all were used. For example, on such a machine with four 12-core chips and 128 GB of memory, with CPUs 0-11 on the first chip, 12-23 on the second, and so on, it is better to run using 24 processors (6 on each chip) and give them 72 GB/24 procs = 3 GB memory each, rather than use all 48 with only 1.5 GB of memory each. The required input directives would be:

```
%Mem=72GB
%CPU=0-47/2
```

where the /2 means to use every other core: i.e., cores 0, 2, 4, 6, 8, and 10 (on chip 0), 12, 14, 16, 18, 20, and 22 (on chip 1), etc.

With the most recent generations of Intel processors (Haswell and later), the memory bandwidth is better and using all the cores on each chip works well.

As long as sufficient memory is available and threads are tied to specific cores, then parallel efficiency on large molecules is good up to 64 or more cores.

Disable hyperthreading. Hyperthreading is not useful for Gaussian since it effectively divides resources such as memory bandwidth among threads on the same physical CPU. If hyperthreading cannot be turned off, Gaussian jobs should use only one hyperthread on each physical CPU. Under Linux, hyperthreads on different processors are grouped together. That is, if a machine has 2 chips each with 8 cores and 3-way hyperthreading, then "CPUs" 0-7 are across the 8 cores on chip 0, 8-15 are across the 8 cores on chip 1, and 16-23 are the second hyperthreads on the 8 cores of chip 0, and so on. So a job would run best with %CPU=015.

Under AIX, hyperthreads are grouped together with up 8 hyperthread numbers for each CPU even if fewer hyperthreads are in use, so with two 8 core chips and 4-way hyperthreading, "CPUs" 0-3 are all on core 0 of chip 0, 8-11 are on core 1 of chip 0, etc. Thus, one would want to use %CPU=0127/8 to select "CPUs" 0, 8, 16, ··· which are each using a distinct core.

C.4.2 Cluster (Linda) parallelism

Availability. Hartree-Fock and DFT energies, gradients and frequencies run in parallel across clusters, as do MP2 energies and gradients. MP2 frequencies, CCSD, and EOM-CCSD energies and optimizations are SMP parallel but not cluster parallel. Numerical derivatives, such as DFT anharmonic frequencies and CCSD frequencies, are parallelized across nodes of a cluster by doing a complete gradient or second derivative calculation on each node, splitting the directions of differentiation across workers in the cluster.

Combining with MP parallelism. Shared-memory and cluster parallelism can be combined. Generally, one uses shared-memory parallelism across all CPUs in each node of the cluster. Note that %CPU and %Mem apply to each node of the cluster. Thus, if one has 3 nodes names *apple*, *banana* and *cherry*, each with two chips which have 8 CPUs each, then one might specify:

```
%Mem=64GB
%CPU=0-15
%LindaWorkers=apple,banana,cherry
# B3LYP/6-311+G(2d,p) Freq  ···
```

This would run 16 threads, each pinned to a CPU, on each of the 3 nodes, giving 4 GB to each of the 48 threads.

For the special case of numerical differentiation only – e.g., Freq=Anharm, CCSD Freq, etc. – one extra worker is used to collect the results. So these jobs should be run with two workers on the master node (where Gaussian 16 is started). For the above example if the job was computing anharmonic frequencies, then one would use:

```
%Mem=64GB
%CPU=0-15
%LindaWorkers=apple:2,banana,cherry
# B3LYP/6-311+G(2d,p) Freq=Anharm  ···
```

where Gaussian 16 is assumed to be started on node *apple*. This will start 2 workers on node *apple*, one of which just collects results, and will do the computational work using the other worker on *apple* and those on *banana* and *cherry*.

C.5 CCSD Performance

C.5.1 Memory requirements for CCSD, CCSD(T) and EOM-CCSD calculations

These calculations can use memory to avoid I/O and will run much more efficiently if they are allowed enough memory to store the amplitudes and product vectors in memory. If there are NO active occupied orbitals (NOA in the output) and NV virtual orbitals (NVB in the output) then approximately $9NO^2NV^2$ words of memory are required. This does not depend on the number of processors used.

C.6 Equivalencies

Most options that control how Gaussian 16 operates can be specified in any of 4 ways. From highest to lowest precedence these are:

1. **As Link 0 input (%-lines)**: This is the usual method to control a specific job and the only way to control a specific step within a multi-step input file. Example: %CPU=1,2,3,4

2. **As options on the command line**: *Command line options* are useful when you want to define aliases or other shortcuts for different common ways of running the program. Example: g16 -c="1,2,3,4" ···
3. **As environment variables**: This is most useful in standard scripts, for example for generating and submitting jobs to batch queuing systems. Example: `export GAUSS_CDEF="1,2,3,4"`
4. **As directives in the Default.Route file**: This is most useful when one wants to change the program defaults for all jobs. Example: -C- 1,2,3,4

When searching for a *Default.Route* file the current default directory is checked first, followed by the directories in the path for Gaussian 16 executables: environment variable GAUSS_EXEDIR, which normally points to $g16root/g16.

The following table lists the equivalences among Link 0 commands, command line options, *Default.Route* items and environment variables. The -h, -o options and the -i and -o option classes were introduced in [**REV B**], as were their corresponding environment variables.

Default.Route	**Link 0**	**Option**	**Env. Var.**	**Description**
Gaussian 16 execution defaults				
-R-		-r	GAUSS_RDEF	Route section keyword list.
-M-	%Mem	-m	GAUSS_MDEF	Memory amount for Gaussian jobs.
-C-	%CPU	-c	GAUSS_CDEF	Processor/core list for multiprocessor parallel jobs.
-G-	%GPUCPU	-g	GAUSS_GDEF	GPUs=Cores list for GPU parallel jobs.
-S-	%UseSSH, %UseRSH	-s	GAUSS_SDEF	Program to start workers for network parallel jobs: rsh or ssh.
-W-	%LindaWorkers	-w	GAUSS_WDEF	List of hostnames for network parallel jobs.
-P-	%NProcShared	-p	GAUSS_PDEF	#processors/cores for multiprocessor parallel jobs. Deprecated; use -C-.
-L-	%NProcLinda	-l	GAUSS_LDEF	#nodes for network parallel jobs. Deprecated; use -W-.
Archive entry data				
-H-		-h	GAUSS_HDEF	Computer hostname.
-O-		-o	GAUSS_ODEF	Organization (site) name.
Utility program defaults				
-F-			GAUSS_FDEF	Options for the formchk utility.
-U-			GAUSS_UDEF	Memory amount for utilities.
Parameters for scripts and external programs				
	# *section*	-x	GAUSS_XDEF	Complete route for the job (route not read from input file).
	%Chk	-y	GAUSS_YDEF	Checkpoint file for job.
	%RWF	-z	GAUSS_ZDEF	Read-write file for job.
	%OldChk	-ic	GAUSS_ICDEF	Existing checkpoint file from which to read input.

%OldMatrix	-im	GAUSS_IMDEF	Matrix element file from which to read input.
%OldMatrix=(*file*,i4lab)	-im4	GAUSS_IM4DEF	Matrix element file using 4-byte integers from which to read input.
%OldRaw	-ir	GAUSS_IRDEF	Raw matrix element file from which to read input.
%OldRaw=(*file*,i4lab)	-im4	GAUSS_IR4DEF	Raw matrix element file using 4-byte integers from which to read input.
	-oc	GAUSS_OCDEF	Output checkpoint file. Generally redundant with -y/GAUSS_YDEF.
	-om	GAUSS_OMDEF	Output matrix element file.
	-om4	GAUSS_OM4DEF	Output matrix element file using 4-byte integers.
	-or	GAUSS_IRDEF	Output raw matrix element file.
	-or4	GAUSS_IR4DEF	Output raw matrix element file using 4-byte integers.

Note that the quotation marks are normally required around the specified value for the command line and environment variables to avoid modification of the parameter string by the shell.

C.7 Bugs Fixed

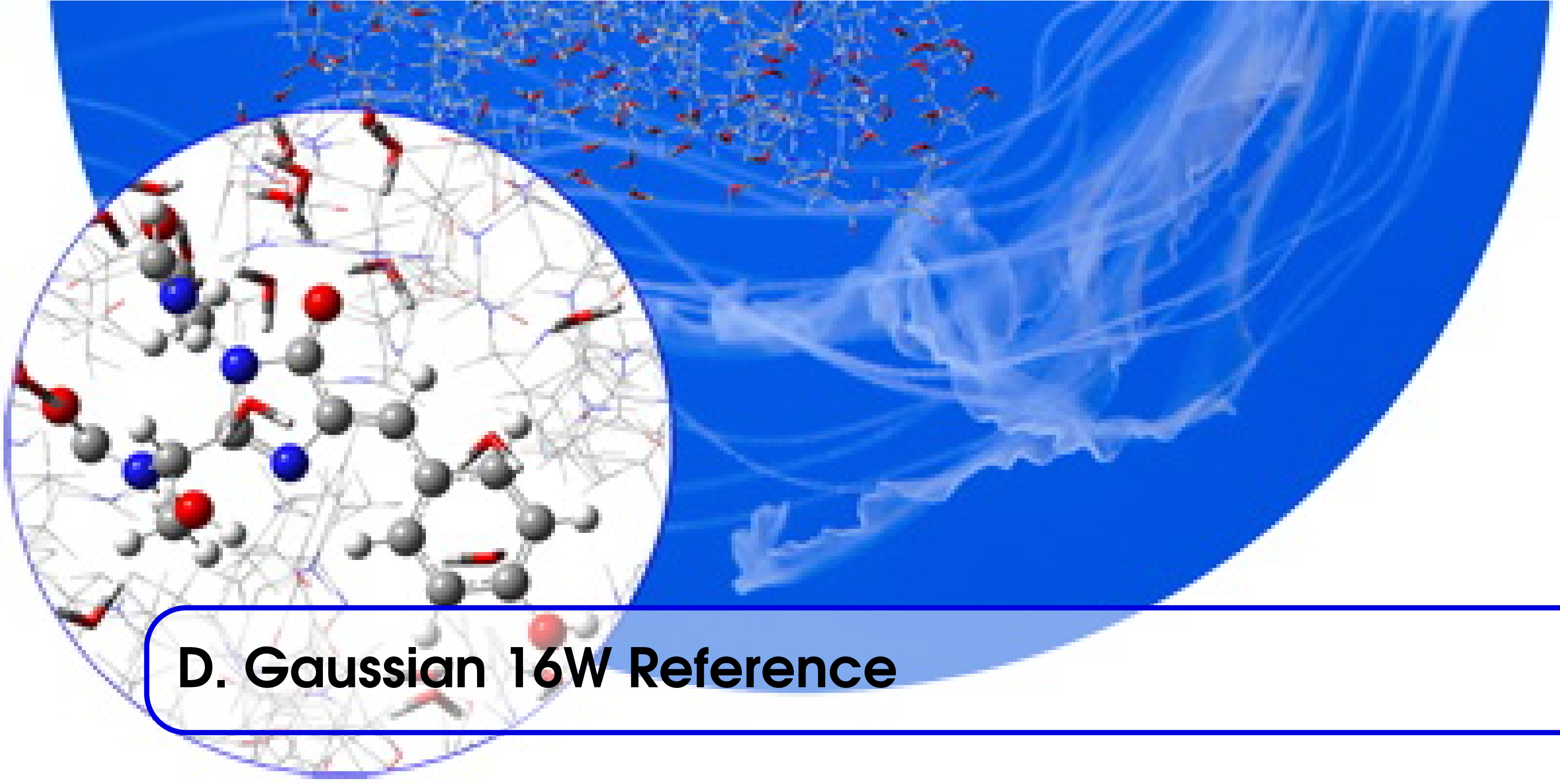
The following bugs are fixed in Rev. B.01:

- A problem with restarting in the middle of a job step (from the RWF) when using SCF=QC was fixed.
- When doing the regular SCF part of SCF=XQC or SCF=YQC, the orbitals and density are only saved when a new lowest energy wavefunction is found. If L502 fails to converge and the calculation moves on to L508 (QC or steepest descent SCF) then the best wavefunction from the regular SCF iterations is used.
- Problems with restarting from the RWF in the middle of EOM-CC calculations were fixed.
- Problems with ROMP4 and with EOM-CC when there was an empty beta spin-space or a full alpha spin-space were fixed.
- The erroneous labels in the summary table for G4 and G4MP2 jobs were corrected.
- Problems with naming scratch files for NBO when the RWF was split across physical files were fixed.
- An allocation problem which caused CIS and TD frequency jobs on large molecules using very small amounts of memory to fail was fixed. These jobs now complete, but they would run much more efficiently if given more memory (i.e., a larger value to %Mem).
- A bug which caused jobs which used the FormCheck keyword to fail was fixed. This keyword is deprecated, and the -fchk command line option, which is more flexible, is the preferred alternative.
- Unnecessary warnings which were printed by formchk when operating on a checkpoint file from a calculation which included PCM solvation were removed.
- The route for Opt=(TS,ReCalcFC=N) was corrected.
- Molecular mechanics parameters are now stored correctly in formatted checkpoint files.

◊ The route for doing interaction deletions using NBO6 (Pop=NBO6Del) was corrected.

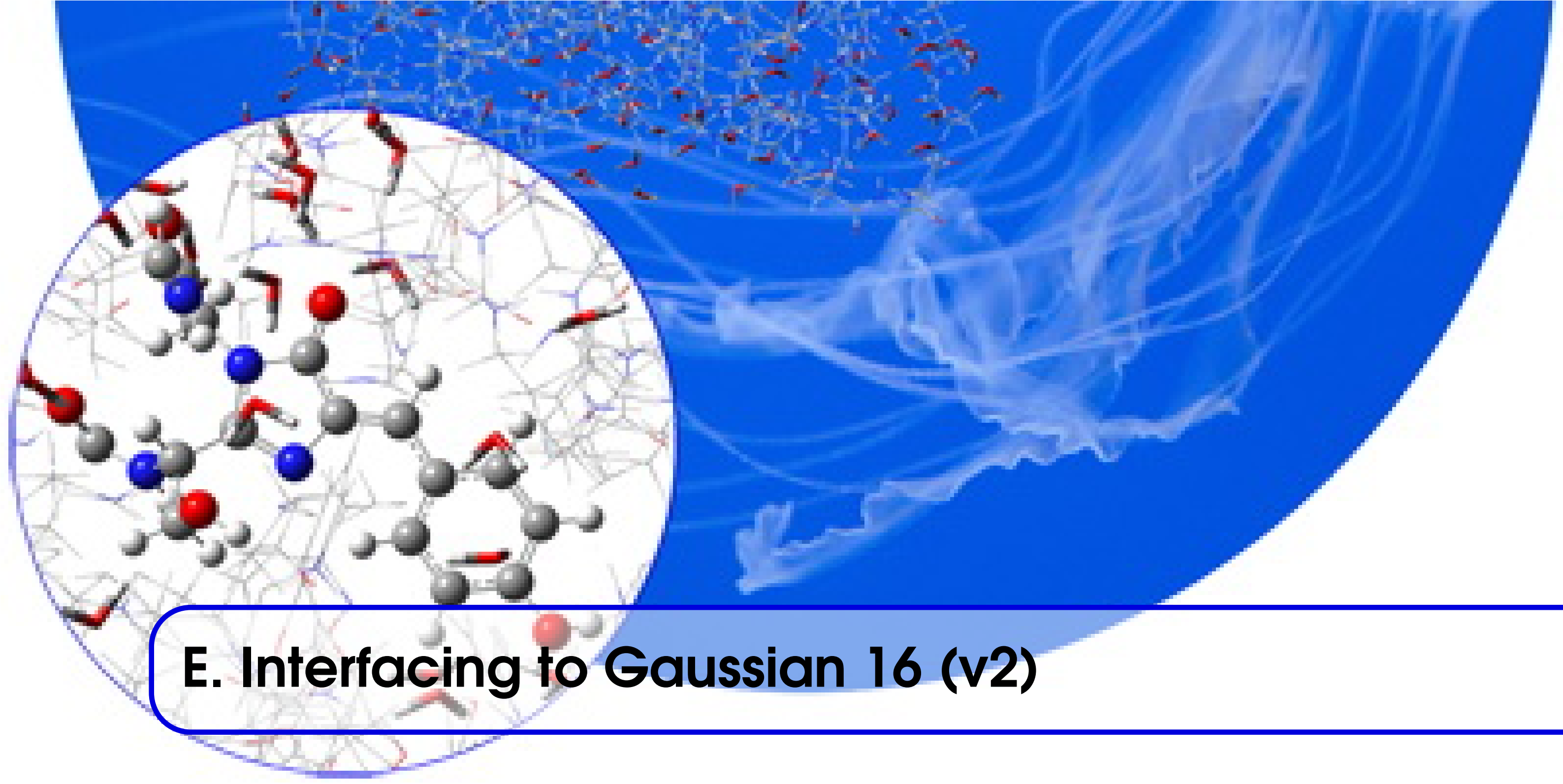
◊ A bug which prevented GPUs from being enabled in later steps of a compound job was fixed.

◊ A problem with parsing the obsolete keywords QMom and Magneton in atomic property lists was corrected.



D. Gaussian 16W Reference

Not included in this document. See `http://gaussian.com/g16w/`.



E. Interfacing to Gaussian 16 (v2)

E.1 Introduction

This documentation covers version 2 of the interfacing files. See "Release History" for information about changes from earlier versions.

There are a variety of circumstances where it is useful to run one or more calculations in Gaussian as part of a larger task, with data passed between Gaussian and the other part(s) of the task:

◊ *Scripting to automate running multiple, related Gaussian jobs.* This is typically done with the shell scripts or programs in scripting languages such as Python and Perl. Some of the tools provided here for more closely integrated computations are also useful for this purpose.

◊ *Post-processing of Gaussian calculation results.* In the past, the formatted checkpoint file (.fchk) has been the standard method to communicate results for this purpose. It is well suited to analysis tasks such as visualization, but is less suitable for numerically intensive follow-on calculations for reasons discussed below. Wavefunction (.wfn) files provide some additional information for these purposes but share some of the limitations of .fchk files.

◊ *Computations in other programs using intermediate data generated by Gaussian.* This strategy avoids the need to program features in the other program which would duplicate functionality already in Gaussian. Typical uses fall in three main categories:

  - Running production calculations combining Gaussian results with those of an independent program. These programs would typically be implemented in a compiled language such as Fortran or C. For example, SCF methods such as Valence Bond methods or unusual MCSCF methods can use one- and two-electron integrals computed by Gaussian. Similarly, post-SCF methods can use the required integrals over molecular orbitals produced by Gaussian.
  - Gaussian's infrastructure can also incorporate results from external programs. For example, geometry optimizations and ONIOM calculations can be done in Gaussian using energies and forces provided by an external program (e.g., computed using a theoretical method not available in Gaussian).

- Prototyping new models or methods, implemented in either a compiled language or an interpreted one such as Python. It can be convenient to have all the quantities computed by Gaussian available so that only the unique aspects of the new model need to be implemented and debugged.
- Classroom exercises, which created using interpreted languages such as Python or MatLab, so that the exercise can focus on a specific issue without having to provide code for unrelated parts (e.g., coding a simple SCF algorithm without having to evaluate integrals).

There are several methods available to facilitate this type of communication, with data flowing to Gaussian from anther source, from Gaussian to another program, or both. The process can be controlled with Gaussian, by a script or by another program. In general, data interchange happens via a text file known as the *matrix element* file. Its structure is designed to be very general. *Some interface routines are provided*; they are implemented with open source software and hence are useful for communication between user programs and other quantum chemistry packages as well.

The matrix element file is the recommended tool for communicating intermediate results with external programs since all quantities are included to full precision. The formatted checkpoint file remains a simpler and machine-independent option for visualization, post-processing and other cases where a moderate number of significant figures is adequate or having a plain text file is desirable.

E.2 Description

In the past, some people have used functionality from Gaussian by linking the executables for their programs with Gaussian's libraries, or by having their code called within a Gaussian link. However, this is problematic in many ways. It requires having Gaussian source code, that the user compile their own code with the same compiler and options as Gaussian, and that the user understand the internal data structures and calling conventions within Gaussian. It made the resulting code fragile in that internal changes in future versions of Gaussian, transparent to ordinary users, could break the user's code. In addition, it created unnecessary complications with respect to code ownership and intellectual property.

E.2.1 Goals for a General Interface

Our goals for a general interface are to facilitate all of the above uses in clean and maintainable ways. It should:

- ◊ Allow communication between other programs or scripts and standard Gaussian binaries without requiring recompilation, modification of Gaussian, or access to Gaussian source code.
- ◊ Be as simple for the user as possible, in particular by providing an interface which does not require knowledge of Gaussian's internal data structures or algorithms, and requiring as little knowledge of the internal structure of interface files as possible.
- ◊ Be self-defining and upward compatible, so that new types of data and new functionality can be added without breaking user code which worked with a previous version of the interface.
- ◊ Facilitate the use of Gaussian as either the controlling or the subordinate program.
- ◊ Be as non-Gaussian-centric as possible, to allow code that uses the Gaussian interface to also use other quantum chemistry codes which provide the same interface.

Previously, we have provided the formatted checkpoint file as an interface mechanism, particularly for post-processing of results from Gaussian. This file is still supported and available. It is self-defining and extensible.

Being an ordinary text file gives it a relatively simple format to process. However, while this design is suitable for post-processing results from Gaussian, especially for visualization, it is not adequate as a general interface for two reasons. First, the limited precision with which values are stored is not sufficient for intermediate data such as integrals. Second, reading and writing the file is expensive when large amounts of data are involved because of the conversion to and from text form. In addition, since the file was intended for post-processing and for moving data between Gaussian running on different types of hardware, it mirrors many of the internal Gaussian data structures.

E.2.2 Structure of the General Interface

There are three basic components of the new interface:

◊ A binary data file which is self-defining but reasonably compact, and which attempts to store standard data in simple forms rather than using Gaussian's internal data structures. These forms are supported in either Fortran unformatted form, which is useful for interfacing to programs in Fortran or Python, and as raw binary data, which is suitable for use in C, C++, and Perl programs. Data is organized in the same way in either type of file. Details of the file structure can be found in *Matrix Element File*. The formchk and unfchk utilities can convert between Gaussian checkpoint files and either type of binary interface file.

◊ Functionality within Gaussian to use such binary data files as input or output for complete Gaussian jobs and as input/output between Gaussian and external programs being run from within Gaussian. A subordinate program is connected to Gaussian via the External interface. Such a program can be used:
  - In lieu of a Gaussian SCF link.
  - As a post-SCF step using orbitals generated within Gaussian.
  - To compute derivatives for use in optimizations and frequency calculations controlled by Gaussian.
  - To perform one or more of the sub-calculations within an ONIOM model.

  Gaussian can also be run as a subordinate program, taking most or all of its input from a data file provided by the controlling program, and producing a new data file containing the results from Gaussian.

◊ Libraries and scripts which provide easy-to-use interfaces for reading and writing these file. They are open source and hence can be easily incorporated in user programs.

The interface will continue to evolve as additional capabilities are required. The quantities which are currently available in the data file include the most common quantities used in SCF and post-SCF calculations, such as the location and identities of atoms, basis set, one and two electron integrals over atomic orbitals, molecular orbital coefficients, and two electron integrals over molecular orbitals.

E.2.3 About the Interface Libraries and Scripts

The open source interface libraries and scripts have reached varying degrees of maturity depending on their extent of use so far:

◊ The Fortran interface routines (in *qcmatrix.F* and *qcmatrixio.F*) provide for convenient reading and writing data from the files with minimal knowledge of their structure. The Fortran interfaces have been tested with the PGI, Gnu and Intel Fortran compilers.

◊ The Python modules import the Fortran interface via **f2py** but add higher-level functionality, including object interfaces for both the individual items in the file (atomic coordinates, operator matrices, etc. in *QCOpMat.py*) and for the entire file as an object (in *QCMatEl.py*). In addition to reading and writing the file, the file object also provides methods to transparently run Gaussian to generate additional data, so a

Python user can request and consume results from Gaussian without any knowledge of Gaussian's input or any knowledge of the structure of the interface file on disk. The actual data from the file are stored as **NumPy** arrays and hence are useful directly in **NumPy** and **SciPy** routines. The Python interface uses many features of Python3 and is not supported with Python2.

◊ The Perl modules (*OpMat.pm* and *MatEl.pm*) provide transparent object interfaces to the data items and entire file in the same style as the Python modules, but do not include all of the higher level functionality.

Details of all the interfaces are in the comments in the source files.

The interface routines assume 4 byte integers in the data file. For the raw form of the file, a fixed integer size must be assumed because there is no reliable way to determine this when the file is read. Since both Perl and Python currently use 32-bit addressing and cannot handle arrays larger than 2 GB, this is not a limitation for these languages. Versions of the Fortran interface routines are built for both 4-byte and 8-byte integers (the latter have 8 appended to their filename).

There is not currently a separate C interface library; depending on the circumstances, C code can either read the raw binary files using the same logic as in the Perl modules, or can call the Fortran interface routines. Since the details of calling Fortran from C or C++ depend on the particular operating system and compilers, we have not attempted to standardize this.

E.2.4 Generating Interface Data Files in Gaussian

The binary matrix element file can be produced in three ways:

◊ Using the formchk utility to convert the binary checkpoint file left after a Gaussian job.

◊ Using an external script or program is to be run as part of the calculation in Gaussian via the External keyword.

◊ Using the Output=MatrixElement or Output=RawMatrixElement keywords to cause a file to be generated at the end of the current Gaussian job step.

A formatted checkpoint file can similarly be produced in three ways:

◊ Using the formchk utility to convert the binary checkpoint file left after a Gaussian job.

◊ Generating it when an external script or program is to be run as part of the calculation in Gaussian (see External).

◊ By using the **-fchk** switch on the **g16** command line, which causes a formatted checkpoint file to be written or rewritten at each "interesting" point of a job step: for a new geometry during an optimization, the end of the job step, and the like.

E.2.5 Using Data Files as Input to Gaussian

Either the formatted checkpoint file or the binary matrix element file can be used as input to Gaussian by converting it to a binary checkpoint file. This is done using the unfchk utility. You then specify the name of the generated binary checkpoint file with the %Chk or %OldChk Link 0 command to provide the file to the Gaussian job step.

In addition, a matrix element file can be specified as input using the %OldMatrixElement or %OldRawMatrixElement Link 0 commands. These take the data from the matrix element file and move it to the checkpoint file for the current job step, where it can be used to provide the geometry, initial wavefunction, etc.

E.2.6 Running Gaussian from within other programs

There are at least two ways this can be accomplished:

◊ The other program can generate an Gaussian input file. This input filspecifies that Gaussian take input from a data file and/or generate a data file containing its results. The controlling program can then run Gaussian in a subprocess using the UNIX system command, the fork/exec facility, or equivalent functionality.

◊ The Python interface (*QCMatEl.py*) provides a matrix element class which includes methods which automatically run Gaussian and update a matrix element object with additonal data. This can be used directly by a Python program, or by a program in another language can create a subprocess running a simple Python script which in turn runs Gaussian.

E.3 Matrix Element File

The matrix element file is a simple unformatted file designed to exchange data, such as the overlap and core Hamiltonian matrix and two-electron integrals, in an extensible format. It can be written either as a Fortran unformatted file, suitable for processing by other Fortran programs, or as a raw binary fine without any flags or lengths for records, suitable for processing in C/C++/Perl/Python. This file format can be used as input to and/or output from a program run via the external interface (using the External keyword), or written as file by Gaussian for separate use by another program (using the Output=MatrixElement or Output=RawMatrixElement keywords). Various options to Output in combination with one of these will select various additional data within the file (e.g., the two-electron integrals over MOs). Finally, the data within any internal Gaussian file can be included within the matrix element file via the Output=Files option.

The structure of the unformatted matrix element is as follows. The initial section contains records relating to the molecular structure and other general job characteristics:

Record Format and Description

1 `Character*64 LabFil, Integer IVers, Integer NLab, Character*64 GVers`
A label for the file type, version number for the file format, the number of general data records which precede the matrix element data (including this record), and the version of Gaussian which wrote the file.

2 `Character*64 Title, Integer NAtoms, Integer NBasis, NBsUse, ICharg, Multip, NE, Len12L, Len4L, IOpCl, ICGU`
The first 64 characters of the title section from the run which created the file, and the number of atoms and basis functions. *NBsUse* is the number of linearly independent basis functions. *ICharg* is the molecular charge, *Multip* the spin multiplicity (1=singlet, etc.), and *NE* is the number of electrons. *Len12L* is the number of bytes used for the integer labels for sparse 1D and 2D matrices, and *Len4L* is the number of bytes used for 4D matrices (2 electron integrals). *Len12L* and *Len4L* will be 4 if the file is written in raw mode (i.e., all integers will be I*4), for compatibility with other languages.

IOpCl is the closed/open-shell flag, which set if the matrix element file is written after an initial guess or the SCF has completed (otherwise it is -1, meaning unspecified). *ICGU* encodes whether the calculation is complex and/or GHF in a simpler way than *IOpCl*. Its three-digit value is interpreted as *klm*, where *k* is 1 for the spin-aligned case and 2 for GHF; *l* is 1 for real and 2 for complex; and *m* is 1 for RHF/GHF and 2 for UHF (i.e., 1 vs. 2 spin blocks). When *k*=2, then *NBasis* is the number of spatial basis functions, but the operator matrices are over the spin orbital basis and hence have dimension *k*NBasis*. When the file is read back into Gaussian, *IOpCl* can be -1 to indicate that *ICGU* should be used to specify these parameters. If *IOpCl*≥*0* and *ICGU*≥*0*, then they are checked for consistency.

3 `Integer IAn(NAtoms)`

Atomic numbers, padded to an even number of integers if using I*4.

4 `Integer IAtTyp(NAtoms)`

Atom type information. The main aspect of interest to other programs is that negative values indicate inactive atoms during an ONIOM model system calculation. By default, the file is written with inactive atoms omitted from all arrays and *NAtoms* set to the number of active atoms, but all atoms can optionally be included. It is padded to an even number of integers if using I*4.

5 `Real*8 AtmChg(NAtoms)`

Nuclear charges; may be different from atomic numbers if ECPs were used.

6 `Real*8 C(3,NAtoms)`

Cartesian nuclear coordinates in Bohr.

7 `Integer IBfAtm(NBasis), IBfTyp(NBasis)`

IBfAtm is the map from basis functions to atoms. *IBfTyp* is a type flag for each basis function. Each is of the form *lllmmm*, where *lll* is the angular momentum and *mmm* is the component number. Negative values indicate pure functions, and positive values indicate Cartesian functions. Thus, Cartesian *d* functions are numbered 2001 through 2006, and pure *d* functions are -2001 through -2005.

8 `Real*8 AtmWgt(NAtoms)`

Atomic weights.

9 `NFC, NFV, ITran, IDum`

Window information for MO 2 electron integrals. The MOs include *NFC* frozen core orbitals and *NFV* frozen virtuals, so the MO 2 electron integrals will be over *NBsUse-NFC-NFV* orbitals. *ITran*=0 if no MO integrals were stored, *ITran*=4 if only MOs involving at least one occupied orbital were stored, or *ITran*=5 if a full transformation was done.

10 to *NLab* Other scalar data about the calculation (if applicable).

Record 10 is only present when *NLab*>9. It contains *NLab*-10 integers, each of which specifies the number of 32-bit words in the corresponding initial record (allowing programs to skip them when the file is stored without record marks/lengths). E.g., if *NLab* were to be 13, record 10 will contain 3 integers, specifying the lengths of the eleventh through thirteenth initial records. Currently, *NLab*'s maximum value is 11, and record 10 contains the single integer **16**.

11 16 additional integers, of which only the first five are currently used:

NShellAO: Number of contracted shells of AO basis functions, needed if shell data is provided.

NPrimAO: Number of primitive AO shells.

NShellDB: Number of contracted shells of density fitting functions, needed if fitting shell data is provided.

NPrimDB: Number of primitive density fitting shells.

NBTot: Total Number of bonds in connectivity data, if any.

In general, programs should read (or skip) *NLab* records at the beginning of the file in order to reach the matrix element data. Doing so will ensure that additional initial records added in subsequent versions of the matrix element file are handled properly.

These *NLab* records are followed by zero or more matrix sections, each of which has an initial record:

```
Character*64 Label, Integer NI, NR, NTot, NPerRec, N1, N2, N3, N4, N5, ISym
```

where the fields contain a label, the number of integers and number of reals for each element, the total number of elements, and number of elements per record. *NI* can be zero for dense matrices (stored with zeros included). *NR* is negative to flag complex rather than real data.

N1 through *N5* are dimensions for the object as a matrix, with 0 for unused dimensions and negative values for ones which are lower triangular. For example, if *NBasis*=50 and *NBsUse*=49 then *N1* $\cdots$ *N5* would be **-50 50 0 0 0** for the overlap matrix and **50 49 0 0 0** for the transformation to an orthogonal basis. *ISym*=-1 for anti-symmetric/Hermetian matrices.

The end of the file is marked with a record whose label is **END** and having all 0 values for the integers.

This initial record is followed by *(NTot+NPerRec-1)/NPerRec* records, each of the one of the following forms:

```
Integer ID(NI,NPerRec), Real*8 DX(NR,NPerRec)
```

or just:

```
Real*8 DX(NR,NPerRec)
```

If *NI*=0 or *NR*<0, then the format is:

```
Integer ID(NI,NPerRec), Complex*16 DX(-NR,NPerRec)
```

or

```
Complex*16 DX(NR,NPerRec)
```

with the labels (if any) in *ID*, and values in *DX*. All matrix elements are over pure or Cartesian basis functions in accord with that specified in the Gaussian route or defaulted for the particular basis set.

E.3.1 Labeled Sections

The possible labels and sections include the following:

```
OVERLAP
CORE HAMILTONIAN ALPHA
CORE HAMILTONIAN BETA
KINETIC ENERGY
```

Each is a lower triangular matrix stored dense (all N*(N+1)/2 elements with no labels). The two core Hamiltonians are identical unless a Fermi contact perturbation has been applied.

```
ORTHOGONAL BASIS
```

If present, this is an (*NBasis*,*NBsUse*) matrix giving the transformation from AOs to a linearly independent orthonormal set.

`DIPOLE INTEGRALS`

If present, this includes three lower-triangular matrices holding the X, Y, and Z dipole matrix elements.

`QUADRUPOLE INTEGRALS`
`OCTOPOLE INTEGRALS`
`HEXADECAPOLE INTEGRALS`

If present, these items are the 6, 10 or 15 matrices of the Cartesian multipole integrals (respectively).

`DIP VEL INTEGRALS`
`R X DEL INTEGRALS`

If present, these are the 3 matrices of the Del and RxDel one-electron operators.

`GIAO D2H/DBDM`

If present, each is a $9 \times NAtoms$ lower triangular matrices holding the GIAO core Hamiltonian second derivatives with respect to an external field and nuclear magnetic moments (the matrix elements required for the diamagnetic shielding term).

`GIAO L/R3`

If present, this each is a $3 \times NAtoms$ lower triangular matrices holding the GIAO magnetic perturbations (the matrices required for the paramagnetic shielding term).

`ALPHA ORBITAL ENERGIES`

If present, this is an *NBsUse* vector of initial guess or converged orbital energies.

`ALPHA MO COEFFICIENTS`

If present, this is an $NBasis \times NBsUse$ matrix of initial guess or converged orbitals.

`BETA ORBITAL ENERGIES`

If present, this is an *NBsUse* vector of initial guess or converged orbital energies.

`BETA MO COEFFICIENTS`

If present, this is an $NBasis \times NBsUse$ matrix of initial guess or converged orbitals.

`ALPHA DENSITY MATRIX`

If present, this is a lower-triangular matrix containing the alpha spin part of the density matrix selected by the options for population analysis, etc. If read back into Gaussian, it is stored in place of the SCF density.

`BETA DENSITY MATRIX`

If present, this is a lower-triangular matrix containing the beta spin part of the density matrix selected by the options for population analysis, etc. If read back into Gaussian, it is stored in place of the SCF density.

`ALPHA SCF DENSITY MATRIX`

If present, this is a lower-triangular matrix containing an initial guess or converged SCF density.

`BETA SCF DENSITY MATRIX`

If present, this is a lower-triangular matrix containing an initial guess or converged SCF density.

`[ALPHA,BETA] [MP2,MP3,MP4,CI,QCI/CC] DENSITY MATRIX`

If present, these are the alpha and beta densities computed including the indicated type of correlation.

`[MULLIKEN,ESP,AIM,NPA,MBS] CHARGES`

Atomic charges of the specific type. Multiple charge sections may be present depending on the options to the Population keyword. Note that all types of electrostatic potential-derived charges are labeled as “ESP CHARGES”.

`GAUSSIAN SCALARS`

This a vector of labeled reals. Labels between 1 and 1000 refer to elements of the Gaussian /Gen/ file (for a table, see below). Labels higher than 1000 are reserved for specific scalars to/from the external program.

`ALPHA DENSITY DERIVATIVES`

`BETA DENSITY DERIVATIVES`

If present, these hold the lower triangular AO density derivatives with respect to whatever perturbations were applied during the CPHF. The number of matrices is the number of perturbations, and it varies.

`ALPHA MO DERIVATIVES`

`BETA MO DERIVATIVES`

If present, these are the (*NBasis*,*NAE*,3) and (*NBasis*,*NBE*,3) MO coefficient derivatives with respect to a magnetic field.

`ALPHA FOCK MATRIX`

If present, this is the alpha-spin Fock matrix, or the only Fock matrix for closed-shell and GHF.

`BETA FOCK MATRIX`

If present, this is the beta-spin Fock matrix for unrestricted SCF calculations.

`NUCLEAR GRADIENT`

If present, there are the 3**NAtoms* derivatives of the energy with respect to the nuclear coordinates. This is more likely to be a field returned by an external program than one used as input to the external program.

`NUCLEAR FORCE CONSTANTS`

If present, these are the second derivatives of the energy with respect to the nuclear coordinates. This is more likely to be a field returned by an external program than one used as input to the external program. They are stored as a lower triangular matrix of dimension 3**NAtoms*.

`ELECTRIC DIPOLE MOMENT`

`ELECTRIC DIPOLE DERIVATIVES`

`ELECTRIC DIPOLE POLARIZABILITY`

`DIPOLE POLARIZABILITY DERIVATIVES`

`ELECTRIC DIPOLE HYPERPOLARIZABILITY`

`ATOMIC AXIAL TENSORS`

If present, these are derivatives with respect to static external fields. If provided back to Gaussian, only the properties in the file are updated in the Gaussian internal data; any other properties already present in the Gaussian internal data are unaltered.

```
SHELL TYPE
NUMBER OF PRIMITIVES PER SHELL
CONTRACTION COEFFICIENTS
P(S=P) CONTRACTION COEFFICIENTS
COORDINATES OF EACH SHELL
```

These specify the basis set and are in the same format as the same fields in the fchk file. The same field names prefixed with `DENSITY` specify the density fitting set, if any.

```
OVERLAP DERIVATIVES
CORE HAMILTONIAN DERIVATIVES
F(X)
DENSITY DERIVATIVES
FOCK DERIVATIVES
ALPHA UX
BETA UX
```

These records contain the results of a derivative SCF (CPHF) calculation and are in the notation of [795]: Sx, Hx, F(x), Px, Fx, Ux, and Cx. F(x) and Fx contain all alpha followed by all beta Fock derivatives for open-shell, while the spin-cases of Ux and Cx are in separate records.

```
[Alpha,Beta] [SCF,MP2,MP3,MP4,CI Rho(1),CI,CC] DENSITY
```

These records hold post-SCF densities, alpha followed by beta for open-shell.

```
REGULAR 2E INTEGRALS   or
RAFFENETTI 2E INTEGRALS
```

Only non-zero integrals are stored with 4 indices $i \geq j$, $i \geq k$, $k \geq l$, $j \geq l$ if $i = k$. *NR* is 1 for regular integrals and 1, 2, or 3 for Raffenetti integral combinations:

$$R1(i,j,k,l) = (ij|kl) - 1/4[(ik|jl) + (il|jk)]$$
$$R2(i,j,k,l) = (ij|kl) + (il|jk)$$
$$R3(i,j,k,l) = (ik|jl) - (il|jk)$$

The default is *R1* only for closed-shell systems, *R1* and *R2* for UHF. The NoRaff keyword forces regular integrals, and IOp(3/11=*N*) can be used for force a particular set of Raffenetti integrals.

If MO 2 electron integrals were requested, then these are stored dense (zeroes included and no integer labels). For restricted calculations there will be one set labelled `AA MO 2E INTEGRALS` with integrals stored over the unique quartets of indices if a fill transformation (*ITran*=5 above) is done. That is, if *NROrb* = *NBsUse* - *NFC* - *NFV* is the number of active orbitals (included in the transformation) then there will *NO4*=(*NOrbTT**(*NOrbTT*+1))/2 integrals, where *NOrbTT*=(*NROrb**(*NROrb*+1))/2, when there was a full transformation. For unrestricted and a full transformation, there will be 3 sets of integrals, AA, BA, and BB. AA and BB are length *NO4* and BA is *NOrbTT*^{2}, with the beta spin indices running fastest.

For a partial transformation (*ITran*=4) there will be one set of MO integrals for restricted, dimensioned (*NOrbTT,NROrb,NOA*) where *NOA=NAE-NFC* is the number of active occupieds. For restricted there will be AA, AB, BA and BB with dimensions (*NOrbTT,NROrb,NOA*), (*NOrbTT,NROrb,NOB*), (*NOrbTT,NROrb,NOA*), and (*NOrbTT,NROrb,NOB*), respectively.

`TRANS MO COEFFICIENTS`

If MO two-electron integrals are included, this record holds the MO coefficients used in the transformation (i.e., with any frozen core or virtual orbitals omitted), alpha followed by beta spins.

E.3.2 Gaussian Scalars

1	Virial ratio
2-4	Components of applied electric field, if any
5	2e SCF energy
6	SCRF g-factor
7	SCRF a0
8	Thermal energy
9	E(CI/CC/QCI/BD)
10	E(CCD+ST4(CCD)/QCISD(T)/BD(T)/CI+Davidson)
11	E(VAR1)
12	Zero-point energy
13	Multi-step (G1, G2, etc.) energy
14	Number of imaginary frequencies
15	D(PUHF)
16	EPUHF
17	ECBS2
18	ECBSI
19	EPMP2-0
20	EPMP3-0
21	Root-mean-squared force of optimized parameters
22	E(CIS-MP2)
23	RMS error in density matrix
24	S^2 after annihilation of first contaminant
25	CIS energy
26	UMP4D (=UMP4DQ - E4(R+Q))
27	Reference energy for BD
28	MP5
29	S4SD (computed in ANNIL in L502, used by PSCF spin projection routines)
30	Frozen-core part of total energy
31	“TAU” from SCFDM
32	SCF energy
33	UMP2 energy
34	UMP3 energy

35	UMP4(SDTQ) energy
36	CBS OIii
37	Total energy with RF from L116
38	MP4DQ energy
39	MP4SDQ energy
40	Used by L116
41	Nuclear repulsion energy
42	T (length of correction of reference determinant)
43	Updated energy for optimizations
44	<S^2> of SCF wave function
45	<S^2> corrected to first order (after DOUBAR)
46	<S^2> corrected for doubles (not implemented)
47	A0
48	Used to accumulate energy during Opt=Simult
49	Temperature for thermochemistry
50	Pressure for thermochemistry
51	Scale factor for frequencies in thermochemistry
52	Nuclear repulsion contribution from inactive atom pairs
53	Singles contribution to E2 in ROMP2
54	E(2) with current orbitals for extrapolation
55	Nuclear term in the reaction field energy
56	Electronic term in the reaction field energy
57	Curvature from projected frequency jobs
58	Reaction coordinate for single-points along IRCs
59	Flag for status from external programs; see RunExt.
60	SCF energy at first iteration
61	Job status: -1=in progress; 0=undefined/old chk file; 1=finished successfully; 2=step in multi-step job completed successfully; 3=error termination in Link 9999
62	Highest order of nuclear coordinate derivatives available.
63	Number of iterations in most recent SCF.
64	Nuclear repulsion energy without external field contribution.

E.4 FChk File

This file is designed to be machine independent with a structure that makes it easy for post-processors to extract required data and ignore the remainder. The latter fact is important for extensibility as future additions will not interfere with applications designed for previous revisions. Typically a job is run specifying a .chk file, which is the binary file containing results from a calculation which are potentially useful in later calculations or for post-processing, and then after Gaussian has completed, the formchk utility is run to generate the text .fchk file from the binary .chk file. There is also a utility, unfchk, to reverse the process. For backwards compatibility, running formchk without any options produces a subset of the full information. This document describes the results of running formchk -3 *chkfile fchkfile*, which produces a version 3 formatted checkpoint file (the current and most full-featured version).

Here is a description of the data in Fortran formatted form, although there is no particular reason to use Fortran as opposed to other languages to read the data.

The first two lines in the file contain strings describing the job:

Initial 72 characters of the title section.	*Complete route and title appear later.*
Type, Method, Basis	*Format: A10,A30,A30*

Type is one of the following keywords:

SP	Single point
FOPT	Full optimization to a minimum
POPT	Partial optimization to a minimum
FTS	Full optimization to a transition state
PTS	Partial optimization to a transition state
FSADDLE	Full optimization to a saddle point of order 2 or higher
PSADDLE	Partial optimization to a saddle point of order 2 or higher
FORCE	Energy+gradient calculation
FREQ	Vibrational frequency (2nd derivative) calculation
SCAN	Potential surface scan
GUESS=ONLY	Generate molecular orbitals only, also used with localized orbital generation
LST	Linear synchronous transit
STABILITY	Test of SCF/KS stability
REARCHIVE/MS-RESTART	Generate archive information from checkpoint file
MIXED	Mixed method model chemistry (CBS-x, G1, G2, etc.), with method and basis set implied by model

Method is the method of computing the energy (AM1, RHF, CASSCF, MP4, etc.), and *Basis* is the basis set.

All other data contained in the file is located in a labeled line/section set up in one of the following forms:

◊ Scalar values appear on the same line as their data label. This line consists of a string describing the data item, a flag indicating the data type, and finally the value:
  - Integer scalars: *Name*,**I**,*IValue*, using format A40,3X,A1,5X,I12.
  - Real scalars: *Name*,**R**,*Value*, using format A40,3X,A1,5X,E22.15.
  - Character string scalars: *Name*,**C**,*Value*, using format A40,3X,A1,5X,A12.
  - Logical scalars: *Name*,**L**,*Value*, using format A40,3X,A1,5X,L1.

◊ Vector and array data sections begin with a line naming the data and giving the type and number of values, followed by the data on one or more succeeding lines (as needed):
  - Integer arrays: *Name*,I,*Num*, using format A40,3X,A1,3X,'N=',I12. The N= indicates that this is an array, and the string is followed by the number of values. The array elements then follow starting on the next line in format 6I12.
  - Real arrays: *Name*,R,*Num*, using format A40,3X,A1,3X,'N=',I12, where the N= string again in-

dicates an array and is followed by the number of elements. The elements themselves follow on succeeding lines in format 5E16.8. Note that the Real format has been chosen to ensure that at least one space is present between elements, to facilitate reading the data in C.

- Character string arrays (first type): *Name*,C,*Num*, using format A40,3X,A1,3X,'N=',I12, where the N= string indicates an array and is followed by the number of elements. The elements themselves follow on succeeding lines in format 5A12.
- Character string arrays (second type): *Name*,H,*Num*, using format A40,3X,A1,3X,'N=',I12, where the N= string indicates an array and is followed by the number of elements. The elements themselves follow on succeeding lines in format 9A8.
- Logical arrays: *Name*,L,*Num*, using format A40,3X,A1,3X,'N=',I12, where the N= string indicates an array and is followed by the number of elements. The elements themselves follow on succeeding lines in format 72L1.

All quantities are in atomic units and in the standard orientation, if that was determined by the Gaussian run. Standard orientation is seldom an interesting visual perspective, but it is the natural orientation for the vector fields. The field names are fairly verbose to make them informative and should not be an impediment as only the interface program needs to use them. An example program, demofc, is distributed with Gaussian and demonstrates how to extract a named field.

Basis Set Data

The basis set information is provided in a reasonably general way which does not assume the specific structure of Gaussian's Common /B/, which is rather obscure and reflects history more than clarity. The basis set data will include scalars giving the number of shells (*NShell*), largest degree of contraction, highest angular momentum present, and number of primitive shells (*NPrim*). There will then be arrays containing:

◊ Shell types (*NShell* values): 0=s, 1=p, -1=sp, 2=6d, -2=5d, 3=10f, -3=7f

◊ Number of primitives per shell (*NShell* values).

◊ Shell to atom map (*NShell* values): number of the atom on which each shell is located.

◊ Primitive exponents (*NPrim* values).

◊ Contraction coefficients (*NPrim* values): contraction coefficients of each normalized primitive shell. Contains the S coefficient for any S=P shells.

◊ P(S=P) Contraction coefficients (*NPrim* values): contraction coefficients for p portions of S=P shells. Not present if there are no S=P shells. Contains zeros for every primitive which is not part of an S=P shell.

◊ Coordinates of each shell: (3,*NShell*) array of XYZ coordinates for each shell.

Other data, such as basis function indexing arrays, are easily derived from the above. The order of basis functions within shells is the usual Gaussian order:

```
S,X,Y,Z,XX,YY,ZZ,XY,XZ,YZ,XXX,YYY,ZZZ,XYY,XXY,XXZ,XZZ,YZZ,YYZ,XYZ
```

or

```
3ZZ-RR,XZ,YZ,XX-YY,XY,ZZZ-ZRR,XZZ-XRR,YZZ-YRR,XXZ-YYZ,XYZ,XXX-XYY,XXY-YYY
```

Available Items

The following items are among those currently defined:

◊ Route

- ◊ Full Title
- ◊ Number of atoms
- ◊ Charge
- ◊ Multiplicity
- ◊ Number of electrons
- ◊ Number of alpha electrons
- ◊ Number of beta electrons
- ◊ Number of basis functions
- ◊ Number of contracted shells
- ◊ Highest angular momentum
- ◊ Largest degree of contraction
- ◊ Number of primitive shells
- ◊ Virial Ratio
- ◊ Atomic numbers
- ◊ Nuclear charges
- ◊ Current Cartesian coordinates
- ◊ Alpha Orbital Energies
- ◊ Beta Orbital Energies
- ◊ Alpha MO coefficients
- ◊ Beta MO coefficients
- ◊ Shell types
- ◊ Number of primitives per shell
- ◊ Shell to atom map
- ◊ Primitive exponents
- ◊ Contraction coefficients
- ◊ P(S=P) Contraction coefficients
- ◊ Coordinates of each shell
- ◊ Total SCF Density
- ◊ Spin SCF Density
- ◊ Total MP2 Density
- ◊ Spin MP2 Density
- ◊ Total CI Density
- ◊ Spin CI Density
- ◊ Total CC Density
- ◊ Spin CC Density
- ◊ Cartesian Forces
- ◊ Cartesian Force Constants
- ◊ Dipole Moment
- ◊ Dipole Derivatives
- ◊ Polarizability
- ◊ Dipole 2nd Derivatives

◊ Polarizability Derivatives

◊ HyperPolarizability

E.4.1 Formatted Checkpoint File FAQ

Which energy should be used by default?

The Total Energy field has the energy at whatever level of theory the user requested. This is so other programs don't have to figure out where the energy is from the *Method* string. In particular, we can add new methods and you won't have to change logic to find the energy you'll normally want.

Why does the field descriptor include the data type information?

The purpose of including the data type for each field is to facilitate skipping that field if it's not of interest, as illustrated in the demo program below.

How are ECP atomic charges handled?

The "Nuclear charges" will differ from the atomic numbers if ECPs are in use.

Which density matrix will be present?

The total density will always be present; the spin density will be stored only for open-shell systems. By default this will be the SCF density. If a post-SCF density is desired, include the Density keyword in the Gaussian input.

When will force constants be present?

The force constants may be present and zero for cases for which only first derivatives were actually computed, or when they were computed at the first point of a geometry optimization but not at later points. They should only be used for vibrational analysis if the job type is Freq.

Why is there no mapping array between shells and primitives?

It was pointed out that the mapping from shells to primitives is not made explicit, so that the primitive data is stored separately for every atom, even if some have the same basis set. The information that atoms have the same basis set is discarded early in Gaussian. The basis set is only of interest if the orbitals or density is also used. Since the latter are quadratic in the size of the molecule, the potential savings for large molecules from removing redundant primitives seemed modest.

E.5 Example FChk File

Here is an example formatted checkpoint file for an APFD/6-311+G(2d,p) energy calculation on HOF:

```
Example
SP        RAPFD                                      6-311+G(2d,p)
Number of atoms                            I              3
Info1-9                                    I   N=         9
           8           8           0           0           0         111
           1          18        -402
Full Title                                 C   N=         1
Example
Route                                      C   N=         4
#p apfd/6-311+g(2d,p) geom=modela test
Charge                                     I              0
Multiplicity                               I              1
Number of electrons                        I             18
Number of alpha electrons                   I               9
Number of beta electrons                    I               9
Number of basis functions                   I              60
Number of independent functions             I              60
Number of point charges in /Mol/            I               0
Number of translation vectors               I               0
Atomic numbers                              I   N=          3
           8           9           1
Nuclear charges                             R   N=          3
  8.00000000E+00  9.00000000E+00  1.00000000E+00
Current cartesian coordinates               R   N=          9
  9.50213801E-02  1.30811042E+00 -4.93038066E-32  9.50213801E-02 -1.37530068E+00
  4.93038066E-32 -1.61536346E+00  1.91282278E+00 -4.93038066E-32
Number of symbols in /Mol/                  I               0
...
```

See `http://gaussian.com/interfacing/` for the full content.

E.6 License

Gaussian Interface Code Open Source Public License, v. 1.0

Based on the Mozilla Public License version 2.0

Omitted. See `http://gaussian.com/interfacing/`.

E.7 Download

Omitted. See `http://gaussian.com/interfacing/`.

E.8 Release History

1 June 2018: Version 2.0

This version of the interface package:

◊ Fixes problems in returning arrays holding complex numbers in Python as well as a problem in writing the matrix element file from Python when the object contained 2-electron integrals.

◊ Addresses the multiple naming conventions for Python library files by setting __version__ in Python modules.

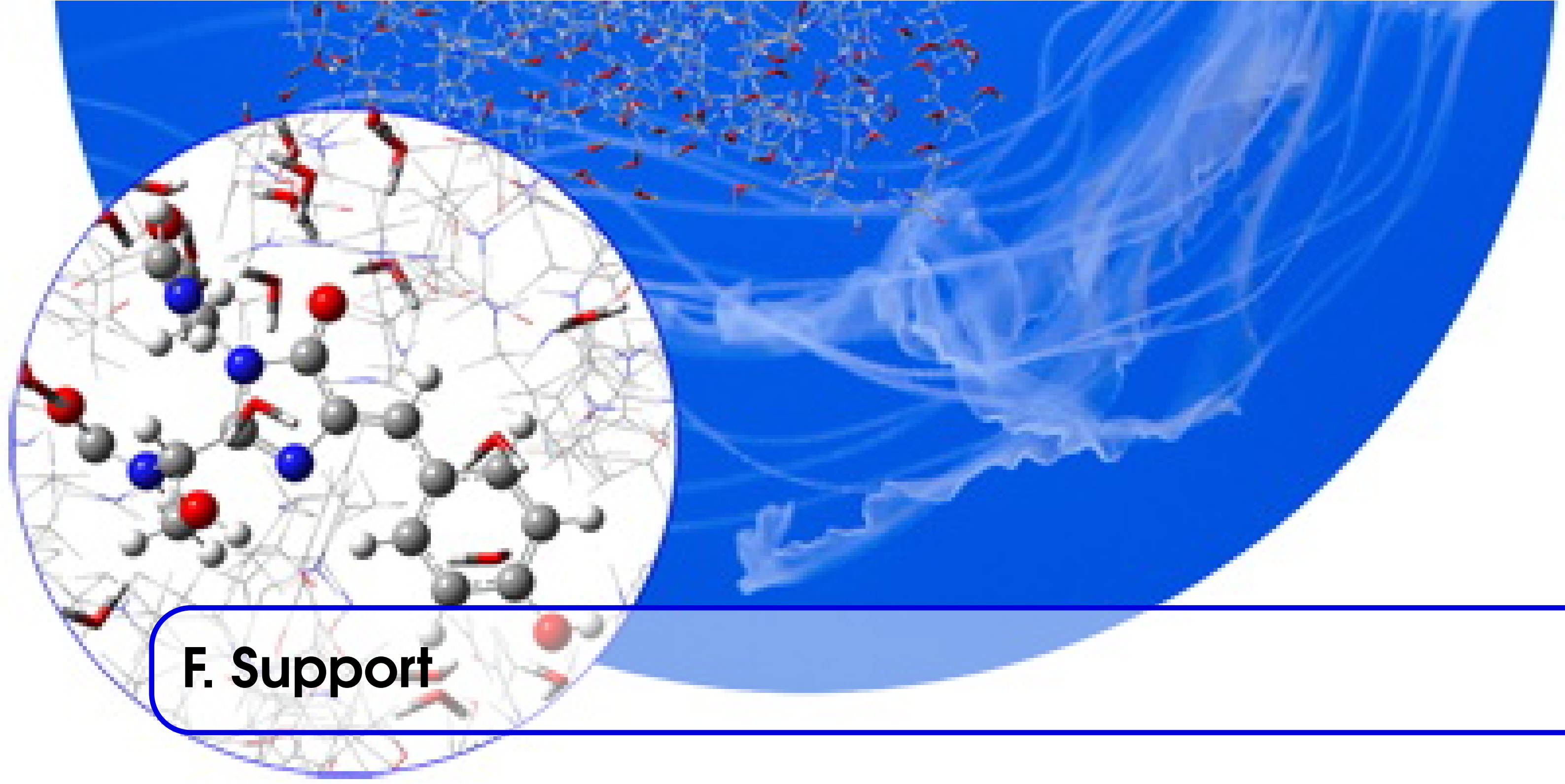
◊ Provides *precompiled shared libraries* for the Python interface for systems lacking a Fortran compiler. See the file so.list in the zip archive for information about the appropriate file names for various Python versions in common hardware+OS environments.

25 February 2018: Version 1.1 for Gaussian 16 Rev B.01

Updated for this revision of Gaussian 16. Not compatible with Gaussian Revision A.03.

5 February 2017: Version 1.0

Initial release.



F. Support

This page contains links to information about obtaining technical support from `help@gaussian.com` as well as many other resources.

◊ Contacting Gaussian, Inc. Technical Support
◊ Gaussian Technical support in Mandarin
◊ The Gaussian Maintenance Program
◊ PC/Mac Product Registration

F.1 Gaussian 16 Documentation

◊ Release Notes
◊ Keyword List
◊ G16 Users Reference
◊ G16 IOps Reference

F.2 GaussView 6 Help Documentation

◊ GaussView 6 Help

F.3 Installation Instructions

◊ Gaussian 16 source code
◊ Gaussian 16 UNIX binary
◊ Gaussian 16 Mac OS X binary
◊ GaussView 6 for UNIX
◊ GaussView 6 for Mac OS X

F.4 Linda Documentation

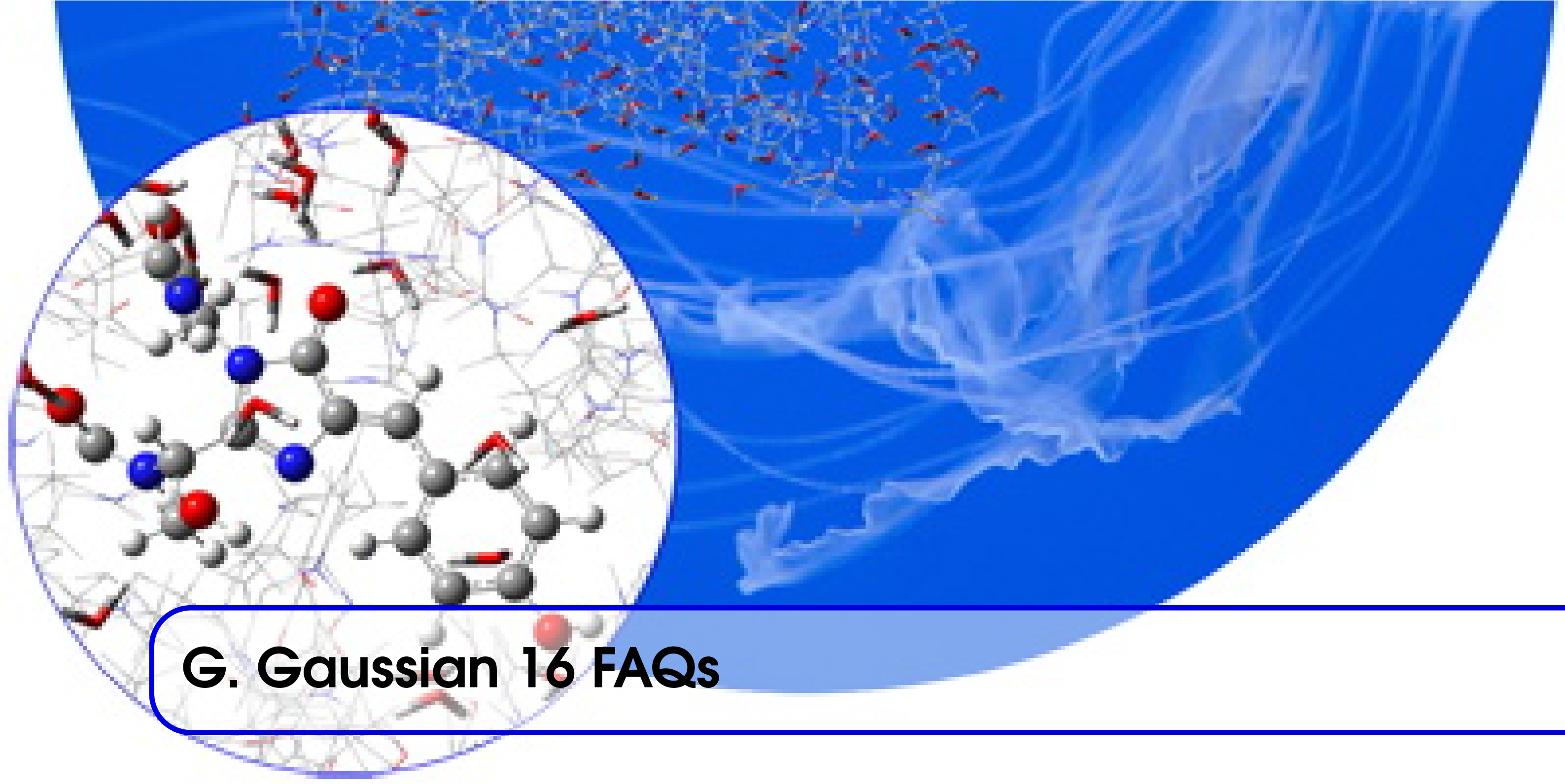
- ◊ Linda Manual (gzipped)

F.5 White Papers and Technical Notes

- ◊ Creating UV/Visible Plots from the Results of Excited States Calculations
- ◊ Modeling Antiferromagnetic Coupling in Gaussian
- ◊ Transition State Optimizations with Opt=QST2
- ◊ Using Gaussian to Teach Physical Chemistry
- ◊ Comparing NMR Methods in ChemDraw and Gaussian
- ◊ Vibrational Analysis in Gaussian
- ◊ Thermochemistry in Gaussian
- ◊ Visualizing Results when Gaussian and GaussView are on Different Machines
- ◊ Studying Chirality with Vibrational Circular Dichroism
- ◊ Investigating the Reactivity and Spectra of Large Molecules with ONIOM

F.6 Historical Documents

- ◊ G09 Citation
- ◊ Versions of Gaussian
- ◊ G09 Keyword pages (gzipped tar archive)
- ◊ GaussView 5 Reference (zipped)



G. Gaussian 16 FAQs

◊ **How can I get a breakdown of the SCF or DFT energy into all its component parts?**

See `http://gaussian.com/faq1/`.

◊ **What does the output from Link 608 mean?**

See `http://gaussian.com/faq1/`.

◊ **How can I restart a job that was interrupted?**

See `http://gaussian.com/faq2/`.

◊ **Restarting Interrupted Geometry Optimizations: Opt=Restart**

See `http://gaussian.com/faq2/`.

◊ **"Restarting" a Geometry Optimization from a Specific Point: Geom=(AllCheck,Step=*n*)**

See `http://gaussian.com/faq2/`.

◊ **Restarting IRC Calculations: IRC=Restart**

See `http://gaussian.com/faq2/`.

◊ **Restarting Analytic Frequency Calculations (and Other Long Jobs): # Restart**

See `http://gaussian.com/faq2/`.

◊ **Restarting Numerical Frequency Calculations: Freq=(Numer,Restart)**

See `http://gaussian.com/faq2/`.

◊ **Locating the Read-Write File for a Job**

See `http://gaussian.com/faq2/`.

◊ **For More Information ···**

See `http://gaussian.com/faq2/`.

◊ **I optimized a structure, then calculated the frequencies for it. The frequency calculation showed the structure was not converged even though the optimization completed. Is my structure reliable?**

See `http://gaussian.com/faq3/`.

◊ **Convergence Disagreements Between Optimizations and Frequency Calculations**

See `http://gaussian.com/faq3/`.

◊ **What to Do in Cases Like These**

See `http://gaussian.com/faq3/`.

◊ **How Much Difference Does It Make?**

See `http://gaussian.com/faq3/`.

◊ **What If the The Second Frequency Job Still Fails to Converge?**

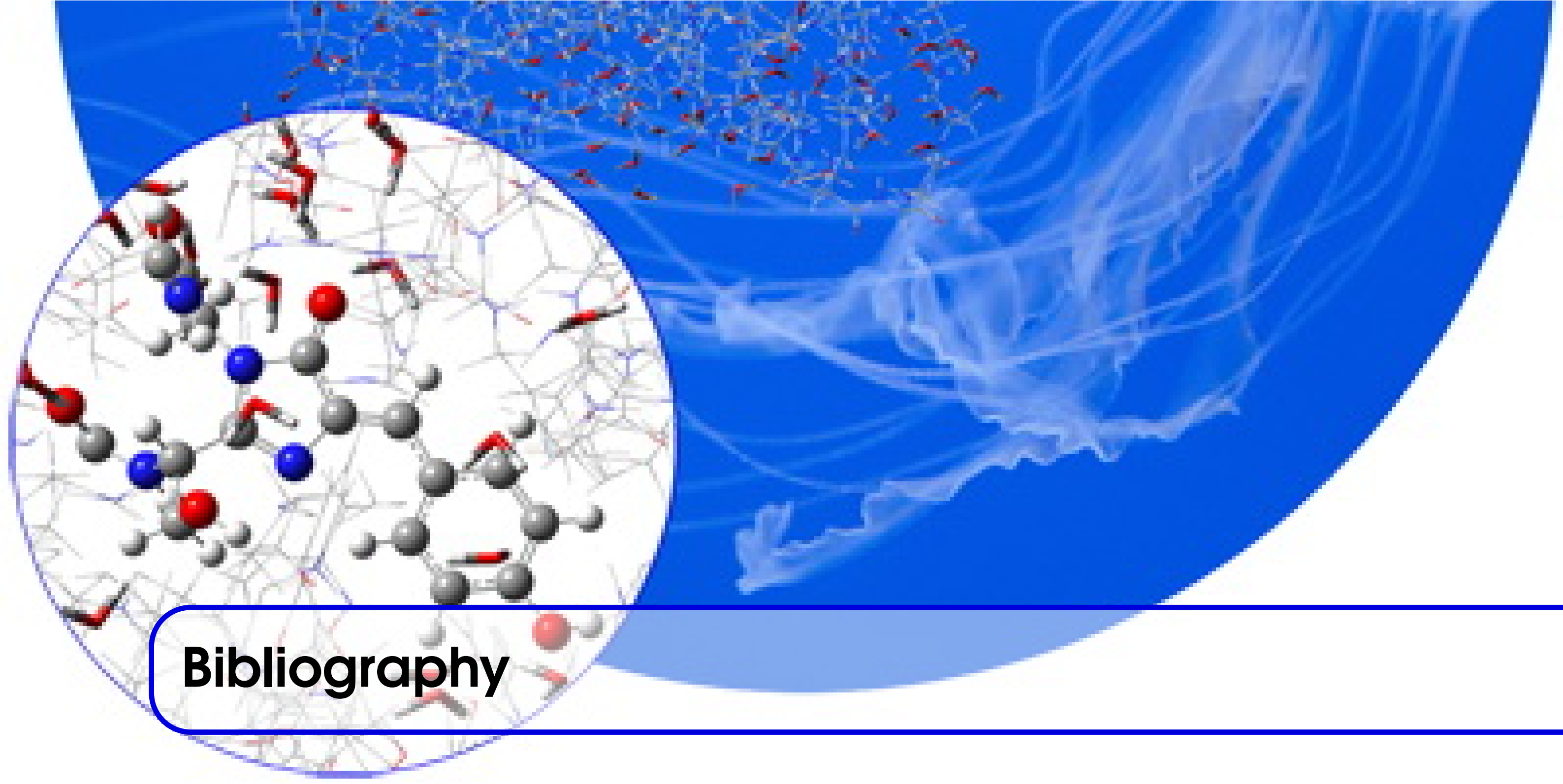
See `http://gaussian.com/faq3/`.

◊ **How do I generate Natural Transition Orbitals?**

See `http://gaussian.com/faq4/`.

◊ **Gaussian Jobs for Generating NTOs**

See `http://gaussian.com/faq4/`.



Bibliography

[1] W. J. Hehre, W. A. Lathan, R. Ditchfield, M. D. Newton, and J. A. Pople. Gaussian 70, (Quantum Chemistry Program Exchange, Program No. 237, 1970). (cited on page 3).

[2] J. S. Binkley, R. A. Whiteside, P. C. Hariharan, R. Seeger, J. A. Pople, W. J. Hehre, and M. D. Newton. Gaussian 76, (Carnegie-Mellon University, Pittsburgh, PA, 1976). (cited on page 3).

[3] J. S. Binkley, R. A. Whiteside, R. Krishnan, R. Seeger, D. J. Defrees, H. B. Schlegel, S. Topiol, L. R. Kahn, and J. A. Pople. Gaussian 80, (Carnegie-Mellon Quantum Chemistry Publishing Unit, Pittsburgh, PA, 1980). (cited on page 3).

[4] J. S. Binkley, M. J. Frisch, D. J. Defrees, R. Krishnan, R. A. Whiteside, H. B. Schlegel, E. M. Fluder, and J. A. Pople. Gaussian 82, (Carnegie-Mellon Quantum Chemistry Publishing Unit, Pittsburgh, PA, 1982). (cited on page 3).

[5] M. J. Frisch, J. S. Binkley, H. B. Schlegel, K. Raghavachari, C. F. Melius, R. L. Martin, J. J. P. Stewart, F. W. Bobrowicz, C. M. Rohlfing, L. R. Kahn, D. J. Defrees, R. Seeger, R. A. Whiteside, D. J. Fox, E. M. Fluder, and J. A. Pople. Gaussian 86, (Gaussian, Inc., Pittsburgh, PA, 1986). (cited on page 3).

[6] M. J. Frisch, M. Head-Gordon, H. B. Schlegel, K. Raghavachari, J. S. Binkley, C. Gonzalez, D. J. Defrees, D. J. Fox, R. A. Whiteside, R. Seeger, C. F. Melius, J. Baker, L. R. Kahn, J. J. P. Stewart, E. M. Fluder, S. Topiol, and J. A. Pople. Gaussian 88, (Gaussian, Inc., Pittsburgh, PA, 1988). (cited on page 3).

[7] M. J. Frisch, M. Head-Gordon, G. W. Trucks, J. B. Foresman, K. Raghavachari, H. B. Schlegel, M. Robb, J. S. Binkley, C. Gonzalez, D. J. Defrees, D. J. Fox, R. A. Whiteside, R. Seeger, C. F. Melius, J. Baker, L. R. Kahn, J. J. P. Stewart, E. M. Fluder, S. Topiol, and J. A. Pople. Gaussian 90, (Gaussian, Inc., Pittsburgh, PA, 1990). (cited on page 3).

[8] M. J. Frisch, G. W. Trucks, M. Head-Gordon, P. M. W. Gill, M. W. Wong, J. B. Foresman, B. G. Johnson, H. B. Schlegel, M. A. Robb, E. S. Replogle, R. Gomperts, J. L. Andres, K. Raghavachari, J. S. Binkley, C. Gonzalez, R. L. Martin, D. J. Fox, D. J. Defrees, J. Baker, J. J. P. Stewart, and J. A. Pople. Gaussian 92, (Gaussian, Inc., Pittsburgh, PA, 1992). (cited on page 3).

[9] M. J. Frisch, G. W. Trucks, H. B. Schlegel, P. M. W. Gill, B. G. Johnson, M. W. Wong, J. B. Foresman, M. A. Robb, M. Head-Gordon, E. S. Replogle, R. Gomperts, J. L. Andres, K. Raghavachari, J. S. Binkley, C. Gonzalez, R. L. Martin, D. J. Fox, D. J. Defrees, J. Baker, J. J. P. Stewart, and J. A. Pople. Gaussian 92/DFT, (Gaussian, Inc., Pittsburgh, PA, 1993). (cited on page 3).

[10] M. J. Frisch, G. W. Trucks, H. B. Schlegel, P. M. W. Gill, B. G. Johnson, M. A. Robb, J. R. Cheeseman, T. A. Keith, G. A. Petersson, J. A. Montgomery Jr., K. Raghavachari, M. A. Al-Laham, V. G. Zakrzewski, J. V. Ortiz, J. B. Foresman, J. Cioslowski, B. B. Stefanov, A. Nanayakkara, M. Challacombe, C. Y. Peng, P. Y. Ayala, W. Chen, M. W. Wong, J. L. Andres, E. S. Replogle, R. Gomperts, R. L. Martin, D. J. Fox, J. S. Binkley, D. J. Defrees, J. Baker, J. P. Stewart, M. Head-Gordon, C. Gonzalez, and J. A. Pople. Gaussian 94, (Gaussian, Inc., Pittsburgh, PA, 1995). (cited on page 3).

[11] M. J. Frisch, G. W. Trucks, H. B. Schlegel, G. E. Scuseria, M. A. Robb, J. R. Cheeseman, V. G. Zakrzewski, J. A. Montgomery Jr., R. E. Stratmann, J. C. Burant, S. Dapprich, J. M. Millam, A. D. Daniels, K. N. Kudin, M. C. Strain, O. Farkas, J. Tomasi, V. Barone, M. Cossi, R. Cammi, B. Mennucci, C. Pomelli, C. Adamo, S. Clifford, J. Ochterski, G. A. Petersson, P. Y. Ayala, Q. Cui, K. Morokuma, P. Salvador, J. J. Dannenberg, D. K. Malick, A. D. Rabuck, K. Raghavachari, J. B. Foresman, J. Cioslowski, J. V. Ortiz, A. G. Baboul, B. B. Stefanov, G. Liu, A. Liashenko, P. Piskorz, I. Komaromi, R. Gomperts, R. L. Martin, D. J. Fox, T. Keith, M. A. Al-Laham, C. Y. Peng, A. Nanayakkara, M. Challacombe, P. M. W. Gill, B. Johnson, W. Chen, M. W. Wong, J. L. Andres, C. Gonzalez, M. Head-Gordon, E. S. Replogle, and J. A. Pople. Gaussian 98, (Gaussian, Inc., Pittsburgh, PA, 1998). (cited on page 3).

[12] M. J. Frisch, G. W. Trucks, H. B. Schlegel, G. E. Scuseria, M. A. Rob, J. R. Cheeseman, J. A. Montgomery Jr., T. Vreven, K. N. Kudin, J. C. Burant, J. M. Millam, S. S. Iyengar, J. Tomasi, V. Barone, B. Mennucci, M. Cossi, G. Scalmani, N. Rega, G. A. Petersson, H. Nakatsuji, M. Hada, M. Ehara, K. Toyota, R. Fukuda, J. Hasegawa, M. Ishida, T. Nakajima, Y. Honda, O. Kitao, H. Nakai, M. Klene, X. Li, J. E. Knox, H. P. Hratchian, J. B. Cross, V. Bakken, C. Adamo, J. Jaramillo, R. Gomperts, R. E. Stratmann, O. Yazyev, A. J. Austin, R. Cammi, C. Pomelli, J. W. Ochterski, P. Y. Ayala, K. Morokuma, G. A. Voth, P. Salvador, J. J. Dannenberg, V. G. Zakrzewski, S. Dapprich, A. D. Daniels, M. C. Strain, O. Farkas, D. K. Malick, A. D. Rabuck, K. Raghavachari, J. B. Foresman, J. V. Ortiz, Q. Cui, A. G. Baboul, S. Clifford, J. Cioslowski, B. B. Stefanov, G. Liu, A. Liashenko, P. Piskorz, I. Komaromi, R. L. Martin, D. J. Fox, T. Keith, M. A. Al-Laham, C. Y. Peng, A. Nanayakkara, M. Challacombe, P. M. W. Gill, B. Johnson, W. Chen, M. W. Wong, C. Gonzalez, and J. A. Pople. Gaussian 03, (Gaussian, Inc., Wallingford CT, 2003). (cited on page 3).

[13] M. J. Frisch, G. W. Trucks, H. B. Schlegel, G. E. Scuseria, M. A. Robb, J. R. Cheeseman, G. Scalmani, V. Barone, B. Mennucci, G. A. Petersson, H. Nakatsuji, M. Caricato, X. Li, H. P. Hratchian, A. F. Izmaylov, J. Bloino, G. Zheng, J. L. Sonnenberg, M. Hada, M. Ehara, K. Toyota, R. Fukuda, J. Hasegawa, M. Ishida, T. Nakajima, Y. Honda, O. Kitao, H. Nakai, T. Vreven, J. A. Montgomery Jr., J. E. Peralta, F. Ogliaro, M. Bearpark, J. J. Heyd, E. Brothers, K. N. Kudin, V. N. Staroverov, R. Kobayashi, J. Normand, K. Raghavachari, A. Rendell, J. C. Burant, S. S. Iyengar, J. Tomasi, M. Cossi, N. Rega, J. M. Millam, M. Klene, J. E. Knox, J. B. Cross, V. Bakken, C. Adamo, J. Jaramillo, R. Gomperts, R. E. Stratmann, O. Yazyev, A. J. Austin, R. Cammi, C. Pomelli, J. W. Ochterski, R. L. Martin, K. Morokuma, V. G. Zakrzewski, G. A. Voth, P. Salvador, J. J. Dannenberg, S. Dapprich, A. D. Daniels, Ö. Farkas, J. B. Foresman, J. V. Ortiz, J. Cioslowski, and D. J. Fox. Gaussian 09, (Gaussian, Inc., Wallingford CT, 2009). (cited on page 3).

[14] J. E. Carpenter. “Extension of Lewis structure concepts to open-shell and excited-state molecular species”. PhD thesis. University of Wisconsin, Madison, WI, 1987 (cited on pages 4, 274).

[15] J. E. Carpenter and F. Weinhold. “Analysis of the geometry of the hydroxymethyl radical by the different hybrids for different spins natural bond orbital procedure”. In: *J. Mol. Struct. (Theochem)* 169 (1988), pages 41–62 (cited on pages 4, 274).

[16] J. P. Foster and F. Weinhold. “Natural hybrid orbitals”. In: *J. Am. Chem. Soc.* 102 (1980), pages 7211–18 (cited on pages 4, 274).

[17] A. E. Reed and F. Weinhold. “Natural bond orbital analysis of near-Hartree-Fock water dimer”. In: *J. Chem. Phys.* 78 (1983), pages 4066–73 (cited on pages 4, 274).

[18] A. E. Reed, R. B. Weinstock, and F. Weinhold. “Natural-population analysis”. In: *J. Chem. Phys.* 83 (1985), pages 735–46 (cited on pages 4, 274).

[19] A. E. Reed and F. Weinhold. “Natural Localized Molecular Orbitals”. In: *J. Chem. Phys.* 83 (1985), pages 1736–40 (cited on pages 4, 274).

[20] A. E. Reed, L. A. Curtiss, and F. Weinhold. “Intermolecular interactions from a natural bond orbital, donor-acceptor viewpoint”. In: *Chem. Rev.* 88 (1988), pages 899–926 (cited on pages 4, 274).

[21] F. Weinhold and J. E. Carpenter. In: *The Structure of Small Molecules and Ions*. Edited by R. Naaman and Z. Vager. Plenum, 1988, pages 227–36 (cited on pages 4, 274).

[22] P.-O. Löwdin. “Scaling Problem, Virial Theorem and Connected Relations in Quantum Mechanics”. In: *J. Mol. Spec.* 3 (1959), pages 44–66 (cited on page 8).

[23] D. E. Magnoli and J. R. Murdoch. “Obtaining self-consistent wave functions which satisfy the virial theorem”. In: *Int. J. Quant. Chem.* 22 (1982), pages 1249–62 (cited on page 8).

[24] M. Lehd and F. Jensen. “A General Procedure for Obtaining Wave Functions Obeying the Virial Theorem”. In: *J. Comp. Chem.* 12 (1991), pages 1089–96 (cited on page 8).

[25] W. Shakespeare. *Macbeth*. Volume III.iv.40-107. London, c.1606-1611 (cited on page 16).

[26] W. J. Hehre, R. F. Stewart, and J. A. Pople. “Self-Consistent Molecular Orbital Methods. 1. Use of Gaussian expansions of Slater-type atomic orbitals”. In: *J. Chem. Phys.* 51 (1969), pages 2657–64 (cited on page 18).

[27] J. B. Collins, P. v. R. Schleyer, J. S. Binkley, and J. A. Pople. “Self-Consistent Molecular Orbital Methods. 17. Geometries and binding energies of second-row molecules. A comparison of three basis sets”. In: *J. Chem. Phys.* 64 (1976), pages 5142–51 (cited on page 18).

[28] J. S. Binkley, J. A. Pople, and W. J. Hehre. “Self-Consistent Molecular Orbital Methods. 21. Small Split-Valence Basis Sets for First-Row Elements”. In: *J. Am. Chem. Soc.* 102 (1980), pages 939–47 (cited on page 18).

[29] M. S. Gordon, J. S. Binkley, J. A. Pople, W. J. Pietro, and W. J. Hehre. “Self-Consistent Molecular Orbital Methods. 22. Small Split-Valence Basis Sets for Second-Row Elements”. In: *J. Am. Chem. Soc.* 104 (1982), pages 2797–803 (cited on page 18).

[30] W. J. Pietro, M. M. Francl, W. J. Hehre, D. J. Defrees, J. A. Pople, and J. S. Binkley. “Self-Consistent Molecular Orbital Methods. 24. Supplemented small split-valence basis-sets for 2nd-row elements”. In: *J. Am. Chem. Soc.* 104 (1982), pages 5039–48 (cited on page 18).

[31] K. D. Dobbs and W. J. Hehre. “Molecular-orbital theory of the properties of inorganic and organometallic compounds. 4. Extended basis-sets for 3rd row and 4th row, main-group elements”. In: *J. Comp. Chem.* 7 (1986), pages 359–78 (cited on page 18).

[32] K. D. Dobbs and W. J. Hehre. “Molecular-orbital theory of the properties of inorganic and organometallic compounds. 5. Extended basis-sets for 1st-row transition-metals”. In: *J. Comp. Chem.* 8 (1987), pages 861–79 (cited on page 18).

[33] K. D. Dobbs and W. J. Hehre. “Molecular-orbital theory of the properties of inorganic and organometallic compounds. 6. Extended basis-sets for 2nd-row transition-metals”. In: *J. Comp. Chem.* 8 (1987), pages 880–93 (cited on page 18).

[34] R. Ditchfield, W. J. Hehre, and J. A. Pople. “Self-Consistent Molecular Orbital Methods. 9. Extended Gaussian-type basis for molecular-orbital studies of organic molecules”. In: *J. Chem. Phys.* 54 (1971), page 724 (cited on page 18).

[35] W. J. Hehre, R. Ditchfield, and J. A. Pople. “Self-Consistent Molecular Orbital Methods. 12. Further extensions of Gaussian-type basis sets for use in molecular-orbital studies of organic-molecules”. In: *J. Chem. Phys.* 56 (1972), page 2257 (cited on page 18).

[36] P. C. Hariharan and J. A. Pople. “Accuracy of AH equilibrium geometries by single determinant molecular-orbital theory”. In: *Mol. Phys.* 27 (1974), pages 209–14 (cited on page 18).

[37] M. S. Gordon. “The isomers of silacyclopropane”. In: *Chem. Phys. Lett.* 76 (1980), pages 163–68 (cited on page 18).

[38] P. C. Hariharan and J. A. Pople. “Influence of polarization functions on molecular-orbital hydrogenation energies”. In: *Theor. Chem. Acc.* 28 (1973), pages 213–22 (cited on page 18).

[39] M. M. Francl, W. J. Pietro, W. J. Hehre, J. S. Binkley, D. J. DeFrees, J. A. Pople, and M. S. Gordon. “Self-Consistent Molecular Orbital Methods. 23. A polarization-type basis set for 2nd-row elements”. In: *J. Chem. Phys.* 77 (1982), pages 3654–65 (cited on page 18).

[40] R. C. Binning Jr. and L. A. Curtiss. “Compact contracted basis-sets for 3rd-row atoms – Ga-Kr”. In: *J. Comp. Chem.* 11 (1990), pages 1206–16 (cited on page 18).

[41] J.-P. Blaudeau, M. P. McGrath, L. A. Curtiss, and L. Radom. “Extension of Gaussian-2 (G2) theory to molecules containing third-row atoms K and Ca”. In: *J. Chem. Phys.* 107 (1997), pages 5016–21 (cited on page 18).

[42] V. A. Rassolov, J. A. Pople, M. A. Ratner, and T. L. Windus. “6-31G* basis set for atoms K through Zn”. In: *J. Chem. Phys.* 109 (1998), pages 1223–29 (cited on page 18).

[43] V. A. Rassolov, M. A. Ratner, J. A. Pople, P. C. Redfern, and L. A. Curtiss. “6-31G* Basis Set for Third-Row Atoms”. In: *J. Comp. Chem.* 22 (2001), pages 976–84 (cited on page 18).

[44] G. A. Petersson, A. Bennett, T. G. Tensfeldt, M. A. Al-Laham, W. A. Shirley, and J. Mantzaris. “A complete basis set model chemistry. I. The total energies of closed-shell atoms and hydrides of the first-row atoms”. In: *J. Chem. Phys.* 89 (1988), pages 2193–218 (cited on pages 18, 76, 78).

[45] G. A. Petersson and M. A. Al-Laham. “A complete basis set model chemistry. II. Open-shell systems and the total energies of the first-row atoms”. In: *J. Chem. Phys.* 94 (1991), pages 6081–90 (cited on pages 18, 76, 78).

[46] A. D. McLean and G. S. Chandler. “Contracted Gaussian-basis sets for molecular calculations. 1. 2nd row atoms, Z=11-18”. In: *J. Chem. Phys.* 72 (1980), pages 5639–48 (cited on page 18).

[47] K. Raghavachari, J. S. Binkley, R. Seeger, and J. A. Pople. “Self-Consistent Molecular Orbital Methods. 20. Basis set for correlated wave-functions”. In: *J. Chem. Phys.* 72 (1980), pages 650–54 (cited on page 18).

[48] A. J. H. Wachters. “Gaussian basis set for molecular wavefunctions containing third-row atoms”. In: *J. Chem. Phys.* 52 (1970), page 1033 (cited on page 18).

[49] P. J. Hay. “Gaussian basis sets for molecular calculations – representation of 3D orbitals in transition-metal atoms”. In: *J. Chem. Phys.* 66 (1977), pages 4377–84 (cited on page 18).

[50] K. Raghavachari and G. W. Trucks. “Highly correlated systems: Excitation energies of first row transition metals Sc-Cu”. In: *J. Chem. Phys.* 91 (1989), pages 1062–65 (cited on page 18).

[51] M. P. McGrath and L. Radom. “Extension of Gaussian-1 (G1) theory to bromine-containing molecules”. In: *J. Chem. Phys.* 94 (1991), pages 511–16 (cited on page 18).

[52] L. A. Curtiss, M. P. McGrath, J.-P. Blaudeau, N. E. Davis, R. C. Binning Jr., and L. Radom. “Extension of Gaussian-2 theory to molecules containing third-row atoms Ga-Kr”. In: *J. Chem. Phys.* 103 (1995), pages 6104–13 (cited on page 18).

[53] T. H. Dunning Jr. and P. J. Hay. In: *Modern Theoretical Chemistry, Vol. 3*. Edited by H. F. Schaefer III. New York: Plenum, 1977, pages 1–28 (cited on page 18).

[54] A. K. Rappé, T. Smedly, and W. A. Goddard III. "The Shape and Hamiltonian Consistent (SHC) Effective Potentials". In: *J. Phys. Chem.* 85 (1981), pages 1662–66 (cited on page 18).

[55] W. J. Stevens, H. Basch, and M. Krauss. "Compact effective potentials and efficient shared-exponent basis-sets for the 1st-row and 2nd-row atoms". In: *J. Chem. Phys.* 81 (1984), pages 6026–33 (cited on page 18).

[56] W. J. Stevens, M. Krauss, H. Basch, and P. G. Jasien. "Relativistic compact effective potentials and efficient, shared-exponent basis-sets for the 3rd-row, 4th-row, and 5th-row atoms". In: *Can. J. Chem.* 70 (1992), pages 612–30 (cited on page 18).

[57] T. R. Cundari and W. J. Stevens. "Effective core potential methods for the lanthanides". In: *J. Chem. Phys.* 98 (1993), pages 5555–65 (cited on page 18).

[58] P. J. Hay and W. R. Wadt. "Ab initio effective core potentials for molecular calculations - potentials for the transition-metal atoms Sc to Hg". In: *J. Chem. Phys.* 82 (1985), pages 270–83 (cited on page 18).

[59] W. R. Wadt and P. J. Hay. "Ab initio effective core potentials for molecular calculations - potentials for main group elements Na to Bi". In: *J. Chem. Phys.* 82 (1985), pages 284–98 (cited on page 18).

[60] P. J. Hay and W. R. Wadt. "Ab initio effective core potentials for molecular calculations - potentials for K to Au including the outermost core orbitals". In: *J. Chem. Phys.* 82 (1985), pages 299–310 (cited on page 18).

[61] P. Fuentealba, H. Preuss, H. Stoll, and L. v. Szentpály. "A Proper Account of Core-polarization with Pseudopotentials - Single Valence-Electron Alkali Compounds". In: *Chem. Phys. Lett.* 89 (1982), pages 418–22 (cited on page 18).

[62] L. v. Szentpály, P. Fuentealba, H. Preuss, and H. Stoll. "Pseudopotential calculations on Rb^{+2}, Cs^{+2}, RbH^{+}, CsH^{+} and the mixed alkali dimer ions". In: *Chem. Phys. Lett.* 93 (1982), pages 555–59 (cited on page 18).

[63] P. Fuentealba, H. Stoll, L. v. Szentpály, P. Schwerdtfeger, and H. Preuss. "On the reliability of semi-empirical pseudopotentials - simulation of Hartree-Fock and Dirac-Fock results". In: *J. Phys. B* 16 (1983), pages L323–L28 (cited on page 18).

[64] H. Stoll, P. Fuentealba, P. Schwerdtfeger, J. Flad, L. v. Szentpály, and H. Preuss. "Cu and Ag as one-valence-electron atoms - CI results and quadrupole corrections of Cu2, Ag2, CuH, and AgH". In: *J. Chem. Phys.* 81 (1984), pages 2732–36 (cited on page 18).

[65] P. Fuentealba, L. v. Szentpály, H. Preuss, and H. Stoll. "Pseudopotential calculations for alkaline-earth atoms". In: *J. Phys. B* 18 (1985), pages 1287–96 (cited on page 18).

[66] U. Wedig, M. Dolg, H. Stoll, and H. Preuss. In: *Quantum Chemistry: The Challenge of Transition Metals and Coordination Chemistry*. Edited by A. Veillard. Reidel and Dordrecht, 1986, page 79 (cited on page 18).

[67] M. Dolg, U. Wedig, H. Stoll, and H. Preuss. "Energy-adjusted ab initio pseudopotentials for the first row transition elements". In: *J. Chem. Phys.* 86 (1987), pages 866–72 (cited on page 18).

[68] G. Igel-Mann, H. Stoll, and H. Preuss. "Pseudopotentials for main group elements (IIIA through VIIA)". In: *Mol. Phys.* 65 (1988), pages 1321–28 (cited on page 18).

[69] M. Dolg, H. Stoll, and H. Preuss. "Energy-adjusted ab initio pseudopotentials for the rare earth elements". In: *J. Chem. Phys.* 90 (1989), pages 1730–34 (cited on page 18).

[70] P. Schwerdtfeger, M. Dolg, W. H. E. Schwarz, G. A. Bowmaker, and P. D. W. Boyd. "Relativistic effects in gold chemistry. 1. Diatomic gold compounds". In: *J. Chem. Phys.* 91 (1989), pages 1762–74 (cited on page 18).

[71] M. Dolg, H. Stoll, A. Savin, and H. Preuss. "Energy-adjusted pseudopotentials for the rare-earth elements". In: *Theor. Chem. Acc.* 75 (1989), pages 173–94 (cited on page 18).

[72] D. Andrae, U. Haeussermann, M. Dolg, H. Stoll, and H. Preuss. “Energy-adjusted ab initio pseudopotentials for the 2nd and 3rd row transition-elements”. In: *Theor. Chem. Acc.* 77 (1990), pages 123–41 (cited on page 18).

[73] M. Dolg, P. Fulde, W. Kuechle, C.-S. Neumann, and H. Stoll. “Ground state calculations of di-pi-cyclooctatetraene cerium”. In: *J. Chem. Phys.* 94 (1991), pages 3011–17 (cited on page 18).

[74] M. Kaupp, P. v. R. Schleyer, H. Stoll, and H. Preuss. “Pseudopotential approaches to Ca, Sr, and Ba hydrides. Why are some alkaline-earth MX_2 compounds bent?” In: *J. Chem. Phys.* 94 (1991), pages 1360–66 (cited on page 18).

[75] W. Kuechle, M. Dolg, H. Stoll, and H. Preuss. “Ab initio pseudopotentials for Hg through Rn. 1. Parameter sets and atomic calculations”. In: *Mol. Phys.* 74 (1991), pages 1245–63 (cited on page 18).

[76] M. Dolg, H. Stoll, H.-J. Flad, and H. Preuss. “Ab initio pseudopotential study of Yb and YbO”. In: *J. Chem. Phys.* 97 (1992), pages 1162–73 (cited on page 18).

[77] A. Bergner, M. Dolg, W. Kuechle, H. Stoll, and H. Preuss. “Ab-initio energy-adjusted pseudopotentials for elements of groups 13-17”. In: *Mol. Phys.* 80 (1993), pages 1431–41 (cited on page 18).

[78] M. Dolg, H. Stoll, and H. Preuss. “A combination of quasi-relativistic pseudopotential and ligand-field calculations for lanthanoid compounds”. In: *Theor. Chem. Acc.* 85 (1993), pages 441–50 (cited on page 18).

[79] U. Haeussermann, M. Dolg, H. Stoll, and H. Preuss. “Accuracy of energy-adjusted quasi-relativistic ab initio pseudopotentials - all-electron and pseudopotential benchmark calculations for Hg, HgH and their cations”. In: *Mol. Phys.* 78 (1993), pages 1211–24 (cited on page 18).

[80] M. Dolg, H. Stoll, H. Preuss, and R. M. Pitzer. “Relativistic and correlation-effects for element 105 (Hahnium, Ha) - a comparative-study of M and MO (M = NB, TA, HA) using energy-adjusted ab initio pseudopotentials”. In: *J. Phys. Chem.* 97 (1993), pages 5852–59 (cited on page 18).

[81] W. Kuechle, M. Dolg, H. Stoll, and H. Preuss. “Energy-adjusted pseudopotentials for the actinides. Parameter sets and test calculations for thorium and thorium molecules”. In: *J. Chem. Phys.* 100 (1994), pages 7535–42 (cited on page 18).

[82] A. Nicklass, M. Dolg, H. Stoll, and H. Preuss. “Ab initio energy-adjusted pseudopotentials for the noble gases Ne through Xe: Calculation of atomic dipole and quadrupole polarizabilities”. In: *J. Chem. Phys.* 102 (1995), pages 8942–52 (cited on page 18).

[83] T. Leininger, A. Nicklass, H. Stoll, M. Dolg, and P. Schwerdtfeger. “The accuracy of the pseudopotential approximation. II. A comparison of various core sizes for indium pseudopotentials in calculations for spectroscopic constants of InH, InF, and InCl”. In: *J. Chem. Phys.* 105 (1996), pages 1052–59 (cited on page 18).

[84] X. Y. Cao and M. Dolg. “Valence basis sets for relativistic energy-consistent small-core lanthanide pseudopotentials”. In: *J. Chem. Phys.* 115 (2001), pages 7348–55 (cited on page 18).

[85] X. Y. Cao and M. Dolg. “Segmented contraction scheme for small-core lanthanide pseudopotential basis sets”. In: *J. Mol. Struct. (Theochem)* 581 (2002), pages 139–47 (cited on page 18).

[86] T. H. Dunning Jr. “Gaussian basis sets for use in correlated molecular calculations. I. The atoms boron through neon and hydrogen”. In: *J. Chem. Phys.* 90 (1989), pages 1007–23 (cited on page 18).

[87] R. A. Kendall, T. H. Dunning Jr., and R. J. Harrison. “Electron affinities of the first-row atoms revisited. Systematic basis sets and wave functions”. In: *J. Chem. Phys.* 96 (1992), pages 6796–806 (cited on pages 18, 20).

[88] D. E. Woon and T. H. Dunning Jr. “Gaussian-basis sets for use in correlated molecular calculations. 3. The atoms aluminum through argon”. In: *J. Chem. Phys.* 98 (1993), pages 1358–71 (cited on pages 18, 20).

[89] K. A. Peterson, D. E. Woon, and T. H. Dunning Jr. “Benchmark calculations with correlated molecular wave functions. IV. The classical barrier height of the $H + H_2 \rightarrow H_2 + H$ reaction”. In: *J. Chem. Phys.* 100 (1994), pages 7410–15 (cited on page 18).

[90] A. K. Wilson, T. van Mourik, and T. H. Dunning Jr. “Gaussian Basis Sets for use in Correlated Molecular Calculations. VI. Sextuple zeta correlation consistent basis sets for boron through neon”. In: *J. Mol. Struct. (Theochem)* 388 (1996), pages 339–49 (cited on page 18).

[91] E. R. Davidson. “Comment on “Comment on Dunning’s correlation-consistent basis sets””. In: *Chem. Phys. Lett.* 260 (1996), pages 514–18 (cited on page 18).

[92] A. Schaefer, H. Horn, and R. Ahlrichs. “Fully optimized contracted Gaussian-basis sets for atoms Li to Kr”. In: *J. Chem. Phys.* 97 (1992), pages 2571–77 (cited on page 19).

[93] A. Schaefer, C. Huber, and R. Ahlrichs. “Fully optimized contracted Gaussian-basis sets of triple zeta valence quality for atoms Li to Kr”. In: *J. Chem. Phys.* 100 (1994), pages 5829–35 (cited on page 19).

[94] F. Weigend and R. Ahlrichs. “Balanced basis sets of split valence, triple zeta valence and quadruple zeta valence quality for H to Rn: Design and assessment of accuracy”. In: *Phys. Chem. Chem. Phys.* 7 (2005), pages 3297–305 (cited on pages 19, 22).

[95] F. Weigend. “Accurate Coulomb-fitting basis sets for H to Rn”. In: *Phys. Chem. Chem. Phys.* 8 (2006), pages 1057–65 (cited on pages 19, 22).

[96] R. E. Easton, D. J. Giesen, A. Welch, C. J. Cramer, and D. G. Truhlar. “The MIDI! basis set for quantum mechanical calculations of molecular geometries and partial charges”. In: *Theor. Chem. Acc.* 93 (1996), pages 281–301 (cited on page 19).

[97] V. Barone. In: *Recent Advances in Density Functional Methods, Part I*. Edited by D. P. Chong. Singapore: World Scientific Publ. Co., 1996 (cited on pages 19, 277, 278).

[98] D. M. Silver, S. Wilson, and W. C. Nieuwpoort. “Universal basis sets and transferability of integrals”. In: *Int. J. Quantum Chem.* 14 (1978), pages 635–39 (cited on page 19).

[99] D. M. Silver and W. C. Nieuwpoort. “Universal atomic basis sets”. In: *Chem. Phys. Lett.* 57 (1978), pages 421–22 (cited on page 19).

[100] J. R. Mohallem, R. M. Dreizler, and M. Trsic. “A Griffin-Hill-Wheeler version of the Hartree-Fock equations”. In: *Int. J. Quantum Chem., Quant. Chem. Symp.* 30 (S20) (1986), pages 45–55 (cited on page 19).

[101] J. R. Mohallem and M. Trsic. “A universal Gaussian basis set for atoms Li through Ne based on a generator coordinate version of the Hartree-Fock equations”. In: *J. Chem. Phys.* 86 (1987), pages 5043–44 (cited on page 19).

[102] H. F. M. da Costa, M. Trsic, and J. R. Mohallem. “Universal Gaussian and Slater-type basis-sets for atoms He to Ar based on an integral version of the Hartree-Fock equations”. In: *Mol. Phys.* 62 (1987), pages 91–95 (cited on page 19).

[103] A. B. F. da Silva, H. F. M. da Costa, and M. Trsic. “Universal Gaussian and Slater-type bases for atoms H to Xe based on the generator-coordinate Hartree-Fock method .1. Ground and certain low-lying excited-states of the neutral atoms”. In: *Mol. Phys.* 68 (1989), pages 433–45 (cited on page 19).

[104] F. E. Jorge, E. V. R. de Castro, and A. B. F. da Silva. “A universal Gaussian basis set for atoms Cerium through Lawrencium generated with the generator coordinate Hartree-Fock method”. In: *J. Comp. Chem.* 18 (1997), pages 1565–69 (cited on page 19).

[105] F. E. Jorge, E. V. R. de Castro, and A. B. F. da Silva. “Accurate universal Gaussian basis set for hydrogen through lanthanum generated with the generator coordinate Hartree-Fock method”. In: *Chem. Phys.* 216 (1997), pages 317–21 (cited on page 19).

[106] E. V. R. de Castro and F. E. Jorge. “Accurate universal gaussian basis set for all atoms of the periodic table”. In: *J. Chem. Phys.* 108 (1998), pages 5225–29 (cited on page 19).

[107] J. M. L. Martin and G. de Oliveira. “Towards standard methods for benchmark quality ab initio thermochemistry - W1 and W2 theory”. In: *J. Chem. Phys.* 111 (1999), pages 1843–56 (cited on pages 19, 330).

[108] N. Godbout, D. R. Salahub, J. Andzelm, and E. Wimmer. "Optimization of Gaussian-type basis sets for local spin density functional calculations. Part I. Boron through neon, optimization technique and validation". In: *Can. J. Chem.* 70 (1992), pages 560–71 (cited on pages 19, 22).

[109] C. Sosa, J. Andzelm, B. C. Elkin, E. Wimmer, K. D. Dobbs, and D. A. Dixon. "A Local Density Functional Study of the Structure and Vibrational Frequencies of Molecular Transition-Metal Compounds". In: *J. Phys. Chem.* 96 (1992), pages 6630–36 (cited on pages 19, 22).

[110] J. A. Montgomery Jr., M. J. Frisch, J. W. Ochterski, and G. A. Petersson. "A complete basis set model chemistry. VI. Use of density functional geometries and frequencies". In: *J. Chem. Phys.* 110 (1999), pages 2822–27 (cited on pages 19, 76, 271, 380).

[111] T. Clark, J. Chandrasekhar, G. W. Spitznagel, and P. v. R. Schleyer. "Efficient diffuse function-augmented basis-sets for anion calculations. 3. The 3-21+G basis set for 1st-row elements, Li-F". In: *J. Comp. Chem.* 4 (1983), pages 294–301 (cited on page 20).

[112] M. J. Frisch, J. A. Pople, and J. S. Binkley. "Self-Consistent Molecular Orbital Methods. 25. Supplementary Functions for Gaussian Basis Sets". In: *J. Chem. Phys.* 80 (1984), pages 3265–69 (cited on page 20).

[113] E. Papajak, J. Zheng, H. R. Leverentz, and D. G. Truhlar. "Perspectives on Basis Sets Beautiful: Seasonal Plantings of Diffuse Basis Functions". In: *J. Chem. Theory and Comput.* 7 (2011), page 3027 (cited on page 20).

[114] H. B. Schlegel and M. J. Frisch. "Transformation between Cartesian and Pure Spherical Harmonic Gaussians". In: *Int. J. Quantum Chem.* 54 (1995), pages 83–87 (cited on pages 22, 159).

[115] B. I. Dunlap. "Fitting the Coulomb Potential Variationally in X-Alpha Molecular Calculations". In: *J. Chem. Phys.* 78 (1983), pages 3140–42 (cited on page 22).

[116] B. I. Dunlap. "Robust and variational fitting: Removing the four-center integrals from center stage in quantum chemistry". In: *J. Mol. Struct. (Theochem)* 529 (2000), pages 37–40 (cited on page 22).

[117] K. Eichkorn, O. Treutler, H. Ohm, M. Haser, and R. Ahlrichs. "Auxiliary basis-sets to approximate Coulomb potentials". In: *Chem. Phys. Lett.* 240 (1995), pages 283–89 (cited on page 22).

[118] K. Eichkorn, F. Weigend, O. Treutler, and R. Ahlrichs. "Auxiliary basis sets for main row atoms and transition metals and their use to approximate Coulomb potentials". In: *Theor. Chem. Acc.* 97 (1997), pages 119–24 (cited on page 22).

[119] J. A. Pople and M. S. Gordon. "Molecular orbital theory of electronic structure of organic compounds. 1. Substituent effects and dipole methods". In: *J. Am. Chem. Soc.* 89 (1967), page 4253 (cited on page 27).

[120] D. L. Bunker. "Classical Trajectory Methods". In: *Meth. Comp. Phys.* 10 (1971), page 287 (cited on pages 55, 61).

[121] L. M. Raff and D. L. Thompson. In: *Theory of Chemical Reaction Dynamics*. Edited by M. Baer. Boca Raton, FL: CRC, 1985 (cited on pages 55, 61).

[122] W. L. Hase, editor. *Advances in Classical Trajectory Methods*. Volume 1-3. Stamford, CT: JAI, 1991 (cited on pages 55, 61).

[123] D. L. Thompson. In: *Encyclopedia of Computational Chemistry*. Edited by P. v. R. Schleyer, N. L. Allinger, P. A. Kollman, T. Clark, H. F. Schaefer III, J. Gasteiger, and P. R. Schreiner. Chichester: Wiley, 1998, pages 3056–73 (cited on pages 55, 61).

[124] S. S. Iyengar, H. B. Schlegel, J. M. Millam, G. A. Voth, G. E. Scuseria, and M. J. Frisch. "Ab initio molecular dynamics: Propagating the density matrix with Gaussian orbitals. II. Generalizations based on mass-weighting, idempotency, energy conservation and choice of initial conditions". In: *J. Chem. Phys.* 115 (2001), pages 10291–302 (cited on page 55).

[125] H. B. Schlegel, J. M. Millam, S. S. Iyengar, G. A. Voth, G. E. Scuseria, A. D. Daniels, and M. J. Frisch. “Ab initio molecular dynamics: Propagating the density matrix with Gaussian orbitals”. In: *J. Chem. Phys.* 114 (2001), pages 9758–63 (cited on page 55).

[126] H. B. Schlegel, S. S. Iyengar, X. Li, J. M. Millam, G. A. Voth, G. E. Scuseria, and M. J. Frisch. “Ab initio molecular dynamics: Propagating the density matrix with Gaussian orbitals. III. Comparison with Born-Oppenheimer dynamics”. In: *J. Chem. Phys.* 117 (2002), pages 8694–704 (cited on page 55).

[127] R. Car and M. Parrinello. “Unified Approach for Molecular-Dynamics and Density-Functional Theory”. In: *Phys. Rev. Lett.* 55 (1985), pages 2471–74 (cited on page 55).

[128] G. Martyna, C. Cheng, and M. L. Klein. “Electronic States and Dynamic Behavior of Lixen and Csxen Clusters”. In: *J. Chem. Phys.* 95 (1991), pages 1318–36 (cited on page 55).

[129] G. Lippert, J. Hutter, and M. Parrinello. “A hybrid Gaussian and plane wave density functional scheme”. In: *Mol. Phys.* 92 (1997), pages 477–87 (cited on page 55).

[130] G. Lippert, J. Hutter, and M. Parrinello. “The Gaussian and augmented-plane-wave density functional method for ab initio molecular dynamics simulations”. In: *Theor. Chem. Acc.* 103 (1999), pages 124–40 (cited on page 55).

[131] C. E. Dykstra. “Examination of Brueckner condition for selection of molecular-orbitals in correlated wavefunctions”. In: *Chem. Phys. Lett.* 45 (1977), pages 466–69 (cited on page 59).

[132] N. C. Handy, J. A. Pople, M. Head-Gordon, K. Raghavachari, and G. W. Trucks. “Size-consistent Brueckner theory limited to double substitutions”. In: *Chem. Phys. Lett.* 164 (1989), pages 185–92 (cited on page 59).

[133] R. Kobayashi, N. C. Handy, R. D. Amos, G. W. Trucks, M. J. Frisch, and J. A. Pople. “Gradient theory applied to the Brueckner doubles method”. In: *J. Chem. Phys.* 95 (1991), pages 6723–33 (cited on page 59).

[134] K. Raghavachari, J. A. Pople, E. S. Replogle, and M. Head-Gordon. “Fifth Order Mller-Plesset Perturbation Theory: Comparison of Existing Correlation Methods and Implementation of New Methods Correct to Fifth Order”. In: *J. Phys. Chem.* 94 (1990), pages 5579–86 (cited on pages 59, 226, 286).

[135] T. Helgaker, E. Uggerud, and H. J. A. Jensen. “Integration of the Classical Equations of Motion on ab initio Molecular-Potential Energy Surfaces Using Gradients and Hessians - Application to Translational Energy-Release Upon Fragmentation”. In: *Chem. Phys. Lett.* 173 (1990), pages 145–50 (cited on page 61).

[136] E. Uggerud and T. Helgaker. “Dynamics of the Reaction CH2OH+ $\rightarrow$ CHO+ + H2. Translational Energy-Release from ab initio Trajectory Calculations”. In: *J. Am. Chem. Soc.* 114 (1992), pages 4265–68 (cited on page 61).

[137] K. Bolton, W. L. Hase, and G. H. Peslherbe. In: *Modern Methods for Multidimensional Dynamics Computation in Chemistry*. Edited by D. L. Thompson. Singapore: World Scientific, 1998, page 143 (cited on page 61).

[138] W. Chen, W. L. Hase, and H. B. Schlegel. “Ab initio classical trajectory study of H2CO $\rightarrow$ H2 + CO dissociation”. In: *Chem. Phys. Lett.* 228 (1994), pages 436–42 (cited on page 61).

[139] J. M. Millam, V. Bakken, W. Chen, W. L. Hase, and H. B. Schlegel. “Ab initio classical trajectories on the Born-Oppenheimer Surface: Hessian-Based Integrators using Fifth Order Polynomial and Rational Function Fits”. In: *J. Chem. Phys.* 111 (1999), pages 3800–05 (cited on page 61).

[140] X. Li, J. M. Millam, and H. B. Schlegel. “Ab initio molecular dynamics studies of the photodissociation of formaldehyde, H2CO $\rightarrow$ H2 + CO: Direct classical trajectory calculations by MP2 and density functional theory”. In: *J. Chem. Phys.* 113 (2000), pages 10062–67 (cited on page 61).

[141] W. L. Hase, R. J. Duchovic, X. Hu, A. Komornicki, K. F. Lim, D.-H. Lu, G. H. Peslherbe, K. N. Swamy, S. R. V. Linde, A. Varandas, H. Wang, and R. J. Wolfe. “VENUS96: A General Chemical Dynamics Computer Program”. In: *QCPE* 16 (1996), page 671 (cited on page 61).

[142] D. Hegarty and M. A. Robb. “Application of unitary group-methods to configuration-interaction calculations”. In: *Mol. Phys.* 38 (1979), pages 1795–812 (cited on page 68).

[143] R. H. A. Eade and M. A. Robb. “Direct minimization in MC SCF theory - the Quasi-Newton method”. In: *Chem. Phys. Lett.* 83 (1981), pages 362–68 (cited on page 68).

[144] H. B. Schlegel and M. A. Robb. “MC SCF gradient optimization of the H2CO $\rightarrow$ H2 + CO transition structure”. In: *Chem. Phys. Lett.* 93 (1982), pages 43–46 (cited on page 68).

[145] F. Bernardi, A. Bottini, J. J. W. McDougall, M. A. Robb, and H. B. Schlegel. “MCSCF gradient calculation of transition structures in organic reactions”. In: *Far. Symp. Chem. Soc.* 19 (1984), pages 137–47 (cited on pages 68, 69).

[146] M. J. Frisch, I. N. Ragazos, M. A. Robb, and H. B. Schlegel. “An Evaluation of 3 Direct MC-SCF Procedures”. In: *Chem. Phys. Lett.* 189 (1992), pages 524–28 (cited on page 68).

[147] N. Yamamoto, T. Vreven, M. A. Robb, M. J. Frisch, and H. B. Schlegel. “A Direct Derivative MC-SCF Procedure”. In: *Chem. Phys. Lett.* 250 (1996), pages 373–78 (cited on page 68).

[148] P. E. M. Siegbahn. “A new direct CI method for large CI expansions in a small orbital space”. In: *Chem. Phys. Lett.* 109 (1984), pages 417–23 (cited on page 68).

[149] M. A. Robb and U. Niazi. “The Unitary Group Approach to Electronic Structure Computations”. In: *Reports in Molecular Theory*. Edited by H. Weinstein and G. Náray-Szabó. Volume 1. Boca Raton, FL: CRC Press, 1990, pages 23–55 (cited on page 68).

[150] M. Klene, M. A. Robb, M. J. Frisch, and P. Celani. “Parallel implementation of the CI-vector evaluation in full CI/CAS-SCF”. In: *J. Chem. Phys.* 113 (2000), pages 5653–65 (cited on page 68).

[151] S. Li. “Development of algorithms for the direct multi-configuration self-consistent field (MCSCF) method”. Supervisors: M. A. Robb and M. Bearpark. URL: `spiral.imperial.ac.uk:8443/handle/10044/1/6945`. PhD thesis. London, UK: Imperial College, 2011 (cited on page 69).

[152] J. B. Foresman and Æ. Frisch. *Exploring Chemistry with Electronic Structure Methods*. 3rd ed. ISBN: 978-1-935522-03-4. Wallingford, CT: Gaussian, Inc., 2015 (cited on pages 69, 89, 128, 238, 295).

[153] F. Bernardi, A. Bottoni, M. J. Field, M. F. Guest, I. H. Hillier, M. A. Robb, and A. Venturini. “MC-SCF Study of the Diels-Alder Reaction Between Ethylene and Butadiene”. In: *J. Am. Chem. Soc.* 110 (1988), pages 3050–55 (cited on page 69).

[154] F. Bernardi, A. Bottoni, M. Olivucci, M. A. Robb, H. B. Schlegel, and G. Tonachini. “Do Supra-Antara Paths Really Exist for 2+2 Cycloaddition Reactions? Analytical Computation of the MC-SCF Hessians for Transition States of C2H4 with C2H4, Singlet O2, and Ketene”. In: *J. Am. Chem. Soc.* 110 (1988), pages 5993–95 (cited on page 69).

[155] F. Bernardi, A. Bottoni, M. A. Robb, and A. Venturini. “MC-SCF study of the cycloaddition reaction between ketene and ethylene”. In: *J. Am. Chem. Soc.* 112 (1990), pages 2106–14 (cited on page 69).

[156] G. Tonachini, H. B. Schlegel, F. Bernardi, and M. A. Robb. “MC-SCF Study of the Addition Reaction of the ${}^1D_g$ Oxygen Molecule to Ethene”. In: *J. Am. Chem. Soc.* 112 (1990), pages 483–91 (cited on page 69).

[157] F. Bernardi, M. Olivucci, I. Palmer, and M. A. Robb. “An MC-SCF study of the thermal and photochemical cycloaddition of Dewar benzene”. In: *J. Organic Chem.* 57 (1992), pages 5081–87 (cited on page 69).

[158] I. J. Palmer, F. Bernardi, M. Olivucci, I. N. Ragazos, and M. A. Robb. “An MC-SCF study of the (photochemical) Paterno-Buchi reaction”. In: *J. Am. Chem. Soc.* 116 (1994), pages 2121–32 (cited on page 69).

[159] T. Vreven, F. Bernardi, M. Garavelli, M. Olivucci, M. A. Robb, and H. B. Schlegel. “Ab initio photoisomerization dynamics of a simple retinal chromophore model”. In: *J. Am. Chem. Soc.* 119 (1997), pages 12687–88 (cited on page 69).

[160] J. J. McDouall, K. Peasley, and M. A. Robb. “A Simple MC-SCF Perturbation Theory: Orthogonal Valence Bond Møller-Plesset 2 (OVB-MP2)”. In: *Chem. Phys. Lett.* 148 (1988), pages 183–89 (cited on page 69).

[161] I. N. Ragazos, M. A. Robb, F. Bernardi, and M. Olivucci. “Optimization and Characterization of the Lowest Energy Point on a Conical Intersection using an MC-SCF Lagrangian”. In: *Chem. Phys. Lett.* 197 (1992), pages 217–23 (cited on page 69).

[162] M. J. Bearpark, M. A. Robb, and H. B. Schlegel. “A Direct Method for the Location of the Lowest Energy Point on a Potential Surface Crossing”. In: *Chem. Phys. Lett.* 223 (1994), pages 269–74 (cited on page 69).

[163] F. Bernardi, M. A. Robb, and M. Olivucci. “Potential energy surface crossings in organic photochemistry”. In: *Chem. Soc. Reviews* 25 (1996), page 321 (cited on page 69).

[164] T. E. H. Walker. “Molecular spin-orbit coupling constants. Role of core polarization”. In: *J. Chem. Phys.* 52 (1970), page 1311 (cited on page 69).

[165] P. W. Abegg and T.-K. Ha. “Ab initio calculation of spin-orbit-coupling constant from Gaussian lobe SCF molecular wavefunctions”. In: *Mol. Phys.* 27 (1974), pages 763–67 (cited on page 69).

[166] P. W. Abegg. “Ab initio calculation of spin-orbit-coupling constants for Gaussian lobe and Gaussian-type wavefunctions”. In: *Mol. Phys.* 30 (1975), pages 579–96 (cited on page 69).

[167] R. Cimiraglia, M. Persico, and J. Tomasi. “Roto-electronic and spin-orbit couplings in the predissociation of HNO - a theoretical calculation”. In: *Chem. Phys. Lett.* 76 (1980), pages 169–71 (cited on page 69).

[168] S. Koseki, M. W. Schmidt, and M. S. Gordon. “MCSCF/6-31G(d,p) calculations of one-electron spin-orbit-coupling constants in diatomic-molecules”. In: *J. Phys. Chem.* 96 (1992), pages 10768–72 (cited on page 69).

[169] S. Koseki, M. S. Gordon, M. W. Schmidt, and N. Matsunaga. “Main-group effective nuclear charges for spin-orbit calculations”. In: *J. Phys. Chem.* 99 (1995), pages 12764–72 (cited on page 69).

[170] S. Koseki, M. W. Schmidt, and M. S. Gordon. “Effective nuclear charges for the first- through third-row transition metal elements in spin-orbit calculations”. In: *J. Phys. Chem. A* 102 (1998), pages 10430–35 (cited on page 69).

[171] J. Olsen, B. O. Roos, P. Jørgensen, and H. J. A. Jensen. “Determinant Based Configuration-Interaction Algorithms for Complete and Restricted Configuration-Interaction Spaces”. In: *J. Chem. Phys.* 89 (1988), pages 2185–92 (cited on page 69).

[172] M. Klene, M. A. Robb, L. Blancafort, and M. J. Frisch. “A New Efficient Approach to the Direct RASSCF Method”. In: *J. Chem. Phys.* 119 (2003), pages 713–28 (cited on page 69).

[173] T. P. Hamilton and P. Pulay. “UHF natural orbitals for defining and starting MC-SCF calculations”. In: *J. Chem. Phys.* 88 (1988), pages 4926–33 (cited on page 71).

[174] J. M. Bofill and P. Pulay. “The unrestricted natural orbital-complete active space (UNO-CAS) method: An inexpensive alternative to the complete active space-self-consistent-field (CAS-SCF) method”. In: *J. Chem. Phys.* 90 (1989), pages 3637–46 (cited on page 71).

[175] S. Clifford, M. J. Bearpark, and M. A. Robb. “A hybrid MC-SCF method: Generalized valence bond (GVB) with complete active space SCF (CASSCF)”. In: *Chem. Phys. Lett.* 255 (1996), pages 320–26 (cited on page 72).

[176] J. B. Foresman and Æ. Frisch. *Exploring Chemistry with Electronic Structure Methods*. 2nd ed. Pittsburgh, PA: Gaussian, Inc., 1996 (cited on pages 75, 93).

[177] M. R. Nyden and G. A. Petersson. “Complete basis set correlation energies. I. The asymptotic convergence of pair natural orbital expansions”. In: *J. Chem. Phys.* 75 (1981), pages 1843–62 (cited on pages 76, 78).

[178] G. A. Petersson, T. G. Tensfeldt, and J. A. Montgomery Jr. “A complete basis set model chemistry. III. The complete basis set-quadratic configuration interaction family of methods”. In: *J. Chem. Phys.* 94 (1991), pages 6091–101 (cited on pages 76, 78, 380).

[179] J. A. Montgomery Jr., J. W. Ochterski, and G. A. Petersson. “A complete basis set model chemistry. IV. An improved atomic pair natural orbital method”. In: *J. Chem. Phys.* 101 (1994), pages 5900–09 (cited on page 76).

[180] J. W. Ochterski, G. A. Petersson, and J. A. Montgomery Jr. “A complete basis set model chemistry. V. Extensions to six or more heavy atoms”. In: *J. Chem. Phys.* 104 (1996), pages 2598–619 (cited on pages 76, 380).

[181] J. A. Montgomery Jr., M. J. Frisch, J. W. Ochterski, and G. A. Petersson. “A complete basis set model chemistry. VII. Use of the minimum population localization method”. In: *J. Chem. Phys.* 112 (2000), pages 6532–42 (cited on pages 76, 77, 79, 271).

[182] G. P. F. Wood, L. Radom, G. A. Petersson, E. C. Barnes, M. J. Frisch, and J. A. Montgomery Jr. “A restricted-open-shell complete-basis-set model chemistry”. In: *J. Chem. Phys.* 125 (2006), 094106:1–16 (cited on page 76).

[183] J. Pipek and P. G. Mezey. “A fast intrinsic localization procedure applicable for ab initio and semiempirical linear combination of atomic orbital wave functions”. In: *J. Chem. Phys.* 90 (1989), pages 4916–26 (cited on page 79).

[184] S. F. Boys. “Construction of Molecular Orbitals to be Approximately Invariant for Changes from One Molecule to Another”. In: *Rev. Mod. Phys.* 32 (1960), pages 296–99 (cited on pages 79, 179).

[185] J. M. Foster and S. F. Boys. “Canonical configurational interaction procedure”. In: *Rev. Mod. Phys.* 32 (1960), pages 300–02 (cited on page 79).

[186] S. F. Boys. In: *Quantum Theory of Atoms, Molecules and the Solid State*. Edited by P.-O. Löwdin. New York: Academic Press, 1966, page 253 (cited on page 79).

[187] R. J. Bartlett and G. D. Purvis III. “Many-body perturbation-theory, coupled-pair many-electron theory, and importance of quadruple excitations for correlation problem”. In: *Int. J. Quantum Chem.* 14 (1978), pages 561–81 (cited on page 79).

[188] J. A. Pople, R. Krishnan, H. B. Schlegel, and J. S. Binkley. “Electron Correlation Theories and Their Application to the Study of Simple Reaction Potential Surfaces”. In: *Int. J. Quantum Chem.* 14 (1978), pages 545–60 (cited on page 79).

[189] J. Cíek. In: *Advances in Chemical Physics*. Edited by P. C. Hariharan. Volume 14. New York: Wiley Interscience, 1969, page 35 (cited on page 79).

[190] G. D. Purvis III and R. J. Bartlett. “A full coupled-cluster singles and doubles model - the inclusion of disconnected triples”. In: *J. Chem. Phys.* 76 (1982), pages 1910–18 (cited on pages 79, 80).

[191] G. E. Scuseria, C. L. Janssen, and H. F. Schaefer III. “An efficient reformulation of the closed-shell coupled cluster single and double excitation (CCSD) equations”. In: *J. Chem. Phys.* 89 (1988), pages 7382–87 (cited on page 79).

[192] G. E. Scuseria and H. F. Schaefer III. “Is coupled cluster singles and doubles (CCSD) more computationally intensive than quadratic configuration-interaction (QCISD)?” In: *J. Chem. Phys.* 90 (1989), pages 3700–03 (cited on page 79).

[193] J. D. Watts, J. Gauss, and R. J. Bartlett. “Coupled-cluster methods with noniterative triple excitations for restricted open-shell Hartree-Fock and other general single determinant reference functions. Energies and analytical gradients”. In: *J. Chem. Phys.* 98 (1993), page 8718 (cited on page 79).

[194] J. A. Pople, M. Head-Gordon, and K. Raghavachari. “Quadratic configuration interaction - a general technique for determining electron correlation energies”. In: *J. Chem. Phys.* 87 (1987), pages 5968–75 (cited on pages 80, 285, 286).

[195] T. J. Lee and P. R. Taylor. “A diagnostic for determining the quality of single-reference electron correlation methods”. In: *Int. J. Quantum Chem., Quant. Chem. Symp.* S23 (1989), pages 199–207 (cited on pages 80, 286).

[196] G. G. Hall and C. M. Smith. “Fitting electron-densities of molecules”. In: *Int. J. Quantum Chem.* 25 (1984), pages 881–90 (cited on page 81).

[197] C. M. Smith and G. G. Hall. “Approximation of electron-densities”. In: *Theor. Chem. Acc.* 69 (1986), pages 63–69 (cited on page 81).

[198] J. A. Pople, R. Seeger, and R. Krishnan. “Variational Configuration Interaction Methods and Comparison with Perturbation Theory”. In: *Int. J. Quantum Chem., Suppl.* Y-11 (1977), pages 149–63 (cited on pages 82, 226).

[199] K. Raghavachari, H. B. Schlegel, and J. A. Pople. “Derivative studies in configuration-interaction theory”. In: *J. Chem. Phys.* 72 (1980), pages 4654–55 (cited on page 82).

[200] K. Raghavachari and J. A. Pople. “Calculation of one-electron properties using limited configuration-interaction techniques”. In: *Int. J. Quantum Chem.* 20 (1981), pages 1067–71 (cited on pages 82, 119).

[201] J. B. Foresman, M. Head-Gordon, J. A. Pople, and M. J. Frisch. “Toward a Systematic Molecular Orbital Theory for Excited States”. In: *J. Phys. Chem.* 96 (1992), pages 135–49 (cited on page 83).

[202] M. Head-Gordon, R. J. Rico, M. Oumi, and T. J. Lee. “A Doubles Correction to Electronic Excited-States from Configuration-Interaction in the Space of Single Substitutions”. In: *Chem. Phys. Lett.* 219 (1994), pages 21–29 (cited on page 84).

[203] M. Head-Gordon, D. Maurice, and M. Oumi. “A Perturbative Correction to Restricted Open-Shell Configuration-Interaction with Single Substitutions for Excited-States of Radicals”. In: *Chem. Phys. Lett.* 246 (1995), pages 114–21 (cited on page 84).

[204] J. A. Pople and G. Segal. “Approximate self-consistent molecular orbital theory. 3. CNDO results for AB2 and AB3 systems”. In: *J. Chem. Phys.* 44 (1966), pages 3289–96 (cited on page 87).

[205] P. J. Mohr, B. N. Taylor, and D. B. Newell. “CODATA Recommended Values of the Fundamental Physical Constants: 2010”. In: *Rev. Mod. Phys.* 84 (2012), pages 1527–1605 (cited on pages 88, 89).

[206] P. J. Mohr, B. N. Taylor, and D. B. Newell. “CODATA Recommended Values of the Fundamental Physical Constants: 2010”. In: *Chem. Ref. Data* 41 (2012), page 043109 (cited on pages 88, 89).

[207] P. J. Mohr, B. N. Taylor, and D. B. Newell. “CODATA Recommended Values of the Fundamental Physical Constants: 2006”. In: *Rev. Mod. Phys.* 80 (2008), pages 633–730 (cited on page 88).

[208] P. J. Mohr and B. N. Taylor. “CODATA Recommended Values of the Fundamental Physical Constants: 1998”. In: *Rev. Mod. Phys.* 72 (2000), pages 351–495 (cited on page 88).

[209] E. R. Cohen and B. N. Taylor. “The 1986 Adjustment of the Fundamental Physical Constants”. In: *CODATA Bulletin*. Elmsford, NY: Pergamon, 1986 (cited on page 88).

[210] D. R. Lide, editor. *CRC Handbook of Chemistry and Physics*. 60th ed. Boca Raton, FL: CRC Press, 1980 (cited on page 88).

[211] M. L. McGlashan. “Manual of Symbols and Terminology for Physicochemical Quantities and Units”. In: *Pure and Applied Chemistry* 51 (1979), page 1 (cited on page 88).

[212] S. F. Boys and F. Bernardi. “Calculation of Small Molecular Interactions by Differences of Separate Total Energies - Some Procedures with Reduced Errors”. In: *Mol. Phys.* 19 (1970), page 553 (cited on page 89).

[213] S. Simon, M. Duran, and J. J. Dannenberg. “How does basis set superposition error change the potential surfaces for hydrogen bonded dimers?” In: *J. Chem. Phys.* 105 (1996), pages 11024–31 (cited on page 89).

[214] R. McWeeny. “Some recent advances in density matrix theory”. In: *Rev. Mod. Phys.* 32 (1960), pages 335–69 (cited on page 90).

[215] R. McWeeny. “Perturbation Theory for Fock-Dirac Density Matrix”. In: *Phys. Rev.* 126 (1962), page 1028 (cited on pages 90, 228).

[216] R. M. Stevens, R. M. Pitzer, and W. N. Lipscomb. “Perturbed Hartree-Fock calculations. 1. Magnetic susceptibility and shielding in LiH molecule”. In: *J. Chem. Phys.* 38 (1963), page 550 (cited on pages 90, 92).

[217] J. Gerratt and I. M. Mills. “Force constants and dipole-moment derivatives of molecules from perturbed Hartree-Fock calculations. I.” In: *J. Chem. Phys.* 49 (1968), page 1719 (cited on page 90).

[218] J. L. Dodds, R. McWeeny, W. T. Raynes, and J. P. Riley. “SCF theory for multiple perturbations”. In: *Mol. Phys.* 33 (1977), pages 611–17 (cited on page 90).

[219] J. L. Dodds, R. McWeeny, and A. J. Sadlej. “Self-consistent perturbation theory: Generalization for perturbation-dependent non-orthogonal basis set”. In: *Mol. Phys.* 34 (1977), pages 1779–91 (cited on page 90).

[220] K. Wolinski and A. Sadlej. “Self-consistent perturbation theory: Open-shell states in perturbation-dependent non-orthogonal basis sets”. In: *Mol. Phys.* 41 (1980), pages 1419–30 (cited on page 90).

[221] Y. Osamura, Y. Yamaguchi, and H. F. Schaefer III. “Analytic configuration-interaction (CI) gradient techniques for potential-energy hypersurfaces - a method for openshell molecular wave-functions”. In: *J. Chem. Phys.* 75 (1981), pages 2919–22 (cited on pages 90, 92).

[222] Y. Osamura, Y. Yamaguchi, and H. F. Schaefer III. “Generalization of analytic configuration-interaction (CI) gradient techniques for potential-energy hypersurfaces, including a solution to the coupled perturbed Hartree-Fock equations for multiconfiguration SCF molecular wave-functions”. In: *J. Chem. Phys.* 77 (1982), pages 383–90 (cited on pages 90, 92).

[223] P. Pulay. “2nd and 3rd derivatives of variational energy expressions - application to multi-configurational self-consistent field wave-functions”. In: *J. Chem. Phys.* 78 (1983), pages 5043–51 (cited on pages 90, 92).

[224] C. E. Dykstra and P. G. Jasien. “Derivative Hartree-Fock theory to all orders”. In: *Chem. Phys. Lett.* 109 (1984), pages 388–93 (cited on page 90).

[225] G. H. F. Diercksen, B. O. Roos, and A. J. Sadlej. “Legitimate calculation of 1st-order molecular-properties in the case of limited CI functions - dipole-moments”. In: *Chem. Phys.* 59 (1981), pages 29–39 (cited on pages 92, 362).

[226] G. H. F. Diercksen and A. J. Sadlej. “Perturbation-theory of the electron correlation-effects for atomic and molecular-properties - 2nd-order and 3rd-order correlation corrections to molecular dipole-moments and polarizabilities”. In: *J. Chem. Phys.* 75 (1981), pages 1253–66 (cited on pages 92, 362).

[227] N. C. Handy and H. F. Schaefer III. “On the evaluation of analytic energy derivatives for correlated wave-functions”. In: *J. Chem. Phys.* 81 (1984), pages 5031–33 (cited on pages 92, 227, 362).

[228] K. B. Wiberg, C. M. Hadad, T. J. LePage, C. M. Breneman, and M. J. Frisch. “An Analysis of the Effect of Electron Correlation on Charge Density Distributions”. In: *J. Phys. Chem.* 96 (1992), pages 671–79 (cited on pages 92, 93, 119).

[229] P. Hohenberg and W. Kohn. “Inhomogeneous Electron Gas”. In: *Phys. Rev.* 136 (1964), B864–B71 (cited on pages 95, 99).

[230] W. Kohn and L. J. Sham. “Self-Consistent Equations Including Exchange and Correlation Effects”. In: *Phys. Rev.* 140 (1965), A1133–A38 (cited on pages 95, 96, 99).

[231] R. G. Parr and W. Yang. *Density-functional theory of atoms and molecules*. Oxford: Oxford Univ. Press, 1989 (cited on page 95).

[232] D. R. Salahub and M. C. Zerner, editors. *The Challenge of d and f Electrons*. Washington, D.C.: ACS, 1989 (cited on page 95).

[233] J. K. Labanowski and J. W. Andzelm, editors. *Density Functional Methods in Chemistry*. New York: Springer-Verlag, 1991 (cited on page 95).

[234] J. Andzelm and E. Wimmer. “Density functional Gaussian-type-orbital approach to molecular geometries, vibrations, and reaction energies”. In: *J. Chem. Phys.* 96 (1992), pages 1280–303 (cited on page 95).

[235] A. D. Becke. “Density-functional thermochemistry. I. The effect of the exchange-only gradient correction”. In: *J. Chem. Phys.* 96 (1992), pages 2155–60 (cited on page 95).

[236] P. M. W. Gill, B. G. Johnson, J. A. Pople, and M. J. Frisch. “The performance of the Becke-Lee-Yang-Parr (B-LYP) density functional theory with various basis sets”. In: *Chem. Phys. Lett.* 197 (1992), pages 499–505 (cited on page 95).

[237] J. P. Perdew, J. A. Chevary, S. H. Vosko, K. A. Jackson, M. R. Pederson, D. J. Singh, and C. Fiolhais. “Atoms, molecules, solids, and surfaces: Applications of the generalized gradient approximation for exchange and correlation”. In: *Phys. Rev. B* 46 (1992), pages 6671–87 (cited on pages 95, 99, 100).

[238] G. E. Scuseria. “Comparison of coupled-cluster results with a hybrid of Hartree-Fock and density functional theory”. In: *J. Chem. Phys.* 97 (1992), pages 7528–30 (cited on page 95).

[239] A. D. Becke. “Density-functional thermochemistry. II. The effect of the Perdew-Wang generalized-gradient correlation correction”. In: *J. Chem. Phys.* 97 (1992), pages 9173–77 (cited on page 95).

[240] J. P. Perdew and Y. Wang. “Accurate and Simple Analytic Representation of the Electron Gas Correlation Energy”. In: *Phys. Rev. B* 45 (1992), pages 13244–49 (cited on page 95).

[241] J. P. Perdew, J. A. Chevary, S. H. Vosko, K. A. Jackson, M. R. Pederson, D. J. Singh, and C. Fiolhais. “Erratum: Atoms, molecules, solids, and surfaces - Applications of the generalized gradient approximation for exchange and correlation”. In: *Phys. Rev. B* 48 (1993), page 4978 (cited on pages 95, 99, 100).

[242] C. Sosa and C. Lee. “Density-functional description of transition structures using nonlocal corrections: Silylene insertion reactions into the hydrogen molecule”. In: *J. Chem. Phys.* 98 (1993), pages 8004–11 (cited on page 95).

[243] P. J. Stephens, F. J. Devlin, M. J. Frisch, and C. F. Chabalowski. “Ab initio Calculation of Vibrational Absorption and Circular Dichroism Spectra Using Density Functional Force Fields”. In: *J. Phys. Chem.* 98 (1994), pages 11623–27 (cited on page 95).

[244] P. J. Stephens, F. J. Devlin, C. S. Ashvar, C. F. Chabalowski, and M. J. Frisch. “Theoretical Calculation of Vibrational Circular Dichroism Spectra”. In: *Faraday Discuss.* 99 (1994), pages 103–19 (cited on page 95).

[245] A. Ricca and C. W. Bauschlicher Jr. “Successive H2O binding energies for Fe(H2O)N+”. In: *J. Phys. Chem.* 99 (1995), pages 9003–07 (cited on page 95).

[246] J. A. Pople, P. M. W. Gill, and B. G. Johnson. “Kohn-Sham density-functional theory within a finite basis set”. In: *Chem. Phys. Lett.* 199 (1992), pages 557–60 (cited on page 95).

[247] B. G. Johnson and M. J. Frisch. “Analytic second derivatives of the gradient-corrected density functional energy: Effect of quadrature weight derivatives”. In: *Chem. Phys. Lett.* 216 (1993), pages 133–40 (cited on page 95).

[248] B. G. Johnson and M. J. Frisch. “An implementation of analytic second derivatives of the gradient-corrected density functional energy”. In: *J. Chem. Phys.* 100 (1994), pages 7429–42 (cited on page 95).

[249] R. E. Stratmann, J. C. Burant, G. E. Scuseria, and M. J. Frisch. “Improving harmonic vibrational frequencies calculations in density functional theory”. In: *J. Chem. Phys.* 106 (1997), pages 10175–83 (cited on page 95).

[250] A. D. Becke. “Density-functional thermochemistry. III. The role of exact exchange”. In: *J. Chem. Phys.* 98 (1993), pages 5648–52 (cited on page 96).

[251] A. J. Cohen and N. C. Handy. “Dynamic correlation”. In: *Mol. Phys.* 99 (2001), pages 607–15 (cited on page 97).

[252] A. Austin, G. Petersson, M. J. Frisch, F. J. Dobek, G. Scalmani, and K. Throssell. “A density functional with spherical atom dispersion terms”. In: *J. Chem. Theory and Comput.* 8 (2012), page 4989 (cited on pages 97, 100).

[253] T. M. Henderson, A. F. Izmaylov, G. Scalmani, and G. E. Scuseria. “Can short-range hybrids describe long-range-dependent properties?” In: *J. Chem. Phys.* 131 (2009), page 044108 (cited on pages 97, 99).

[254] O. A. Vydrov and G. E. Scuseria. “Assessment of a long range corrected hybrid functional”. In: *J. Chem. Phys.* 125 (2006), page 234109 (cited on page 97).

[255] O. A. Vydrov, J. Heyd, A. Krukau, and G. E. Scuseria. “Importance of short-range versus long-range Hartree-Fock exchange for the performance of hybrid density functionals”. In: *J. Chem. Phys.* 125 (2006), page 074106 (cited on page 97).

[256] O. A. Vydrov, G. E. Scuseria, and J. P. Perdew. “Tests of functionals for systems with fractional electron number”. In: *J. Chem. Phys.* 126 (2007), page 154109 (cited on page 97).

[257] T. Yanai, D. Tew, and N. Handy. “A new hybrid exchange-correlation functional using the Coulomb-attenuating method (CAM-B3LYP)”. In: *Chem. Phys. Lett.* 393 (2004), pages 51–57 (cited on page 97).

[258] J.-D. Chai and M. Head-Gordon. “Long-range corrected hybrid density functionals with damped atom-atom dispersion corrections”. In: *Phys. Chem. Chem. Phys.* 10 (2008), pages 6615–20 (cited on page 97).

[259] J.-D. Chai and M. Head-Gordon. “Systematic optimization of long-range corrected hybrid density functionals”. In: *J. Chem. Phys.* 128 (2008), page 084106 (cited on page 97).

[260] H. Iikura, T. Tsuneda, T. Yanai, and K. Hirao. “Long-range correction scheme for generalized-gradient-approximation exchange functionals”. In: *J. Chem. Phys.* 115 (2001), pages 3540–44 (cited on page 97).

[261] H. S. Yu, X. He, S. L. Li, and D. G. Truhlar. “MN15: A Kohn-Sham Global-Hybrid Exchange-Correlation Density Functional with Broad Accuracy for Multi-Reference and Single-Reference Systems and Noncovalent Interactions”. In: *Chemical Science* 7 (2016), pages 5032–5051 (cited on page 97).

[262] R. Peverati and D. G. Truhlar. “Improving the Accuracy of Hybrid Meta-GGA Density Functionals by Range Separation”. In: *J. Phys. Chem. Lett.* 2 (2011), pages 2810–2817 (cited on page 97).

[263] R. Peverati and D. G. Truhlar. “A global hybrid generalized gradient approximation to the exchange-correlation functional that satisfies the second-order density-gradient constraint and has broad applicability in chemistry”. In: *J. Chem. Phys.* 135 (2011), page 191102 (cited on page 97).

[264] R. Peverati and D. G. Truhlar. “Screened-exchange density functionals with broad accuracy for chemistry and solidstate physics”. In: *Phys. Chem. Chem. Phys.* 14 (2012), page 16187 (cited on page 97).

[265] Y. Zhao and D. G. Truhlar. “Design of Density Functionals That Are Broadly Accurate for Thermochemistry, Thermochemical Kinetics, and Nonbonded Interactions”. In: *J. Phys. Chem. A* 109 (2005), page 5656 (cited on page 97).

[266] Y. Zhao and D. G. Truhlar. “Exploring the Limit of Accuracy of the Global Hybrid Meta Density Functional for Main-Group Thermochemistry, Kinetics, and Noncovalent Interactions”. In: *J. Chem. Theory Compute.* 4 (2008), page 1849 (cited on page 97).

[267] Y. Zhao and D. G. Truhlar. “The M06 suite of density functionals for main group thermochemistry, thermochemical kinetics, noncovalent interactions, excited states, and transition elements: two new functionals and systematic testing of four M06-class functionals and 12 other functionals”. In: *Theor. Chem. Acc.* 120 (2008), pages 215–41 (cited on page 97).

[268] Y. Zhao and D. G. Truhlar. “Comparative DFT study of van der Waals complexes: Rare-gas dimers, alkaline-earth dimers, zinc dimer, and zinc-rare-gas dimers”. In: *J. Phys. Chem.* 110 (2006), pages 5121–29 (cited on page 97).

[269] Y. Zhao and D. G. Truhlar. “Density Functional for Spectroscopy: No Long-Range Self-Interaction Error, Good Performance for Rydberg and Charge-Transfer States, and Better Performance on Average than B3LYP for Ground States”. In: *J. Phys. Chem. A* 110 (2006), pages 13126–30 (cited on page 97).

[270] Y. Zhao, N. E. Schultz, and D. G. Truhlar. “Exchange-correlation functional with broad accuracy for metallic and nonmetallic compounds, kinetics, and noncovalent interactions”. In: *J. Chem. Phys.* 123 (2005), page 161103 (cited on page 97).

[271] Y. Zhao, N. E. Schultz, and D. G. Truhlar. “Design of density functionals by combining the method of constraint satisfaction with parametrization for thermochemistry, thermochemical kinetics, and noncovalent interactions”. In: *J. Chem. Theory and Comput.* 2 (2006), pages 364–82 (cited on page 97).

[272] J. P. Perdew, K. Burke, and M. Ernzerhof. “Generalized gradient approximation made simple”. In: *Phys. Rev. Lett.* 77 (1996), pages 3865–68 (cited on pages 97, 99, 100).

[273] J. P. Perdew, K. Burke, and M. Ernzerhof. “Errata: Generalized gradient approximation made simple”. In: *Phys. Rev. Lett.* 78 (1997), page 1396 (cited on pages 97, 99, 100).

[274] C. Adamo and V. Barone. “Toward reliable density functional methods without adjustable parameters: The PBE0 model”. In: *J. Chem. Phys.* 110 (1999), pages 6158–69 (cited on page 97).

[275] M. Ernzerhof and G. E. Scuseria. “Assessment of the Perdew-Burke-Ernzerhof exchange-correlation functional”. In: *J. Chem. Phys.* 110 (1999). DOI: 10.1063/1.478401, pages 5029–36 (cited on page 97).

[276] J. Heyd and G. Scuseria. “Efficient hybrid density functional calculations in solids: The HS-Ernzerhof screened Coulomb hybrid functional”. In: *J. Chem. Phys.* 121 (2004), pages 1187–92 (cited on page 97).

[277] J. Heyd and G. E. Scuseria. “Assessment and validation of a screened Coulomb hybrid density functional”. In: *J. Chem. Phys.* 120 (2004), page 7274 (cited on page 97).

[278] J. Heyd, J. E. Peralta, G. E. Scuseria, and R. L. Martin. “Energy band gaps and lattice parameters evaluated with the Heyd-Scuseria-Ernzerhof screened hybrid functional”. In: *J. Chem. Phys.* 123 (2005), 174101:1–8 (cited on page 97).

[279] J. Heyd, G. E. Scuseria, and M. Ernzerhof. “Erratum: “Hybrid functionals based on a screened Coulomb potential””. In: *J. Chem. Phys.* 124 (2006), page 219906 (cited on page 97).

[280] A. F. Izmaylov, G. Scuseria, and M. J. Frisch. “Efficient evaluation of short-range Hartree-Fock exchange in large molecules and periodic systems”. In: *J. Chem. Phys.* 125 (2006), 104103:1–8 (cited on pages 97, 99, 119).

[281] A. V. Krukau, O. A. Vydrov, A. F. Izmaylov, and G. E. Scuseria. “Influence of the exchange screening parameter on the performance of screened hybrid functionals”. In: *J. Chem. Phys.* 125 (2006), page 224106 (cited on page 97).

[282] M. Ernzerhof and J. P. Perdew. “Generalized gradient approximation to the angle- and system-averaged exchange hole”. In: *J. Chem. Phys.* 109 (1998). DOI: 10.1063/1.476928, pages 3313–20 (cited on pages 98, 99).

[283] A. D. Becke. “Density-functional thermochemistry. IV. A new dynamical correlation functional and implications for exact-exchange mixing”. In: *J. Chem. Phys.* 104 (1996), pages 1040–46 (cited on pages 98, 100).

[284] C. Adamo and V. Barone. “Toward reliable adiabatic connection models free from adjustable parameters”. In: *Chem. Phys. Lett.* 274 (1997), pages 242–50 (cited on page 98).

[285] C. Adamo and V. Barone. “Exchange functionals with improved long-range behavior and adiabatic connection methods without adjustable parameters: The mPW and mPW1PW models”. In: *J. Chem. Phys.* 108 (1998), pages 664–75 (cited on pages 98, 99).

[286] A. D. Becke. “Density-functional thermochemistry. V. Systematic optimization of exchange-correlation functionals”. In: *J. Chem. Phys.* 107 (1997), pages 8554–60 (cited on page 98).

[287] H. L. Schmider and A. D. Becke. “Optimized density functionals from the extended G2 test set”. In: *J. Chem. Phys.* 108 (1998), pages 9624–31 (cited on page 98).

[288] F. A. Hamprecht, A. Cohen, D. J. Tozer, and N. C. Handy. “Development and assessment of new exchange-correlation functionals”. In: *J. Chem. Phys.* 109 (1998), pages 6264–71 (cited on pages 98, 100).

[289] P. J. Wilson, T. J. Bradley, and D. J. Tozer. “Hybrid exchange-correlation functional determined from thermochemical data and ab initio potentials”. In: *J. Chem. Phys.* 115 (2001), pages 9233–42 (cited on page 98).

[290] J. M. Tao, J. P. Perdew, V. N. Staroverov, and G. E. Scuseria. “Climbing the density functional ladder: Nonempirical meta-generalized gradient approximation designed for molecules and solids”. In: *Phys. Rev. Lett.* 91 (2003), page 146401 (cited on pages 98–100).

[291] V. N. Staroverov, G. E. Scuseria, J. Tao, and J. P. Perdew. “Comparative assessment of a new nonempirical density functional: Molecules and hydrogen-bonded complexes”. In: *J. Chem. Phys.* 119 (2003), page 12129 (cited on page 98).

[292] A. D. Boese and N. C. Handy. “New exchange-correlation density functionals: The role of the kinetic-energy density”. In: *J. Chem. Phys.* 116 (2002), pages 9559–69 (cited on pages 98, 100).

[293] A. D. Boese and J. M. L. Martin. “Development of Density Functionals for Thermochemical Kinetics”. In: *J. Chem. Phys.* 121 (2004), pages 3405–16 (cited on page 98).

[294] T. M. Henderson, A. F. Izmaylov, G. E. Scuseria, and A. Savin. “Assessment of a middle range hybrid functional”. In: *J. Chem. Theory and Comput.* 4 (2008), page 1254 (cited on page 98).

[295] X. Xu and W. A. Goddard III. “The X3LYP extended density functional for accurate descriptions of nonbond interactions, spin states, and thermochemical properties”. In: *Proc. Natl. Acad. Sci. USA* 101 (2004), pages 2673–77 (cited on page 98).

[296] A. D. Becke. “A new mixing of Hartree-Fock and local density-functional theories”. In: *J. Chem. Phys.* 98 (1993), pages 1372–77 (cited on page 98).

[297] J. C. Slater. *The Self-Consistent Field for Molecular and Solids, Quantum Theory of Molecular and Solids*. Volume 4. New York: McGraw-Hill, 1974 (cited on page 99).

[298] A. D. Becke. “Density-functional exchange-energy approximation with correct asymptotic-behavior”. In: *Phys. Rev. A* 38 (1988), pages 3098–100 (cited on page 99).

[299] J. P. Perdew. In: *Electronic Structure of Solids ’91*. Edited by P. Ziesche and H. Eschrig. Berlin: Akademie Verlag, 1991, page 11 (cited on pages 99, 100).

[300] J. P. Perdew, K. Burke, and Y. Wang. “Generalized gradient approximation for the exchange-correlation hole of a many-electron system”. In: *Phys. Rev. B* 54 (1996), pages 16533–39 (cited on pages 99, 100).

[301] K. Burke, J. P. Perdew, and Y. Wang. In: *Electronic Density Functional Theory: Recent Progress and New Directions*. Edited by J. F. Dobson, G. Vignale, and M. P. Das. Plenum, 1998 (cited on pages 99, 100).

[302] P. M. W. Gill. “A new gradient-corrected exchange functional”. In: *Mol. Phys.* 89 (1996), pages 433–45 (cited on page 99).

[303] C. Adamo and V. Barone. “Implementation and validation of the Lacks-Gordon exchange functional in conventional density functional and adiabatic connection methods”. In: *J. Comp. Chem.* 19 (1998), pages 418–29 (cited on page 99).

[304] N. C. Handy and A. J. Cohen. “Left-right correlation energy”. In: *Mol. Phys.* 99 (2001), pages 403–12 (cited on page 99).

[305] W.-M. Hoe, A. Cohen, and N. C. Handy. “Assessment of a new local exchange functional OPTX”. In: *Chem. Phys. Lett.* 341 (2001), pages 319–28 (cited on page 99).

[306] John P. Perdew, Adrienn Ruzsinszky, Gábor I. Csonka, Lucian A. Constantin, and Jianwei Sun. “Workhorse Semilocal Density Functional for Condensed Matter Physics and Quantum Chemistry”. In: *Phys. Rev. Lett.* 103 (2009), page 026403 (cited on pages 99, 100).

[307] John P. Perdew, Adrienn Ruzsinszky, Gábor I. Csonka, Lucian A. Constantin, and Jianwei Sun. “Erratum: “Workhorse Semilocal Density Functional for Condensed Matter Physics and Quantum Chemistry” [Phys. Rev. Lett. 103, 026403 (2009)]”. In: *Phys. Rev. Lett.* 106 (2011), 179902(E) (cited on pages 99, 100).

[308] A. D. Becke and M. R. Roussel. “Exchange holes in inhomogeneous systems: A coordinate-space model”. In: *Phys. Rev. A* 39 (1989), pages 3761–67 (cited on pages 99, 100).

[309] J. P. Perdew, S. Kurth, A. Zupan, and P. Blaha. “Accurate density functional with correct formal properties: A step beyond the generalized gradient approximation”. In: *Phys. Rev. Lett.* 82 (1999), pages 2544–47 (cited on pages 99, 100).

[310] J. Heyd, G. Scuseria, and M. Ernzerhof. “Hybrid functionals based on a screened Coulomb potential”. In: *J. Chem. Phys.* 118 (2003), pages 8207–15 (cited on page 99).

[311] S. H. Vosko, L. Wilk, and M. Nusair. “Accurate spin-dependent electron liquid correlation energies for local spin density calculations: A critical analysis”. In: *Can. J. Phys.* 58 (1980), pages 1200–11 (cited on page 100).

[312] C. Lee, W. Yang, and R. G. Parr. “Development of the Colle-Salvetti correlation-energy formula into a functional of the electron density”. In: *Phys. Rev. B* 37 (1988), pages 785–89 (cited on page 100).

[313] B. Miehlich, A. Savin, H. Stoll, and H. Preuss. “Results obtained with the correlation-energy density functionals of Becke and Lee, Yang and Parr”. In: *Chem. Phys. Lett.* 157 (1989), pages 200–06 (cited on page 100).

[314] J. P. Perdew and A. Zunger. “Self-interaction correction to density-functional approximations for many-electron systems”. In: *Phys. Rev. B* 23 (1981), pages 5048–79 (cited on page 100).

[315] J. P. Perdew. “Density-functional approximation for the correlation energy of the inhomogeneous electron gas”. In: *Phys. Rev. B* 33 (1986), pages 8822–24 (cited on page 100).

[316] J. Rey and A. Savin. “Virtual space level shifting and correlation energies”. In: *Int. J. Quantum Chem.* 69 (1998), pages 581–90 (cited on page 100).

[317] J. B. Krieger, J. Q. Chen, G. J. Iafrate, and A. Savin. In: *Electron Correlations and Materials Properties*. Edited by A. Gonis, N. Kioussis, and M. Ciftan. New York: Kluwer Academic, 1999, pages 463–77 (cited on page 100).

[318] J. B. Krieger, J. Q. Chen, and S. Kurth. In: *Density Functional Theory and its Application to Materials*. Edited by V. VanDoren, C. VanAlsenoy, and P. Geerlings. Volume 577. A.I.P. Conference Proceedings. New York: A.I.P., 2001, pages 48–69 (cited on page 100).

[319] J. Toulouse, A. Savin, and C. Adamo. “Validation and assessment of an accurate approach to the correlation problem in density functional theory: The Krieger-Chen-Iafrate-Savin model”. In: *J. Chem. Phys.* 117 (2002), pages 10465–73 (cited on page 100).

[320] T. Van Voorhis and G. E. Scuseria. “A never form for the exchange-correlation energy functional”. In: *J. Chem. Phys.* 109 (1998), pages 400–10 (cited on page 100).

[321] A. D. Boese, N. L. Doltsinis, N. C. Handy, and M. Sprik. “New generalized gradient approximation functionals”. In: *J. Chem. Phys.* 112 (2000), pages 1670–78 (cited on page 100).

[322] A. D. Boese and N. C. Handy. “A new parametrization of exchange-correlation generalized gradient approximation functionals”. In: *J. Chem. Phys.* 114 (2001), pages 5497–503 (cited on page 100).

[323] S. Grimme. “Semiempirical GGA-type density functional constructed with a long-range dispersion correction”. In: *J. Comp. Chem.* 27 (2006), pages 1787–99 (cited on page 100).

[324] S. Grimme, S. Ehrlich, and L. Goerigk. “Effect of the damping function in dispersion corrected density functional theory”. In: *J. Comp. Chem.* 32 (2011), pages 1456–65 (cited on pages 100, 102, 227).

[325] Y. Zhao and D. G. Truhlar. “A new local density functional for main-group thermochemistry, transition metal bonding, thermochemical kinetics, and noncovalent interactions”. In: *J. Chem. Phys.* 125 (2006), 194101:1–18 (cited on page 100).

[326] R. Peverati, Y. Zhao, and D. G. Truhlar. “Generalized Gradient Approximation That Recovers the Second-Order Density-Gradient Expansion with Optimized Across-the-Board Performance”. In: *J. Phys. Chem. Lett.* 2 (2011), pages 1991–1997 (cited on page 100).

[327] R. Peverati and D. G. Truhlar. “M11-L: A Local Density Functional That Provides Improved Accuracy for Electronic Structure Calculations in Chemistry and Physics”. In: *J. Phys. Chem. Lett.* 3 (2012), pages 117–124 (cited on page 100).

[328] R. Peverati and D. G. Truhlar. “An improved and broadly accurate local approximation to the exchange-correlation density functional: The MN12-L functional for electronic structure calculations in chemistry and physics”. In: *Phys. Chem. Chem. Phys.* 10 (2012), page 13171 (cited on page 100).

[329] R. Peverati and D. G. Truhlar. “Exchange-Correlation Functional with Good Accuracy for Both Structural and Energetic Properties while Depending Only on the Density and Its Gradient”. In: *J. Chem. Theory and Comput.* 8 (2012), pages 2310–2319 (cited on page 100).

[330] H. S. Yu, X. He, and D. G. Truhlar. “MN15-L: A New Local Exchange-Correlation Functional for Kohn-Sham Density Functional Theory with Broad Accuracy for Atoms, Molecules, and Solids”. In: *J. Chem. Theo. Comput.* 12 (2016), pages 1280–1293 (cited on page 100).

[331] S. Grimme, J. Antony, S. Ehrlich, and H. Krieg. “A consistent and accurate ab initio parameterization of density functional dispersion correction (DFT-D) for the 94 elements H-Pu”. In: *J. Chem. Phys.* 132 (2010), page 154104 (cited on page 101).

[332] L. Goerigk and S. Grimme. “Efficient and Accurate Double-Hybrid-Meta-GGA Density Functionals–Evaluation with the Extended GMTKN30 Database for General Main Group Thermochemistry, Kinetics, and Noncovalent Interactions”. In: *J. Chem. Theory Comput.* 7 (2011), pages 291–309 (cited on pages 101, 102, 227).

[333] D. Porezag, T. Frauenheim, T. Köhler, G. Seifert, and R. Kaschner. “Construction of tight-binding-like potentials on the basis of density-functional theory: Application to carbon”. In: *Phys. Rev. B* 51 (1995), pages 12947–57 (cited on page 103).

[334] M. Elstner, D. Porezag, G. Jungnickel, J. Elsner, M. Haugk, T. Frauenheim, S. Suhai, and G. Seifert. “Self-consistent-charge density-functional tight-binding method for simulations of complex materials properties”. In: *Phys. Rev. B* 58 (1998), pages 7260–68 (cited on page 103).

[335] G. Zheng, H. Witek, P. Bobadova-Parvanova, S. Irle, D. G. Musaev, R. Prabhakar, K. Morokuma, M. Lundberg, M. Elstner, C. Kohler, and T. Frauenheim. “Parameter calibration of transition-metal elements for the spin-polarized self-consistent-charge density-functional tight-binding (DFTB) method: Sc, Ti, Fe, Co and Ni”. In: *J. Chem. Theory and Comput.* 3 (2007), pages 1349–67 (cited on page 103).

[336] T. Frauenheim, G. Seifert, M. Elstner, Z. Hajnal, G. Jungnickel, D. Porezag, S. Suhai, and R. Scholz. “A self-consistent charge density-functional based tight-binding method for predictive materials simulations in physics, chemistry and biology”. In: *Physica Stat. Sol. B* 217 (2000), pages 41–62 (cited on page 103).

[337] T. Frauenheim, G. Seifert, M. Elstner, T. Niehaus, C. Kohler, M. Amkreutz, M. Sternberg, Z. Hajnal, A. D. Carlo, and S. Suhai. “Atomistic simulations of complex materials: ground-state and excited-state properties”. In: *J. Phys.: Condens. Matter* 14 (2002), pages 3015–47 (cited on page 103).

[338] G. Zheng, S. Irle, and K. Morokuma. “Performance of the DFTB method in comparison to DFT and semiempirical methods for geometries and energies of C20-C86 fullerene isomers”. In: *Chem. Phys. Lett.* 412 (2005), pages 210–16 (cited on page 103).

[339] N. Otte, M. Scholten, and W. Thiel. “Looking at self-consistent-charge density functional tight binding from a semiempirical perspective”. In: *J. Phys. Chem. A* 111 (2007), pages 5751–55 (cited on page 103).

[340] K. W. Sattelmeyer, J. Tirado-Rives, and W. L. Jorgensen. “Comparison of SCC-DFTB and NDDO-based semiempirical molecular orbital methods for organic molecules”. In: *J. Phys. Chem. A* 110 (2006), pages 13551–59 (cited on pages 103, 315).

[341] Th. Förster. “Zwischenmolekulare Energiewanderung und Fluoreszenz”. In: *Ann. Phys.* 437 (1948), pages 55–75 (cited on page 105).

[342] D. L. Dexter. “A Theory of Sensitized Luminescence in Solids”. In: *J. Chem. Phys.* 21 (1953), page 836 (cited on page 105).

[343] G. D. Scholes. “Long-range Resonance Energy Transfer in Molecular Systems”. In: *Annu. Rev. Phys. Chem.* 54 (2003), pages 57–87 (cited on pages 105, 111).

[344] C. Curutchet and B. Mennucci. “Towards a molecular scale interpretation of excitation energy transfer in solvated bichromophoric systems”. In: *J. Am. Chem. Soc.* 127 (2005), pages 16733–16744 (cited on page 105).

[345] V. Russo, C. Curutchet, and B. Mennucci. “Towards a molecular scale interpretation of excitation energy transfer in solvated bichromophoric systems. II. The through bond contribution”. In: *J. Phys. Chem. B* 111 (2007), pages 853–863 (cited on page 105).

[346] M. F. Iozzi, B. Mennucci, J. Tomasi, and R. Cammi. “Excitation energy transfer (EET) between molecules in condensed matter: A novel application of the polarizable continuum model (PCM)”. In: *J. Chem. Phys.* 120 (2004), page 7029 (cited on page 105).

[347] C. P. Hsu, G. R. Fleming, M. Head-Gordon, and T. Head-Gordon. “Excitation energy transfer in condensed media”. In: *J. Chem. Phys.* 114 (2001), page 3065 (cited on page 105).

[348] R. Cammi and B. Mennucci. “Linear response theory for the polarizable continuum model”. In: *J. Chem. Phys.* 110 (1999), pages 9877–86 (cited on page 105).

[349] R. Cammi, B. Mennucci, and J. Tomasi. “Fast evaluation of geometries and properties of excited molecules in solution: A Tamm-Dancoff model with application to 4-dimethylaminobenzonitrile”. In: *J. Phys. Chem. A* 104 (2000), pages 5631–37 (cited on pages 105, 299, 300).

[350] M. Caricato, B. Mennucci, and J. Tomasi. “Solvent effects on the electronic spectra: An extension of the polarizable continuum model to the ZINDO method”. In: *J. Phys. Chem. A* 108 (2004), pages 6248–56 (cited on page 105).

[351] M. Caricato, F. Ingrosso, B. Mennucci, and J. Tomasi. “Time-dependent polarizable continuum model: Theory and application”. In: *J. Chem. Phys.* 122 (2005), 154501:1–10 (cited on page 105).

[352] E. Cancès, B. Mennucci, and J. Tomasi. “A new integral equation formalism for the polarizable continuum model: Theoretical background and applications to isotropic and anistropic dielectrics”. In: *J. Chem. Phys.* 107 (1997), pages 3032–41 (cited on pages 105, 299).

[353] B. Mennucci, E. Cancès, and J. Tomasi. “Evaluation of Solvent Effects in Isotropic and Anisotropic Dielectrics, and in Ionic Solutions with a Unified Integral Equation Method: Theoretical Bases, Computational Implementation and Numerical Applications”. In: *J. Phys. Chem. B* 101 (1997), pages 10506–17 (cited on pages 105, 299).

[354] E. Cances and B. Mennucci. “Analytical derivatives for geometry optimization in solvation continuum models. I. Theory”. In: *J. Chem. Phys.* 109 (1998), pages 249–59 (cited on page 105).

[355] H. Koch and P. Jørgensen. “Coupled cluster response functions”. In: *J. Chem. Phys.* 93 (1990), pages 3333–44 (cited on pages 107, 108).

[356] J. F. Stanton and R. J. Bartlett. “Equation of motion coupled-cluster method: A systematic biorthogonal approach to molecular excitation energies, transition probabilities, and excited state properties”. In: *J. Chem. Phys.* 98 (1993), pages 7029–39 (cited on page 107).

[357] H. Koch, R. Kobayashi, A. Sánchez de Merás, and P. Jørgensen. “Calculation of size-intensive transition moments from the coupled cluster singles and doubles linear response function”. In: *J. Chem. Phys.* 100 (1994), page 4393 (cited on pages 107, 108).

[358] M. Kállay and J. Gauss. “Calculation of excited-state properties using general coupled-cluster and configuration-interaction models”. In: *J. Chem. Phys.* 121 (2004), page 9257 (cited on pages 107, 108).

[359] M. Caricato. “Exploring potential energy surfaces of electronic excited states in solution with the EOM-CCSD-PCM method”. In: *J. Chem. Theory and Comput.* 8 (2012), pages 5081–9 (cited on page 107).

[360] M. Caricato. “Absorption and Emission Spectra of Solvated Molecules with the EOM-CCSD-PCM Method”. In: *J. Chem. Theory Comput.* 8 (2012), page 4494 (cited on pages 107, 299, 300, 302).

[361] M. Caricato, F. Lipparini, G. Scalmani, C. Cappelli, and V. Barone. “Vertical electronic excitations in solution with the EOM-CCSD method combined with a polarizable explicit/implicit solvent model”. In: *J. Chem. Theory and Comput.* 9 (2013), page 3035 (cited on page 107).

[362] M. Caricato. “A Comparison between State-Specific and Linear-Response Formalisms for the Calculation of Vertical Electronic Transition Energy in Solution with the CCSD-PCM Method”. In: *J. Chem. Phys.* 139 (2013), page 044116 (cited on page 107).

[363] M. Caricato. “Implementation of the CCSD-PCM linear response function for frequency dependent properties in solution: Application to polarizability and specific rotation”. In: *J. Chem. Phys.* 139 (2013), 114103:1–6 (cited on page 107).

[364] M. Caricato. “A corrected-linear response formalism for the calculation of electronic excitation energies of solvated molecules with the CCSD-PCM method”. In: *Comput. Theoret. Chem.* 1040-1041 (2014), pages 99–105 (cited on page 107).

[365] J. Goings, M. Caricato, M. J. Frisch, and X. Li. “Assessment of low-scaling approximations to the equation of motion coupled-cluster singles and doubles equations”. In: *J. Chem. Phys.* 141 (2014), page 164116 (cited on page 107).

[366] R. Cammi. “Quantum cluster theory for the polarizable continuum model. I. The CCSD level with analytical first and second derivatives”. In: *J. Chem. Phys.* 131 (2009), page 164104 (cited on pages 107, 299, 300, 302).

[367] R. Cammi. “Coupled-cluster theories for the polarizable continuum model. II. Analytical gradients for excited states of molecular solutes by the equation of motion coupled-cluster method”. In: *Int. J. Quant. Chem.* 110 (2010), pages 3040–52 (cited on pages 107, 299, 300).

[368] A. B. Migdal. *Theory of Finite Fermi Systems and Applications to Atomic Nuclei*. New York: Wiley Interscience, 1967 (cited on page 110).

[369] L. S. Cederbaum. “One-body Green’s function for atoms and molecules: Theory and application”. In: *J. Phys. B* 8 (1975), pages 290–303 (cited on page 110).

[370] L. S. Cederbaum and W. Domcke. In: *Advances in Chemical Physics*. Edited by I. Prigogine and S. A. Rice. Volume 36. New York: Wiley & Sons, 1977, page 205 (cited on page 110).

[371] Y. Öhrn and G. Born. In: *Advances in Quantum Chemistry*. Edited by P.-O. Löwdin. Volume 13. San Diego, CA: Academic Press, 1981, pages 1–88 (cited on page 110).

[372] W. von Niessen, J. Schirmer, and L. S. Cederbaum. “Computational methods for the one-particle Green’s function”. In: *Comp. Phys. Rep.* 1 (1984), pages 57–125 (cited on page 110).

[373] J. V. Ortiz. “Electron binding energies of anionic alkali metal atoms from partial fourth order electron propagator theory calculations”. In: *J. Chem. Phys.* 89 (1988), pages 6348–52 (cited on page 110).

[374] J. V. Ortiz. “Partial fourth order electron propagator theory”. In: *Int. J. Quantum Chem., Quant. Chem. Symp.* 34 (S22) (1988), pages 431–36 (cited on page 110).

[375] J. V. Ortiz. “Electron propagator calculations with nondiagonal partial 4th-order self-energies and unrestricted Hartree-Fock reference states”. In: *Int. J. Quantum Chem., Quant. Chem. Symp.* S23 (1989), pages 321–32 (cited on page 110).

[376] V. G. Zakrzewski and W. von Niessen. “Vectorizable algorithm for Green function and many-body perturbation methods”. In: *J. Comp. Chem.* 14 (1993), pages 13–18 (cited on page 110).

[377] V. G. Zakrzewski and J. V. Ortiz. “Semidirect algorithms in electron propagator calculations”. In: *Int. J. Quantum Chem., Quant. Chem. Symp.* S28 (1994), pages 23–27 (cited on page 110).

[378] V. G. Zakrzewski and J. V. Ortiz. “Semidirect algorithms for third-order electron propagator calculations”. In: *Int. J. Quantum Chem.* 53 (1995), pages 583–90 (cited on page 110).

[379] J. V. Ortiz. “Partial third-order quasiparticle theory: Comparisons for closed-shell ionization energies and an application to the Borazine photoelectron spectrum”. In: *J. Chem. Phys.* 104 (1996), pages 7599–605 (cited on page 110).

[380] V. G. Zakrzewski, J. V. Ortiz, J. A. Nichols, D. Heryadi, D. L. Yeager, and J. T. Golab. “Comparison of perturbative and multiconfigurational electron propagator methods”. In: *Int. J. Quant. Chem.* 60 (1996), pages 29–36 (cited on page 110).

[381] J. V. Ortiz, V. G. Zakrzewski, and O. Dolgounircheva. In: *Conceptual Perspectives in Quantum Chemistry*. Edited by J.-L. Calais and E. Kryachko. Dordrecht: Kluwer Academic, 1997, pages 465–518 (cited on page 110).

[382] A. M. Ferreira, G. Seabra, O. Dolgounitcheva, V. G. Zakrzewski, and J. V. Ortiz. In: *Quantum-Mechanical Prediction of Thermochemical Data*. Edited by J. Cioslowski. Volume 22. Understanding Chemical Reactivity. Dordrecht: Kluwer Academic, 2001, pages 131–60 (cited on page 110).

[383] J. Linderberg and Y. Öhrn. *Propagators in Quantum Chemistry*. 2nd ed. Hoboken, NJ: Wiley & Sons, 2004 (cited on page 110).

[384] J. V. Ortiz. “An efficient, renormalized self-energy for calculating the electron binding energies of closed-shell molecules and anions”. In: *Int. J. Quantum Chem.* 105 (2005), pages 803–808 (cited on page 110).

[385] M. Díaz-Tinoco, O. Dolgounitcheva, V. G. Zakrzewski, and J. V. Ortiz. “Composite electron propagator methods for calculating ionization energies”. In: *J. Chem. Phys.* 144 (2016), pages 224110–12 (cited on pages 110, 386).

[386] V. G. Zakrzewski, O. Dolgounitcheva, A. V. Zakjevskii, and J. V. Ortiz. “Ab initio Electron Propagator Calculations on Electron Detachment Energies of Fullerenes, Macrocyclic Molecules and Nucleotide Fragments”. In: *Advances in Quantum Chemistry* 62 (2011), pages 105–136 (cited on page 111).

[387] L. Greengard and V. Rokhlin. “A fast algorithm for particle simulations”. In: *J. Comp. Phys.* 73 (1987), pages 325–48 (cited on page 119).

[388] L. Greengard. *The Rapid Evaluation of Potential Fields in Particle Systems*. Cambridge, MA: MIT Press, 1988 (cited on page 119).

[389] L. Greengard. “Fast algorithms for classical physics”. In: *Science* 265 (1994), pages 909–14 (cited on page 119).

[390] J. C. Burant, M. C. Strain, G. E. Scuseria, and M. J. Frisch. “Kohn-Sham Analytic Energy Second Derivatives with the Gaussian Very Fast Multipole Method (GvFMM)”. In: *Chem. Phys. Lett.* 258 (1996), pages 45–52 (cited on page 119).

[391] J. C. Burant, M. C. Strain, G. E. Scuseria, and M. J. Frisch. “Analytic Energy Gradients for the Gaussian Very Fast Multipole Method (GvFMM)”. In: *Chem. Phys. Lett.* 248 (1996), pages 43–49 (cited on page 119).

[392] J. C. Burant, G. E. Scuseria, and M. J. Frisch. “Linear scaling method for Hartree-Fock exchange calculations of large molecules”. In: *J. Chem. Phys.* 105 (1996), pages 8969–72 (cited on page 119).

[393] M. C. Strain, G. E. Scuseria, and M. J. Frisch. “Achieving Linear Scaling for the Electronic Quantum Coulomb Problem”. In: *Science* 271 (1996), pages 51–53 (cited on page 119).

[394] J. M. Millam and G. E. Scuseria. “Linear scaling conjugate gradient density matrix search as an alternative to diagonalization for first principles electronic structure calculations”. In: *J. Chem. Phys.* 106 (1997), pages 5569–77 (cited on page 119).

[395] J. R. Cheeseman, M. J. Frisch, F. J. Devlin, and P. J. Stephens. “Ab Initio Calculation of Atomic Axial Tensors and Vibrational Rotational Strengths Using Density Functional Theory”. In: *Chem. Phys. Lett.* 252 (1996), pages 211–20 (cited on page 121).

[396] T. Helgaker, K. Ruud, K. L. Bak, P. Jørgensen, and J. Olsen. “Vibrational Raman Optical-Activity Calculations Using London Atomic Orbitals”. In: *Faraday Discuss.* 99 (1994), pages 165–80 (cited on pages 121, 266).

[397] R. K. Dukor and L. A. Nafie. In: *Encyclopedia of Analytical Chemistry: Instrumentation and Applications*. Edited by R. A. Meyers. Chichester: Wiley & Sons, 2000, pages 662–76 (cited on page 121).

[398] K. Ruud, T. Helgaker, and P. Bour. “Gauge-origin independent density-functional theory calculations of vibrational Raman optical activity”. In: *J. Phys. Chem. A* 106 (2002), pages 7448–55 (cited on page 121).

[399] L. D. Barron. *Molecular Light Scattering and Optical Activity*. 2nd ed. Cambridge, UK: Cambridge University Press, 2004 (cited on pages 121, 267).

[400] A. J. Thorvaldsen, K. Ruud, K. Kristensen, P. Jørgensen, and S. Coriani. “A density matrix-based quasienergy formulation of the Kohn-Sham density functional response theory using perturbation- and time-dependent basis sets”. In: *J. Chem. Phys.* 129 (2008), page 214108 (cited on page 121).

[401] J. R. Cheeseman and M. J. Frisch. “Basis Set Dependence of Vibrational Raman and Raman Optical Activity Intensities”. In: *J. Chem. Theory and Comput.* 7 (2011), pages 3323–3334 (cited on pages 121, 123, 266, 267, 270).

[402] S. Califano. *Vibrational States*. London: Wiley, 1976 (cited on page 121).

[403] W. H. Miller, N. C. Handy, and J. E. Adams. “Reaction-path Hamiltonian for polyatomic-molecules”. In: *J. Chem. Phys.* 72 (1980), pages 99–112 (cited on page 121).

[404] D. Papousek and M. R. Aliev. In: *Molecular Vibrational Spectra*. Edited by J. R. Durig. New York: Elsevier, 1982 (cited on page 121).

[405] D. A. Clabo, W. D. Allen, R. B. Remington, Y. Yamaguchi, and H. F. Schaefer III. “A systematic study of molecular vibrational anharmonicity and vibration-rotation interaction by self-consistent-field higher-derivative methods - asymmetric-top molecules”. In: *Chem. Phys.* 123 (1988), pages 187–239 (cited on page 121).

[406] M. Page and J. W. McIver Jr. “On evaluating the reaction path Hamiltonian”. In: *J. Chem. Phys.* 88 (1988), pages 922–35 (cited on pages 121, 196).

[407] C. Adamo, M. Cossi, N. Rega, and V. Barone. In: *Theoretical Biochemistry: Processes and Properties of Biological Systems, Theoretical and Computational Chemistry*. Volume 9. New York: Elsevier, 1990 (cited on page 121).

[408] W. H. Miller, R. Hernandez, N. C. Handy, D. Jayatilaka, and A. Willets. “Ab initio calculation of anharmonic constants for a transition-state, with application to semiclassical transition-state tunneling probabilities”. In: *Chem. Phys. Lett.* 172 (1990), pages 62–68 (cited on page 121).

[409] M. Page, C. Doubleday Jr., and J. W. McIver Jr. “Following steepest descent reaction paths - the use of higher energy derivatives with ab initio electronic-structure methods”. In: *J. Chem. Phys.* 93 (1990), pages 5634–42 (cited on pages 121, 196).

[410] M. Cossi, N. Rega, G. Scalmani, and V. Barone. “Energies, structures, and electronic properties of molecules in solution with the C-PCM solvation model”. In: *J. Comp. Chem.* 24 (2003), pages 669–81 (cited on pages 121, 263, 299, 301).

[411] V. Barone. “Vibrational zero-point energies and thermodynamic functions beyond the harmonic approximation”. In: *J. Chem. Phys.* 120 (2004), pages 3059–65 (cited on page 121).

[412] V. Barone. “Anharmonic vibrational properties by a fully automated second-order perturbative approach”. In: *J. Chem. Phys.* 122 (2005), 014108:1–10 (cited on page 121).

[413] V. Barone. “Characterization of the potential energy surface of the HO_2 molecular system by a density functional approach”. In: *J. Chem. Phys.* 101 (1994), pages 10666–76 (cited on pages 121, 263).

[414] C. Minichino and V. Barone. “From concepts to algorithms for the characterization of reaction mechanisms. H_2CS as a case study”. In: *J. Chem. Phys.* 100 (1994), pages 3717–41 (cited on pages 122, 263).

[415] V. Barone and C. Minichino. “From concepts to algorithms for the treatment of large-amplitude internal motions and unimolecular reactions”. In: *J. Mol. Struct. (Theochem)* 330 (1995), pages 365–76 (cited on pages 122, 263).

[416] J. Bloino and V. Barone. “A second-order perturbation theory route to vibrational averages and transition properties of molecules: General formulation and application to infrared and vibrational circular dichroism spectroscopies”. In: *J. Chem. Phys.* 136 (2012), page 124108 (cited on page 122).

[417] J. Bloino. “A VPT2 Route to Near-Infrared Spectroscopy: The Role of Mechanical and Electrical Anharmonicity”. In: *J. Phys. Chem. A* 119 (2015), pages 5269–5287 (cited on page 122).

[418] K. M. Kuhler, D. G. Truhlar, and A. D. Isaacson. “General Method for Removing Resonance Singularities in Quantum Mechanical Perturbation Theory”. In: *J. Chem. Phys.* 104 (1996), pages 4664–4671 (cited on page 122).

[419] J. Bloino, M. Biczysko, and V. Barone. “General perturbative approach for spectroscopy, thermodynamics and kinetics: Methodological background and benchmark studies”. In: *J. Chem. Theo. Comput.* 8 (2012), pages 1015–1036 (cited on page 122).

[420] J. Bloino, M. Biczysko, and V. Barone. “Anharmonic Effects on Vibrational Spectra Intensities: Infrared, Raman, Vibrational Circular Dichroism and Raman Optical Activity”. In: *J. Phys. Chem. A* 119 (2015), pages 11862–11874 (cited on page 122).

[421] C. Cappelli, F. Lipparini, J. Bloino, and V. Barone. “Towards an accurate description of anharmonic infrared spectra in solution within the polarizable continuum model: Reaction field, cavity field and nonequilibrium effects”. In: *J. Chem. Phys.* 135 (2011), page 104505 (cited on page 122).

[422] T. E. Sharp and H. M. Rosenstock. “Franck-Condon factors for polyatomic molecules”. In: *J. Chem. Phys.* 41 (1964), page 3453 (cited on pages 122, 124).

[423] E. V. Doktorov, I. A. Malkin, and V. I. Manko. “Dynamical symmetry of vibronic transitions in polyatomic-molecules and Franck-Condon principle. 2.” In: *J. Mol. Spec.* 64 (1977), pages 302–26 (cited on pages 122, 124).

[424] H. Kupka and P. H. Cribb. “Multidimensional Franck-Condon integrals and Duschinsky mixing effects”. In: *J. Chem. Phys.* 85 (1986), pages 1303–15 (cited on pages 122, 124).

[425] Z. Chen. “Rotation procedure in intrinsic reaction coordinate calculations”. In: *Theor. Chim. Acta.* 75 (1989), pages 481–84 (cited on pages 122, 124).

[426] R. Berger and M. Klessinger. “Algorithms for exact counting of energy levels of spectroscopic transitions at different temperatures”. In: *J. Comp. Chem.* 18 (1997), pages 1312–19 (cited on pages 122, 124).

[427] A. Peluso, F. Santoro, and G. del Re. “Vibronic coupling in electronic transitions with significant Duschinsky effect”. In: *Int. J. Quantum Chem.* 63 (1997), pages 233–44 (cited on pages 122, 124).

[428] R. Berger, C. Fischer, and M. Klessinger. “Calculation of the vibronic fine structure in electronic spectra at higher temperatures. 1. Benzene and pyrazine”. In: *J. Phys. Chem. A* 102 (1998), pages 7157–67 (cited on pages 122, 124).

[429] R. Borrelli and A. Peluso. “Dynamics of radiationless transitions in large molecular systems: A Franck-Condon-based method accounting for displacements and rotations of all the normal coordinates”. In: *J. Chem. Phys.* 119 (2003), pages 8437–48 (cited on pages 122, 124).

[430] J. Weber and G. Hohlneicher. “Franck-Condon factors for polyatomic molecules”. In: *Mol. Phys.* 101 (2003), pages 2125–44 (cited on pages 122, 124).

[431] E. A. Coutsias, C. Seok, and K. A. Dill. “Using quaternions to calculate RMSD”. In: *J. Comp. Chem.* 25 (2004), pages 1849–57 (cited on pages 122, 124).

[432] M. Dierksen and S. Grimme. “Density functional calculations of the vibronic structure of electronic absorption spectra”. In: *J. Chem. Phys.* 120 (2004), pages 3544–54 (cited on pages 122, 124).

[433] A. Lami, C. Petrongolo, and F. Santoro. In: *Conical Intersections: Electronic Structure, Dynamics & Spectroscopy*. Edited by W. Domcke, D. R. Yarkony, and H. Koppel. Singapore: World Scientific, 2004 (cited on pages 122, 124).

[434] M. Dierksen and S. Grimme. “The vibronic structure of electronic absorption spectra of large molecules: A time-dependent density functional study on the influence of “Exact” Hartree-Fock exchange”. In: *J. Phys. Chem. A* 108 (2004), pages 10225–37 (cited on pages 122, 124).

[435] M. Dierksen and S. Grimme. “An efficient approach for the calculation of Franck-Condon integrals of large molecules”. In: *J. Chem. Phys.* 122 (2005), page 244101 (cited on pages 122, 124).

[436] J. Liang and H. Y. Li. “Calculation of the multimode Franck-Condon factors based on the coherent state method”. In: *Mol. Phys.* 103 (2005), pages 3337–42 (cited on pages 122, 124).

[437] H.-C. Jankowiak, J. L. Stuber, and R. Berger. “Vibronic transitions in large molecular systems: rigorous prescreening conditions for Franck-Condon factors”. In: *J. Chem. Phys.* 127 (2007), page 234101 (cited on pages 122, 124).

[438] F. Santoro, R. Improta, A. Lami, J. Bloino, and V. Barone. “Effective method to compute Franck-Condon integrals for optical spectra of large molecules in solution”. In: *J. Chem. Phys.* 126 (2007), 084509:1–13 (cited on pages 122, 124).

[439] F. Santoro, A. Lami, R. Improta, and V. Barone. “Effective method to compute vibrationally resolved optical spectra of large molecules at finite temperature in the gas phase and in solution”. In: *J. Chem. Phys.* 126 (2007), page 184102 (cited on pages 122, 124).

[440] F. Santoro, A. Lami, R. Improta, J. Bloino, and V. Barone. “Effective method for the computation of optical spectra of large molecules at finite temperature including the Duschinsky and Herzberg-Teller effect: The Qx band of porphyrin as a case study”. In: *J. Chem. Phys.* 128 (2008), page 224311 (cited on pages 122, 124).

[441] V. Barone, J. Bloino, M. Biczysko, and F. Santoro. “Fully integrated approach to compute vibrationally resolved optical spectra: From small molecules to macrosystems”. In: *J. Chem. Theory and Comput.* 5 (2009), pages 540–54 (cited on pages 122, 124).

[442] J. Bloino, M. Biczysko, F. Santoro, and V. Barone. “General Approach to Compute Vibrationally Resolved One-Photon Electronic Spectra”. In: *J. Chem. Theo. Comput.* 6 (2010), pages 1256–1274 (cited on page 122).

[443] A. Baiardi, J. Bloino, and V. Barone. “General Time Dependent Approach to Vibronic Spectroscopy Including Franck-Condon, Herzberg-Teller and Duschinsky Effects”. In: *J. Chem. Theo. Comput.* 9 (2013), pages 4097–4115 (cited on page 122).

[444] G. Herzberg and E. Teller. “Fluctuation structure of electron transfer in multiatomic molecules”. In: *Z. Phys. Chemie* 21 (1933), page 410 (cited on pages 122, 124).

[445] G. J. Small. “Herzberg-Teller vibronic coupling and Duschinsky effect”. In: *J. Chem. Phys.* 54 (1971), page 3300 (cited on pages 122, 124).

[446] G. Orlandi and W. Siebrand. “Theory of vibronic intensity borrowing - Comparison of Herzberg-Teller and Born-Oppenheimer coupling”. In: *J. Chem. Phys.* 58 (1973), pages 4513–23 (cited on pages 122, 124).

[447] S. H. Lin and H. Eyring. “Study of Franck-Condon and Herzberg-Teller approximations”. In: *PNAS* 71 (1974), pages 3802–04 (cited on pages 122, 124).

[448] F. Egidi, J. Bloino, C. Cappelli, and V. Barone. “A Robust and Effective Time-Independent Route to the Calculation of Resonance Raman Spectra of Large Molecules in Condensed Phases with the Inclusion of Duschinsky, Herzberg-Teller, Anharmonic and Environmental Effects”. In: *J. Chem. Theo. Comput.* 10 (2014), pages 346–363 (cited on page 122).

[449] A. Baiardi, J. Bloino, and V. Barone. “A general time-dependent route to Resonance-Raman spectroscopy including Franck-Condon, Herzberg-Teller and Duschinsky effects”. In: *J. Chem. Phys.* 141 (2014), page 114108 (cited on page 122).

[450] V. Barone, A. Baiardi, M. Biczisko, J. Bloino, C. Cappelli, and F. Lipparini. “Implementation and validation of a multi-purpose virtual spectrometer for large systems in complex environments”. In: *Phys. Chem. Chem. Phys.* 14 (2012), pages 12404–422 (cited on page 122).

[451] V. Barone, A. Baiardi, and J. Bloino. “New Developments of a Multifrequency Virtual Spectrometer: Stereo-Electronic, Dynamical and Environmental Effects on Chiroptical Spectra”. In: *Chirality* 26 (2014), pages 588–600 (cited on page 122).

[452] J. Bloino, A. Baiardi, and M. Biczysko. “Aiming at an accurate prediction of vibrational and electronic spectra for medium-to-large molecules: An overview”. In: *Int. J. Quant. Chem.* 116 (2016), pages 1543–1574 (cited on page 122).

[453] A. G. Baboul and H. B. Schlegel. “Improved Method for Calculating Projected Frequencies along a Reaction Path”. In: *J. Chem. Phys.* 107 (1997), pages 9413–17 (cited on pages 124, 200).

[454] R. B. McClurg, R. C. Flagan, and W. A. Goddard III. “The hindered rotor density-of-states interpolation function”. In: *J. Chem. Phys.* 106 (1997), page 6675 (cited on page 125).

[455] P. Y. Ayala and H. B. Schlegel. “Identification and treatment of internal rotation in normal mode vibrational analysis”. In: *J. Chem. Phys.* 108 (1998), pages 2314–25 (cited on page 125).

[456] R. B. McClurg. “Comment on: “The hindered rotor density-of-states interpolation function” [J. Chem. Phys. 106, 6675 (1997)] and “The hindered rotor density-of-states” [J. Chem. Phys. 108, 2314 (1998)]”. In: *J. Chem. Phys.* 111 (1999), page 7163 (cited on page 125).

[457] D. A. McQuarrie. *Statistical Thermodynamics*. New York: Harper and Row, 1973 (cited on page 129).

[458] S. W. Benson. *Thermochemical Kinetics*. New York: Wiley and Sons, 1968 (cited on page 130).

[459] K. Huang and A. Rhys. “Theory of light absorption and non-radiative transitions in F-centres”. In: *Proc. Roy. Soc. A* 204 (1950), page 406 (cited on page 149).

[460] A. V. Marenich, J. L. Sonnenberg, H. P. Hratchian, and M. J. Frisch. in prep. (cited on pages 160, 238).

[461] C. Peng, P. Y. Ayala, H. B. Schlegel, and M. J. Frisch. “Using redundant internal coordinates to optimize equilibrium geometries and transition states”. In: *J. Comp. Chem.* 17 (1996), pages 49–56 (cited on pages 165, 166, 237, 241).

[462] J. A. Pople, M. Head-Gordon, D. J. Fox, K. Raghavachari, and L. A. Curtiss. “Gaussian-1 theory: A general procedure for prediction of molecular energies”. In: *J. Chem. Phys.* 90 (1989), pages 5622–29 (cited on page 174).

[463] L. A. Curtiss, C. Jones, G. W. Trucks, K. Raghavachari, and J. A. Pople. “Gaussian-1 theory of molecular energies for second-row compounds”. In: *J. Chem. Phys.* 93 (1990), pages 2537–45 (cited on page 174).

[464] L. A. Curtiss, K. Raghavachari, G. W. Trucks, and J. A. Pople. “Gaussian-2 theory for molecular energies of first- and second-row compounds”. In: *J. Chem. Phys.* 94 (1991), pages 7221–30 (cited on page 174).

[465] L. A. Curtiss, K. Raghavachari, P. C. Redfern, V. Rassolov, and J. A. Pople. “Gaussian-3 (G3) theory for molecules containing first and second-row atoms”. In: *J. Chem. Phys.* 109 (1998), pages 7764–76 (cited on page 174).

[466] L. A. Curtiss, P. C. Redfern, and K. Raghavachari. “Gaussian-4 theory”. In: *J. Chem. Phys.* 126 (2007), page 084108 (cited on page 174).

[467] L. A. Curtiss, K. Raghavachari, and J. A. Pople. “Gaussian-2 theory using reduced Møller-Plesset orders”. In: *J. Chem. Phys.* 98 (1993), pages 1293–98 (cited on page 174).

[468] L. A. Curtiss, P. C. Redfern, K. Raghavachari, V. Rassolov, and J. A. Pople. “Gaussian-3 theory using reduced Møller-Plesset order”. In: *J. Chem. Phys.* 110 (1999), pages 4703–09 (cited on page 174).

[469] A. G. Baboul, L. A. Curtiss, P. C. Redfern, and K. Raghavachari. “Gaussian-3 theory using density functional geometries and zero-point energies”. In: *J. Chem. Phys.* 110 (1999), pages 7650–57 (cited on page 174).

[470] L. A. Curtiss, P. C. Redfern, and K. Raghavachari. “Gaussian-4 theory using reduced order perturbation theory”. In: *J. Chem. Phys.* 127 (2007), page 124105 (cited on page 174).

[471] J. Harris. “Simplified method for calculating the energy of weakly interacting fragments”. In: *Phys. Rev. B* 31 (1985), pages 1770–79 (cited on page 177).

[472] F. W. Bobrowicz and W. A. Goddard III. In: *Methods of Electronic Structure Theory*. Edited by H. F. Schaefer III. Volume 3. Modern Theoretical Chemistry. New York: Plenum, 1977, pages 79–126 (cited on page 188).

[473] W. A. Goddard III and L. B. Harding. In: *Annual Review of Physical Chemistry*. Edited by B. S. Rabinovitch. Volume 29. Palo Alto, CA: Annual Reviews, Inc., 1978, pages 363–96 (cited on page 188).

[474] C. C. J. Roothaan. “New Developments in Molecular Orbital Theory”. In: *Rev. Mod. Phys.* 23 (1951), page 69 (cited on page 189).

[475] G. Berthier. “Extension de la methode du champ moleculaire self-consistent a l’etude des couches incompletes”. In: *Comptes rendus hebdomadaires des séances de l’Académie des sciences* 238 (1954), pages 91–93 (cited on page 189).

[476] J. A. Pople and R. K. Nesbet. “Self-Consistent Orbitals for Radicals”. In: *J. Chem. Phys.* 22 (1954), pages 571–72 (cited on page 189).

[477] R. McWeeny and G. Dierksen. “Self-consistent perturbation theory. 2. Extension to open shells”. In: *J. Chem. Phys.* 49 (1968), page 4852 (cited on page 189).

[478] R. Hoffmann. “An Extended Huckel Theory. I. Hydrocarbons”. In: *J. Chem. Phys.* 39 (1963), page 1397 (cited on page 190).

[479] R. Hoffmann. “An Extended Huckel Theory. II. Sigma Orbitals in the Azines”. In: *J. Chem. Phys.* 40 (1964), page 2745 (cited on page 190).

[480] R. Hoffmann. “An Extended Huckel Theory. III. Compounds of Boron and Nitrogen”. In: *J. Chem. Phys.* 40 (1964), page 2474 (cited on page 190).

[481] R. Hoffmann. “An Extended Huckel Theory. IV. Carbonium Ions”. In: *J. Chem. Phys.* 40 (1964), page 2480 (cited on page 190).

[482] R. Hoffmann. “Extended Huckel Theory. V. Cumulenes, Polyenes, Polyacetylenes and Cn”. In: *Tetrahedron* 22 (1966), page 521 (cited on page 190).

[483] P. Pyykko and L. L. Lohr. “Relativistically Parameterized Extended Huckel Calculations. 3. Structure and Bonding for Some Compounds of Uranium and Other Heavy-Elements”. In: *Inorganic Chem.* 20 (1981), pages 1950–59 (cited on page 190).

[484] P. Pyykko and L. Laaksonen. “Relativistically Parameterized Extended Huckel Calculations. 8. Double-Zeta Parameters for the Actinoids Th, Pa, U, Np, Pu, and Am and an Application on Uranyl”. In: *J. Phys. Chem.* 88 (1984), pages 4892–95 (cited on page 190).

[485] N. J. Fitzpatrick and G. H. Murphy. “Double Zeta-D Radial Wave-Functions for Transition-Elements”. In: *Inorg. Chim. Acta* 111 (1986), pages 139–40 (cited on page 190).

[486] J. A. Pople, D. Beveridge, and P. Dobosh. “Approximate self-consistent molecular-orbital theory. 5. Intermediate neglect of differential overlap”. In: *J. Chem. Phys.* 47 (1967), pages 2026–33 (cited on page 191).

[487] M. Krack and A. M. Köster. “An adaptive numerical integrator for molecular integrals”. In: *J. Chem. Phys.* 108 (1998), pages 3226–34 (cited on page 191).

[488] V. I. Lebedev. “Weights and Nodes of Gauss-Markov Quadrature Formulas of Orders 9 to 17 for the Sphere that are Invariant under the Octahedron Group with Inversion”. In: *Zh. Vychisl. Mat. Mat. Fiz.* 15 (1975), pages 48–54 (cited on page 192).

[489] A. D. McLaren. “Optimal Numerical Integration on a Sphere”. In: *Math. Comp.* 17 (1963), pages 361–83 (cited on page 192).

[490] V. I. Lebedev. “Quadratures on a Sphere”. In: *Zh. Vychisl. Mat. Mat. Fiz.* 16 (1976), pages 293–306 (cited on page 192).

[491] V. I. Lebedev. In: *Theory of Cubature Formulas and Computational Mathematics*. Edited by S. L. Sobolev. [in Russian]. Novosibirsk: Nauka, 1980, pages 75–82 (cited on page 192).

[492] V. I. Lebedev and L. Skorokhodov. “Quadrature formulas of orders 41, 47 and 53 for the sphere”. In: *Russian Acad. Sci. Dokl. Math.* 45 (1992), pages 587–92 (cited on page 192).

[493] R. E. Stratmann, G. E. Scuseria, and M. J. Frisch. “Achieving linear scaling in exchange-correlation density functional quadratures”. In: *Chem. Phys. Lett.* 257 (1996), pages 213–23 (cited on page 192).

[494] M. Douglas and N. M. Kroll. “Quantum electrodynamical corrections to fine-structure of helium”. In: *Ann. Phys. (NY)* 82 (1974), pages 89–155 (cited on page 193).

[495] B. A. Hess. “Applicability of the no-pair equation with free-particle projection operators to atomic and molecular-structure calculations”. In: *Phys. Rev. A* 32 (1985), pages 756–63 (cited on page 193).

[496] B. A. Hess. “Relativistic electronic-structure calculations employing a 2-component no-pair formalism with external-field projection operators”. In: *Phys. Rev. A* 33 (1986), pages 3742–48 (cited on page 193).

[497] G. Jansen and B. A. Hess. “Revision of the Douglas-Kroll transformation”. In: *Phys. Rev. A* 39 (1989), pages 6016–17 (cited on page 193).

[498] M. Barysz and A. J. Sadlej. “Two-component methods of relativistic quantum chemistry: From the Douglas-Kroll approximation to the exact two-component formalism”. In: *J. Mol. Struct. (Theochem)* 573 (2001), pages 181–200 (cited on page 193).

[499] W. A. deJong, R. J. Harrison, and D. A. Dixon. “Parallel Douglas-Kroll energy and gradients in NWChem: Estimating scalar relativistic effects using Douglas-Kroll contracted basis sets”. In: *J. Chem. Phys.* 114 (2001), pages 48–53 (cited on page 193).

[500] L. Visscher and K. G. Dyall. “Dirac-Fock atomic electronic structure calculations using different nuclear charge distributions”. In: *Atomic Data and Nuclear Data Tables* 67 (1997), pages 207–24 (cited on page 193).

[501] K. Fukui. “The path of chemical-reactions - The IRC approach”. In: *Acc. Chem. Res.* 14 (1981), pages 363–68 (cited on page 194).

[502] H. P. Hratchian and H. B. Schlegel. In: *Theory and Applications of Computational Chemistry: The First 40 Years*. Edited by C. E. Dykstra, G. Frenking, K. S. Kim, and G. Scuseria. Amsterdam: Elsevier, 2005, pages 195–249 (cited on pages 194, 196, 238).

[503] H. P. Hratchian and H. B. Schlegel. “Accurate reaction paths using a Hessian based predictor-corrector integrator”. In: *J. Chem. Phys.* 120 (2004), pages 9918–24 (cited on pages 194, 196).

[504] H. P. Hratchian and H. B. Schlegel. “Using Hessian updating to increase the efficiency of a Hessian based predictor-corrector reaction path following method”. In: *J. Chem. Theory and Comput.* 1 (2005), pages 61–69 (cited on pages 194, 196).

[505] M. A. Collins. “Molecular potential-energy surfaces for chemical reaction dynamics”. In: *Theor. Chem. Acc.* 108 (2002), pages 313–24 (cited on page 196).

[506] H. P. Hratchian and H. B. Schlegel. “Following reaction pathways using a damped classical trajectory algorithm”. In: *J. Phys. Chem. A* 106 (2002), pages 165–69 (cited on page 196).

[507] C. Gonzalez and H. B. Schlegel. “An Improved Algorithm for Reaction Path Following”. In: *J. Chem. Phys.* 90 (1989), pages 2154–61 (cited on pages 198, 199).

[508] C. Gonzalez and H. B. Schlegel. “Reaction Path Following in Mass-Weighted Internal Coordinates”. In: *J. Phys. Chem.* 94 (1990), pages 5523–27 (cited on pages 198, 199).

[509] H. Eyring. “The activated complex in chemical reactions”. In: *J. Chem. Phys.* 3 (1935), pages 107–15 (cited on pages 199, 201).

[510] D. G. Truhlar. “Adiabatic theory of chemical reactions”. In: *J. Chem. Phys.* 53 (1970), page 2041 (cited on pages 199, 201).

[511] D. G. Truhlar and A. Kuppermann. “Exact tunneling calculations”. In: *J. Am. Chem. Soc.* 93 (1971), page 1840 (cited on pages 199, 201).

[512] B. C. Garrett, D. G. Truhlar, R. S. Grev, and A. W. Magnusson. “Improved treatment of threshold contributions in variational transition-state theory”. In: *J. Phys. Chem.* 84 (1980), pages 1730–48 (cited on pages 199, 201).

[513] D. K. Malick, G. A. Petersson, and J. A. Montgomery Jr. “Transition states for chemical reactions. I. Geometry and classical barrier height”. In: *J. Chem. Phys.* 108 (1998), pages 5704–13 (cited on page 199).

[514] G. A. Petersson. In: *Computational Thermochemistry: Prediction and Estimation of Molecular Thermodynamics*. Edited by K. K. Irikura and D. J. Frurip. Volume 677. ACS Symposium Series. Washington, D.C.: ACS, 1998, page 237 (cited on pages 199, 201).

[515] M. Schwartz, P. Marshall, R. J. Berry, C. J. Ehlers, and G. A. Petersson. “Computational study of the kinetics of hydrogen abstraction from fluoromethanes by the hydroxyl radical”. In: *J. Phys. Chem. A* 102 (1998), pages 10074–81 (cited on page 199).

[516] G. A. Petersson, D. K. Malick, W. G. Wilson, J. W. Ochterski, J. A. Montgomery Jr., and M. J. Frisch. “Calibration and comparison of the Gaussian-2, complete basis set, and density functional methods for computational thermochemistry”. In: *J. Chem. Phys.* 109 (1998), pages 10570–79 (cited on page 199).

[517] R. T. Skodje, D. G. Truhlar, and B. C. Garrett. “Vibrationally adiabatic models for reactive tunneling”. In: *J. Chem. Phys.* 77 (1982), pages 5955–76 (cited on pages 199, 201).

[518] R. C. Bingham, M. J. S. Dewar, and D. H. Lo. “Ground-states of molecules. 25. MINDO-3 - Improved version of MINDO semiempirical SCF-MO method”. In: *J. Am. Chem. Soc.* 97 (1975), pages 1285–93 (cited on page 202).

[519] M. J. S. Dewar and W. Thiel. “Ground-States of Molecules. 38. The MNDO Method: Approximations and Parameters”. In: *J. Am. Chem. Soc.* 99 (1977), pages 4899–907 (cited on pages 202, 315).

[520] M. J. S. Dewar and H. S. Rzepa. “Ground-states of molecules. 45. MNDO results for molecules containing beryllium”. In: *J. Am. Chem. Soc.* 100 (1978), pages 777–84 (cited on page 202).

[521] M. J. S. Dewar, M. L. McKee, and H. S. Rzepa. “MNDO parameters for 3rd period elements”. In: *J. Am. Chem. Soc.* 100 (1978), pages 3607–07 (cited on pages 202, 315).

[522] L. P. Davis and et. al. “MNDO calculations for compounds containing aluminum and boron”. In: *J. Comp. Chem.* 2 (1981), pages 433–45 (cited on pages 202, 315).

[523] M. J. S. Dewar and M. L. McKee. “Ground-states of molecules. 56. MNDO calculations for molecules containing sulfur”. In: *J. Comp. Chem.* 4 (1983), pages 84–103 (cited on page 202).

[524] M. J. S. Dewar and E. F. Healy. “Ground-states of molecules. 64. MNDO calculations for compounds containing bromine”. In: *J. Comp. Chem.* 4 (1983), pages 542–51 (cited on page 202).

[525] M. J. S. Dewar, G. L. Grady, and J. J. P. Stewart. “Ground-states of molecules. 68. MNDO calculations for compounds containing tin”. In: *J. Am. Chem. Soc.* 106 (1984), pages 6771–73 (cited on page 202).

[526] M. J. S. Dewar and et. al. “Ground-states of molecules. 74. MNDO calculations for compounds containing mercury”. In: *Organometallics* 4 (1985), pages 1964–66 (cited on page 202).

[527] M. J. S. Dewar and C. H. Reynolds. “An improved set of MNDO parameters for sulfur”. In: *J. Comp. Chem.* 7 (1986), pages 140–43 (cited on pages 202, 315).

[528] W. D. Cornell, P. Cieplak, C. I. Bayly, I. R. Gould, K. M. Merz Jr., D. M. Ferguson, D. C. Spellmeyer, T. Fox, J. W. Caldwell, and P. A. Kollman. “A second generation force-field for the simulation of proteins, nucleic-acids, and organic-molecules”. In: *J. Am. Chem. Soc.* 117 (1995), pages 5179–97 (cited on page 203).

[529] A. K. Rappé, L. M. Bormann-Rochotte, D. C. Wiser, J. R. Hart, M. A. Pietsch, C. J. Casewit, and W. M. Skiff. “APT: A next generation QM-based reactive force field model”. In: *Mol. Phys.* 105 (2007), page 301 (cited on page 204).

[530] A. K. Rappé and W. A. Goddard III. “Charge equilibration for molecular-dynamics simulations”. In: *J. Phys. Chem.* 95 (1991), pages 3358–63 (cited on page 204).

[531] C. Møller and M. S. Plesset. “Note on an approximation treatment for many-electron systems”. In: *Phys. Rev.* 46 (1934), pages 0618–22 (cited on page 226).

[532] M. J. Frisch, M. Head-Gordon, and J. A. Pople. “Direct MP2 gradient method”. In: *Chem. Phys. Lett.* 166 (1990), pages 275–80 (cited on pages 226, 227).

[533] M. J. Frisch, M. Head-Gordon, and J. A. Pople. “Semi-direct algorithms for the MP2 energy and gradient”. In: *Chem. Phys. Lett.* 166 (1990), pages 281–89 (cited on pages 226, 227).

[534] M. Head-Gordon, J. A. Pople, and M. J. Frisch. “MP2 energy evaluation by direct methods”. In: *Chem. Phys. Lett.* 153 (1988), pages 503–06 (cited on page 226).

[535] S. Saebø and J. Almlöf. “Avoiding the integral storage bottleneck in LCAO calculations of electron correlation”. In: *Chem. Phys. Lett.* 154 (1989), pages 83–89 (cited on page 226).

[536] M. Head-Gordon and T. Head-Gordon. “Analytic MP2 Frequencies Without Fifth Order Storage: Theory and Application to Bifurcated Hydrogen Bonds in the Water Hexamer”. In: *Chem. Phys. Lett.* 220 (1994), pages 122–28 (cited on pages 226, 227).

[537] J. A. Pople, J. S. Binkley, and R. Seeger. “Theoretical Models Incorporating Electron Correlation”. In: *Int. J. Quantum Chem., Suppl.* Y-10 (1976), pages 1–19 (cited on page 226).

[538] K. Raghavachari and J. A. Pople. “Approximate 4th-order perturbation-theory of electron correlation energy”. In: *Int. J. Quantum Chem.* 14 (1978), pages 91–100 (cited on page 226).

[539] K. Raghavachari, M. J. Frisch, and J. A. Pople. “Contribution of triple substitutions to the electron correlation energy in fourth-order perturbation theory”. In: *J. Chem. Phys.* 72 (1980), pages 4244–45 (cited on page 226).

[540] J. A. Pople, K. Raghavachari, H. B. Schlegel, and J. S. Binkley. “Derivative Studies in Hartree-Fock and Møller-Plesset Theories”. In: *Int. J. Quantum Chem., Quant. Chem. Symp.* S13 (1979), pages 225–41 (cited on page 227).

[541] G. W. Trucks, J. D. Watts, E. A. Salter, and R. J. Bartlett. “Analytical MBPT(4) Gradients”. In: *Chem. Phys. Lett.* 153 (1988), pages 490–95 (cited on page 227).

[542] G. W. Trucks, E. A. Salter, C. Sosa, and R. J. Bartlett. “Theory and Implementation of the MBPT Density Matrix: An Application to One-Electron Properties”. In: *Chem. Phys. Lett.* 147 (1988), pages 359–66 (cited on page 227).

[543] P. J. Knowles, J. S. Andrews, R. D. Amos, N. C. Handy, and J. A. Pople. “Restricted Møller-Plesset theory for open shell molecules”. In: *Chem. Phys. Lett.* 186 (1991), pages 130–36 (cited on pages 227, 330).

[544] W. J. Lauderdale, J. F. Stanton, J. Gauss, J. D. Watts, and R. J. Bartlett. “Many-body perturbation theory with a restricted open-shell Hartree-Fock reference”. In: *Chem. Phys. Lett.* 187 (1991), pages 21–28 (cited on page 227).

[545] W. J. Lauderdale, J. F. Stanton, J. Gauss, J. D. Watts, and R. J. Bartlett. “Restricted open-shell Hartree-Fock based many-body perturbation theory: Theory and application of energy and gradient calculations”. In: *J. Chem. Phys.* 97 (1992), page 6606 (cited on page 227).

[546] S. Grimme. “Semiempirical hybrid density functional with perturbative second-order correlation”. In: *J. Chem. Phys.* 124 (2006), page 034108 (cited on page 227).

[547] T. Schwabe and S. Grimme. “Towards chemical accuracy for the thermodynamics of large molecules: new hybrid density functionals including non-local correlation effects”. In: *Phys. Chem. Chem. Phys.* 8 (2006), page 4398 (cited on page 227).

[548] T. Schwabe and S. Grimme. “Double-hybrid density functionals with long-range dispersion corrections: higher accuracy and extended applicability”. In: *Phys. Chem. Chem. Phys.* 9 (2007), page 3397 (cited on page 227).

[549] S. Kozuch and J. M. L. Martin. “DSD-PBEP86: In search of the best double-hybrid DFT with spin-component scaled MP2 and dispersion corrections”. In: *Phys. Chem. Chem. Phys.* 13 (2011), pages 20104–20107 (cited on page 227).

[550] É. Brémond and C. Adamo. “Seeking for parameter-free double-hybrid functionals: The PBE0-DH model”. In: *J. Chem. Phys.* 135 (2011), page 024106 (cited on page 227).

[551] É. Brémond, J. C. Sancho-García, Á. J. Pérez-Jiménez, and C. Adamo. “Communication: Double-hybrid functionals from adiabatic-connection: The QIDH model”. In: *J. Chem. Phys.* 141 (2014), page 031101 (cited on page 227).

[552] J. Gauss. “Calculation of NMR chemical shifts at second-order many-body perturbation theory using gauge-including atomic orbitals”. In: *Chem. Phys. Lett.* 191 (1992), pages 614–20 (cited on page 228).

[553] J. Gauss. “Effects of Electron Correlation in the Calculation of Nuclear- Magnetic-Resonance Chemical-Shifts”. In: *J. Chem. Phys.* 99 (1993), pages 3629–43 (cited on page 228).

[554] J. Gauss. “Accurate Calculation of NMR Chemical-Shifts”. In: *Phys. Chem. Chem. Phys.* 99 (1995), pages 1001–08 (cited on page 228).

[555] J. R. Cheeseman, G. W. Trucks, T. A. Keith, and M. J. Frisch. “A Comparison of Models for Calculating Nuclear Magnetic Resonance Shielding Tensors”. In: *J. Chem. Phys.* 104 (1996), pages 5497–509 (cited on page 228).

[556] T. A. Keith and R. F. W. Bader. “Calculation of magnetic response properties using atoms in molecules”. In: *Chem. Phys. Lett.* 194 (1992), pages 1–8 (cited on pages 228, 229).

[557] T. A. Keith and R. F. W. Bader. “Calculation of magnetic response properties using a continuous set of gauge transformations”. In: *Chem. Phys. Lett.* 210 (1993), pages 223–31 (cited on pages 228, 229).

[558] F. London. “The quantic theory of inter-atomic currents in aromatic combinations”. In: *J. Phys. Radium* 8 (1937), pages 397–409 (cited on page 228).

[559] R. Ditchfield. “Self-consistent perturbation theory of diamagnetism. 1. Gauge-invariant LCAO method for N.M.R. chemical shifts”. In: *Mol. Phys.* 27 (1974), pages 789–807 (cited on page 228).

[560] K. Wolinski, J. F. Hilton, and P. Pulay. “Efficient Implementation of the Gauge-Independent Atomic Orbital Method for NMR Chemical Shift Calculations”. In: *J. Am. Chem. Soc.* 112 (1990), pages 8251–60 (cited on page 228).

[561] K. Ruud, T. Helgaker, K. L. Bak, P. Jørgensen, and H. J. A. Jensen. “Hartree-Fock Limit Magnetizabilities from London Orbitals”. In: *J. Chem. Phys.* 99 (1993), pages 3847–59 (cited on page 228).

[562] T. Helgaker, M. Watson, and N. C. Handy. “Analytical calculation of nuclear magnetic resonance indirect spin-spin coupling constants at the generalized gradient approximation and hybrid levels of density-functional theory”. In: *J. Chem. Phys.* 113 (2000), pages 9402–09 (cited on page 229).

[563] V. Sychrovsky, J. Gräfenstein, and D. Cremer. “Nuclear magnetic resonance spin-spin coupling constants from coupled perturbed density functional theory”. In: *J. Chem. Phys.* 113 (2000), pages 3530–47 (cited on page 229).

[564] V. Barone, J. E. Peralta, R. H. Contreras, and J. P. Snyder. “DFT Calculation of NMR JFF Spin-Spin Coupling Constants in Fluorinated Pyridines”. In: *J. Phys. Chem. A* 106 (2002), pages 5607–12 (cited on page 229).

[565] J. E. Peralta, G. E. Scuseria, J. R. Cheeseman, and M. J. Frisch. “Basis set dependence of NMR Spin-Spin Couplings in Density Functional Theory Calculations: First row and hydrogen atoms”. In: *Chem. Phys. Lett.* 375 (2003), pages 452–58 (cited on page 229).

[566] W. Deng, J. R. Cheeseman, and M. J. Frisch. “Calculation of Nuclear Spin-Spin Coupling Constants of Molecules with First and Second Row Atoms in Study of Basis Set Dependence”. In: *J. Chem. Theory and Comput.* 2 (2006), pages 1028–37 (cited on page 229).

[567] S. Dapprich, I. Komáromi, K. S. Byun, K. Morokuma, and M. J. Frisch. “A New ONIOM Implementation in Gaussian 98. 1. The Calculation of Energies, Gradients and Vibrational Frequencies and Electric Field Derivatives”. In: *J. Mol. Struct. (Theochem)* 462 (1999), pages 1–21 (cited on pages 231, 233).

[568] T. Vreven, K. S. Byun, I. Komáromi, S. Dapprich, J. A. Montgomery Jr., K. Morokuma, and M. J. Frisch. “Combining quantum mechanics methods with molecular mechanics methods in ONIOM”. In: *J. Chem. Theory and Comput.* 2 (2006), pages 815–26 (cited on page 231).

[569] T. Vreven and K. Morokuma. In: *Annual Reports in Computational Chemistry*. Edited by D. C. Spellmeyer. Volume 2. Elsevier, 2006, pages 35–51 (cited on page 231).

[570] F. Clemente, T. Vreven, and M. J. Frisch. In: *Quantum Biochemistry*. Edited by C. Matta. Weinheim: Wiley VCH, 2010, pages 61–84 (cited on pages 231, 232).

[571] T. Vreven and K. Morokuma. In: (2008). Edited by B. Mennucci and R. Cammi (cited on page 231).

[572] F. Maseras and K. Morokuma. “IMOMM - A new integrated ab-initio plus molecular mechanics geometry optimization scheme of equilibrium structures and transition-states”. In: *J. Comp. Chem.* 16 (1995), pages 1170–79 (cited on page 231).

[573] S. Humbel, S. Sieber, and K. Morokuma. “The IMOMO method: Integration of different levels of molecular orbital approximations for geometry optimization of large systems: Test for n-butane conformation and SN2 reaction: RCl+Cl-”. In: *J. Chem. Phys.* 105 (1996), pages 1959–67 (cited on page 231).

[574] T. Matsubara, S. Sieber, and K. Morokuma. “A Test of the New “Integrated MO + MM” (IMOMM) Method for the Conformational Energy of Ethane and n-Butane”. In: *Int. J. Quantum Chem.* 60 (1996), pages 1101–09 (cited on page 231).

[575] M. Svensson, S. Humbel, R. D. J. Froese, T. Matsubara, S. Sieber, and K. Morokuma. “ONIOM: A multi-layered integrated MO+MM method for geometry optimizations and single point energy predictions. A test for Diels-Alder reactions and Pt(P(t-Bu)3)2+H2 oxidative addition”. In: *J. Phys. Chem.* 100 (1996), pages 19357–63 (cited on page 231).

[576] M. Svensson, S. Humbel, and K. Morokuma. “Energetics using the single point IMOMO (integrated molecular orbital plus molecular orbital) calculations: Choices of computational levels and model system”. In: *J. Chem. Phys.* 105 (1996), pages 3654–61 (cited on page 231).

[577] T. Vreven and K. Morokuma. “On the application of the IMOMO (Integrated Molecular Orbital + Molecular Orbital) method”. In: *J. Comp. Chem.* 21 (2000), pages 1419–32 (cited on page 231).

[578] T. Vreven, K. Morokuma, Ö. Farkas, H. B. Schlegel, and M. J. Frisch. “Geometry optimization with QM/MM, ONIOM and other combined methods. I. Microiterations and constraints”. In: *J. Comp. Chem.* 24 (2003), pages 760–69 (cited on page 231).

[579] T. Vreven, M. J. Frisch, K. N. Kudin, H. B. Schlegel, and K. Morokuma. “Geometry optimization with QM/MM Methods. II. Explicit Quadratic Coupling”. In: *Mol. Phys.* 104 (2006), pages 701–14 (cited on pages 231, 245).

[580] M. Lundberg, T. Kawatsu, T. Vreven, M. J. Frisch, and K. Morokuma. “Transition States in the Protein Environment – ONIOM QM:MM Modeling of Isopenicillin N Synthesis”. In: *J. Chem. Theory and Comput.* 5 (2009), pages 222–34 (cited on page 232).

[581] K. Morokuma, D. G. Musaev, T. Vreven, H. Basch, M. Torrent, and D. V. Khoroshun. “Model Studies of the Structures, Reactivities, and Reaction Mechanisms of Metalloenzymes”. In: *IBM J. Res. Dev.* 45 (2001), pages 367–95 (cited on pages 234, 237).

[582] X. Li and M. J. Frisch. “Energy-represented DIIS within a hybrid geometry optimization method”. In: *J. Chem. Theo. Comput.* 2 (2006), pages 835–39 (cited on page 237).

[583] P. Pulay, G. Fogarasi, F. Pang, and J. E. Boggs. “Systematic ab initio gradient calculation of molecular geometries, force constants, and dipole-moment derivatives”. In: *J. Am. Chem. Soc.* 101 (1979), pages 2550–60 (cited on page 237).

[584] G. Fogarasi, X. Zhou, P. Taylor, and P. Pulay. “The calculation of ab initio molecular geometries: Efficient optimization by natural internal coordinates and empirical correction by offset forces”. In: *J. Am. Chem. Soc.* 114 (1992), pages 8191–201 (cited on page 237).

[585] P. Pulay and G. Fogarasi. “Geometry optimization in redundant internal coordinates”. In: *J. Chem. Phys.* 96 (1992), pages 2856–60 (cited on page 237).

[586] J. Baker. “Techniques for geometry optimization - a comparison of cartesian and natural internal coordinates”. In: *J. Comp. Chem.* 14 (1993), pages 1085–100 (cited on page 237).

[587] C. Peng and H. B. Schlegel. “Combining Synchronous Transit and Quasi-Newton Methods for Finding Transition States”. In: *Israel J. Chem.* 33 (1993), pages 449–54 (cited on page 237).

[588] H. B. Schlegel. “Estimating the Hessian for gradient-type geometry optimizations”. In: *Theor. Chem. Acc.* 66 (1984), pages 333–40 (cited on pages 241, 253).

[589] J. Simons, P. Jørgensen, H. Taylor, and J. Ozment. “Walking on Potential Energy Surfaces”. In: *J. Phys. Chem.* 87 (1983), pages 2745–53 (cited on pages 244, 253).

[590] C. J. Cerjan and W. H. Miller. “On Finding Transition States”. In: *J. Chem. Phys.* 75 (1981), pages 2800–06 (cited on page 244).

[591] A. Banerjee, N. Adams, J. Simons, and R. Shepard. “Search for Stationary Points on Surfaces”. In: *J. Phys. Chem.* 89 (1985), pages 52–57 (cited on pages 244, 253).

[592] H. B. Schlegel. “Optimization of Equilibrium Geometries and Transition Structures”. In: *J. Comp. Chem.* 3 (1982), pages 214–18 (cited on pages 245, 253).

[593] H. B. Schlegel. In: *New Theoretical Concepts for Understanding Organic Reactions*. Edited by J. Bertran and I. G. Csizmadia. Volume 267. NATO-ASI series C. The Netherlands: Kluwer Academic, 1989, pages 33–53 (cited on page 245).

[594] H. B. Schlegel. In: *Modern Electronic Structure Theory*. Edited by D. R. Yarkony. Singapore: World Scientific Publishing, 1995, pages 459–500 (cited on page 245).

[595] P. Y. Ayala and H. B. Schlegel. “A combined method for determining reaction paths, minima and transition state geometries”. In: *J. Chem. Phys.* 107 (1997), pages 375–84 (cited on page 246).

[596] J. T. Golab, D. L. Yeager, and P. Jørgensen. “Proper characterization of MC SCF stationary-points”. In: *Chem. Phys.* 78 (1983), pages 175–99 (cited on pages 247, 253).

[597] R. Fletcher. *Practical Methods of Optimization*. New York: Wiley, 1980 (cited on page 253).

[598] J. M. Bofill. “Updated Hessian matrix and the restricted step method for locating transition structures”. In: *J. Comp. Chem.* 15 (1994), pages 1–11 (cited on page 253).

[599] J. M. Bofill and M. Comajuan. “Analysis of the updated Hessian matrices for locating transition structures”. In: *J. Comp. Chem.* 16 (1995), pages 1326–38 (cited on page 253).

[600] J. Baker. “An algorithm for the location of transition-states”. In: *J. Comp. Chem.* 7 (1986), pages 385–95 (cited on page 253).

[601] J. Baker. “An algorithm for geometry optimization without analytical gradients”. In: *J. Comp. Chem.* 8 (1987), pages 563–74 (cited on page 253).

[602] R. F. Curl Jr. “Relationship between Electron Spin Rotation Coupling Constants and G-Tensor Components”. In: *Mol. Phys.* 9 (1965), page 585 (cited on page 263).

[603] E. Hirota. “High-Resolution Spectroscopy of Transient Molecules”. In: Springer Series in Chemical Physics 40 (1985) (cited on page 263).

[604] I. Mills, T. Cvitaš, K. Homann, N. Kállay, and K. Kuchitsu, editors. *Quantities, Units and Symbols in Physical Chemistry*. 2nd ed. Boca Raton: Blackwell, Oxford; dist. CRC Press, 1993 (cited on page 263).

[605] E. Hirota, J. M. Brown, J. T. Hougen, T. Shida, and N. Hirota. “Symbols for fine and hyperfine-structure parameters”. In: *Pure & Appl. Chem.* 66 (1994), pages 571–76 (cited on page 263).

[606] J. Gauss, K. Ruud, and T. Helgaker. “Perturbation-dependent atomic orbitals for the calculation of spin-rotation constants and rotational g tensors”. In: *J. Chem. Phys.* 105 (1996), pages 2804–12 (cited on page 263).

[607] F. Neese. “Prediction of electron paramagnetic resonance g values using coupled perturbed Hartree-Fock and Kohn-Sham theory”. In: *J. Chem. Phys.* 115 (2001), pages 11080–96 (cited on page 263).

[608] H. M. Pickett. “The fitting and prediction of vibration-rotation spectra with spin interactions”. In: *J. Mol. Spectrosc.* 148 (1991), pages 371–377 (cited on page 263).

[609] V. Barone. “Electronic, vibrational and environmental effects on the hyperfine coupling constants of nitroside radicals. H2NO as a case study”. In: *Chem. Phys. Lett.* 262 (1996), pages 201–06 (cited on pages 263, 277, 278).

[610] N. Rega, M. Cossi, and V. Barone. “Development and validation of reliable quantum mechanical approaches for the study of free radicals in solution”. In: *J. Chem. Phys.* 105 (1996), pages 11060–67 (cited on pages 263, 277, 278).

[611] J. Olsen and P. Jørgensen. “Linear and Nonlinear Response Functions for an Exact State and for an MCSCF State”. In: *J. Chem. Phys.* 82 (1985), pages 3235–64 (cited on page 266).

[612] H. Sekino and R. J. Bartlett. “Frequency-Dependent Nonlinear Optical-Properties of Molecules”. In: *J. Chem. Phys.* 85 (1986), pages 976–89 (cited on page 266).

[613] J. E. Rice, R. D. Amos, S. M. Colwell, N. C. Handy, and J. Sanz. “Frequency-Dependent Hyperpolarizabilities with Application to Formaldehyde and Methyl-Fluoride”. In: *J. Chem. Phys.* 93 (1990), pages 8828–39 (cited on page 266).

[614] J. E. Rice and N. C. Handy. “The Calculation of Frequency-Dependent Polarizabilities as Pseudo-Energy Derivatives”. In: *J. Chem. Phys.* 94 (1991), pages 4959–71 (cited on page 266).

[615] J. E. Rice and N. C. Handy. “The Calculation of Frequency-Dependent Hyperpolarizabilities Including Electron Correlation-Effects”. In: *Int. J. Quantum Chem.* 43 (1992), pages 91–118 (cited on page 266).

[616] L. Z. Rosenfeld. In: *Physik* 52 (1928), page 161 (cited on page 266).

[617] E. U. Condon. “Theories of optical rotatory power”. In: *Rev. Mod. Phys.* 9 (1937), pages 432–57 (cited on page 266).

[618] H. Eyring, J. Walter, and G. E. Kimball. *Quantum Chemistry*. New York: Wiley, 1944 (cited on page 266).

[619] A. D. Buckingham. In: *Advances in Chemical Physics*. Edited by I. Prigogine. Volume 12. New York: Wiley Interscience, 1967, page 107 (cited on page 266).

[620] A. D. Buckingham and G. C. Longuet-Higgins. “Quadrupole Moments of Dipolar Molecules”. In: *Mol. Phys.* 14 (1968), page 63 (cited on page 266).

[621] P. W. Atkins and L. D. Barron. “Rayleigh Scattering of Polarized Photons by Molecules”. In: *Mol. Phys.* 16 (1969), page 453 (cited on page 266).

[622] L. D. Barron and A. D. Buckingham. “Rayleigh and Raman Scattering from Optically Active Molecules”. In: *Mol. Phys.* 20 (1971), page 1111 (cited on page 266).

[623] E. Charney. *The Molecular Basis of Optical Activity*. New York: Wiley, 1979 (cited on page 266).

[624] R. D. Amos. “Electric and Magnetic Properties of CO, HF, HCl and CH3F”. In: *Chem. Phys. Lett.* 87 (1982), pages 23–26 (cited on page 266).

[625] P. Jørgensen, H. J. A. Jensen, and J. Olsen. “Linear Response Calculations for Large-Scale Multiconfiguration Self-Consistent Field Wave-Functions”. In: *J. Chem. Phys.* 89 (1988), pages 3654–61 (cited on page 266).

[626] S. P. Karna and M. Dupuis. “Frequency-Dependent Nonlinear Optical-Properties of Molecules - Formulation and Implementation in the Hondo Program”. In: *J. Comp. Chem.* 12 (1991), pages 487–504 (cited on page 266).

[627] T. B. Pedersen and A. E. Hansen. “Ab initio calculation and display of the rotatory strength tensor in the random phase approximation. Method and model studies”. In: *Chem. Phys. Lett.* 246 (1995), pages 1–8 (cited on pages 266, 267).

[628] R. K. Kondru, P. Wipf, and D. N. Beratan. “Theory-assisted determination of absolute stereochemistry for complex natural products via computation of molar rotation angles”. In: *J. Am. Chem. Soc.* 120 (1998), pages 2204–05 (cited on page 266).

[629] P. J. Stephens, F. J. Devlin, J. R. Cheeseman, and M. J. Frisch. “Calculation of optical rotation using density functional theory”. In: *J. Phys. Chem. A* 105 (2001), pages 5356–71 (cited on page 266).

[630] B. Mennucci, J. Tomasi, R. Cammi, J. R. Cheeseman, M. J. Frisch, F. J. Devlin, S. Gabriel, and P. J. Stephens. “Polarizable continuum model (PCM) calculations of solvent effects on optical rotations of chiral molecules”. In: *J. Phys. Chem. A* 106 (2002), pages 6102–13 (cited on page 266).

[631] K. Ruud and T. Helgaker. “Optical rotation studied by density-functional and coupled-cluster methods”. In: *Chem. Phys. Lett.* 352 (2002), pages 533–39 (cited on page 266).

[632] P. J. Stephens, F. J. Devlin, J. R. Cheeseman, M. J. Frisch, and C. Rosini. “Determination of Absolute Configuration Using Optical Rotation Calculated Using Density Functional Theory”. In: *Org. Lett.* 4 (2002), pages 4595–98 (cited on page 266).

[633] P. J. Stephens, F. J. Devlin, J. R. Cheeseman, M. J. Frisch, O. Bortolini, and P. Besse. “Determination of Absolute Configuration Using Ab Initio Calculation of Optical Rotation”. In: *Chirality* 15 (2003), S57–S64 (cited on page 266).

[634] P. J. Stephens, D. M. McCann, J. R. Cheeseman, and M. J. Frisch. "Determination of absolute configurations of chiral molecules using ab initio time-dependent density functional theory calculations of optical rotation: How reliable are absolute configurations obtained for molecules with small rotations?" In: *Chirality* 17 (2005), S52–S64 (cited on page 266).

[635] S. M. Wilson, K. B. Wiberg, J. R. Cheeseman, M. J. Frisch, and P. H. Vaccaro. "Nonresonant optical activity of isolated organic molecules". In: *J. Phys. Chem. A* 109 (2005), pages 11752–64 (cited on page 266).

[636] J. P. Stephens, J. J. Pan, F. J. Devlin, and J. R. Cheeseman. "Determination of the Absolute Configurations of Natural Products Using TDDFT Optical Rotation Calculations: The Iridoid Oruwacin". In: *J. Natural Prod.* 71 (2008), pages 285–88 (cited on page 266).

[637] J. Cioslowski. "A New Population Analysis Based on Atomic Polar Tensors". In: *J. Am. Chem. Soc.* 111 (1989), pages 8333–36 (cited on page 270).

[638] F. L. Hirshfeld. "Bonded-atom fragments for describing molecular charge densities". In: *Theor. Chem. Acc.* 44 (1977), pages 129–38 (cited on page 271).

[639] J. P. Ritchie. "Electron density distribution analysis for nitromethane, nitromethide, and nitramide". In: *J. Am. Chem. Soc.* 107 (1985), pages 1829–37 (cited on page 271).

[640] J. P. Ritchie and S. M. Bachrach. "Some methods and applications of electron density distribution analysis". In: *J. Comp. Chem.* 8 (1987), pages 499–509 (cited on page 271).

[641] A. V. Marenich, S. V. Jerome, C. J. Cramer, and D. G. Truhlar. "Charge Model 5: An Extension of Hirshfeld Population Analysis for the Accurate Description of Molecular Interactions in Gaseous and Condensed Phases". In: *J. Chem. Theory and Comput.* 8 (2012), page 527 (cited on page 271).

[642] T. Le Bahers, C. Adamo, and I. Ciofini. "A Qualitative Index of Spatial Extent in Charge-Transfer Excitations". In: *J. Chem. Theory Comput.* 7 (2011), pages 2498–2506 (cited on page 272).

[643] C. Adamo, T. Le Bahers, M. Savarese, L. Wilbraham, G. García, R. Fukuda, M. Ehara, N. Rega, and I. Ciofini. "Exploring excited states using Time Dependent Density Functional Theory and density-based indexes". In: *Coordination Chemistry Reviews* 304-305 (2015), pages 166–178 (cited on page 272).

[644] R. L. Martin. "Natural transition orbitals". In: *J. Chem. Phys.* 118 (2003), pages 4775–77 (cited on page 272).

[645] U. C. Singh and P. A. Kollman. "An approach to computing electrostatic charges for molecules". In: *J. Comp. Chem.* 5 (1984), pages 129–45 (cited on page 273).

[646] B. H. Besler, K. M. Merz Jr., and P. A. Kollman. "Atomic charges derived from semiempirical methods". In: *J. Comp. Chem.* 11 (1990), pages 431–39 (cited on page 273).

[647] L. E. Chirlian and M. M. Francl. "Atomic charges derived from electrostatic potentials - a detailed study". In: *J. Comp. Chem.* 8 (1987), pages 894–905 (cited on page 273).

[648] C. M. Breneman and K. B. Wiberg. "Determining atom-centered monopoles from molecular electrostatic potentials - the need for high sampling density in formamide conformational-analysis". In: *J. Comp. Chem.* 11 (1990), pages 361–73 (cited on page 273).

[649] H. Hu, Z. Lu, and W. Yang. "Fitting Molecular Electrostatic Potentials from Quantum Mechanical Calculations". In: *J. Chem. Theory and Comput* 3 (2007), pages 1004–13 (cited on page 273).

[650] J. A. Bohmann, F. Weinhold, and T. C. Farrar. "Natural Chemical Shielding Analysis of Nuclear Magnetic Resonance Shielding Tensors from Gauge-Including Atomic Orbital Calculations". In: *J. Chem. Phys.* 107 (1997), pages 1173–84 (cited on page 274).

[651] B. G. Johnson, P. M. W. Gill, and J. A. Pople. "Computing Molecular Electrostatic Potentials with the PRISM Algorithm". In: *Chem. Phys. Lett.* 206 (1993), pages 239–46 (cited on page 277).

[652] J. Gauss and D. Cremer. “Analytical evaluation of energy gradients in quadratic configuration-interaction theory”. In: *Chem. Phys. Lett.* 150 (1988), pages 280–86 (cited on page 285).

[653] E. A. Salter, G. W. Trucks, and R. J. Bartlett. “Analytic Energy Derivatives in Many-Body Methods. I. First Derivatives”. In: *J. Chem. Phys.* 90 (1989), pages 1752–66 (cited on page 285).

[654] T. J. Lee, A. P. Rendell, and P. R. Taylor. “Comparison of the Quadratic Configuration Interaction and Coupled-Cluster Approaches to Electron Correlation Including the Effect of Triple Excitations”. In: *J. Phys. Chem.* 94 (1990), pages 5463–68 (cited on page 286).

[655] H. Nakatsuji and K. Hirao. “Cluster expansion of the wavefunction: Symmetry-adapted-cluster expansion, its variational determination, and extension of open-shell orbital theory”. In: *J. Chem. Phys.* 68 (1978), pages 2053–65 (cited on page 288).

[656] H. Nakatsuji. “Cluster expansion of the wavefunction: Calculation of electron correlations in ground and excited states by SAC and SAC CI theories”. In: *Chem. Phys. Lett.* 67 (1979), pages 334–42 (cited on page 288).

[657] H. Nakatsuji. “Cluster expansion of the wavefunction: Electron correlations in ground and excited states by SAC (Symmetry-Adapted-Cluster) and SAC CI theories”. In: *Chem. Phys. Lett.* 67 (1979), pages 329–33 (cited on page 288).

[658] H. Nakatsuji. “Description of 2-electron and many-electron processes by the SAC-CI method”. In: *Chem. Phys. Lett.* 177 (1991), pages 331–37 (cited on page 288).

[659] H. Nakatsuji. “Exponentially generated configuration interaction theory. Descriptions of excited, ionized and electron attached states”. In: *J. Chem. Phys.* 94 (1991), pages 6716–27 (cited on page 288).

[660] H. Nakatsuji and M. Ehara. “Symmetry-adapted cluster-configuration interaction method applied to high-spin multiplicity”. In: *J. Chem. Phys.* 98 (1993), pages 7179–84 (cited on page 288).

[661] H. Nakatsuji. In: *Computational Chemistry: Reviews of Current Trends*. Edited by J. Leszczynski. Volume 2. Singapore: World Scientific, 1997, pages 62–124 (cited on page 288).

[662] T. Nakajima and H. Nakatsuji. “Analytical energy gradient of the ground, excited, ionized and electron-attached states calculated by the SAC/SAC-CI method”. In: *Chem. Phys. Lett.* 280 (1997), pages 79–84 (cited on page 288).

[663] H. Nakatsuji. “Dipped adcluster model for chemisorption and catalytic reactions”. In: *Prog. Surf. Sci.* 54 (1997), pages 1–68 (cited on page 288).

[664] J. Hasegawa, K. Ohkawa, and H. Nakatsuji. “Excited States of the Photosynthetic Reaction Center of Rhodopseudomonas Viridis: SAC-CI Study”. In: *J. Phys. Chem. B* 102 (1998), pages 10410–19 (cited on page 288).

[665] J. Hasegawa and H. Nakatsuji. “Mechanism and Unidirectionality of the Electron Transfer in the Photosynthetic Reaction Center of Rhodopseudomonas Viridis: SAC-CI Theoretical Study”. In: *J. Phys. Chem. B* 102 (1998), pages 10420–10430 (cited on page 288).

[666] T. Nakajima and H. Nakatsuji. “Energy gradient method for the ground, excited, ionized, and electron-attached states calculated by the SAC (symmetry-adapted cluster)/SAC-CI (configuration interaction) method”. In: *Chem. Phys.* 242 (1999), pages 177–93 (cited on page 288).

[667] M. Ishida, K. Toyota, M. Ehara, and H. Nakatsuji. “Analytical energy gradients of the excited, ionized and electron-attached states calculated by the SAC-CI general-R method”. In: *Chem. Phys. Lett.* 347 (2001), pages 493–98 (cited on page 288).

[668] M. Ishida, K. Toyota, M. Ehara, and H. Nakatsuji. “Analytical energy gradient of high-spin multiplet state calculated by the SAC-CI method”. In: *Chem. Phys. Lett.* 350 (2001), pages 351–58 (cited on page 288).

[669] M. Ehara, M. Ishida, K. Toyota, and H. Nakatsuji. In: *Reviews in Modern Quantum Chemistry*. Edited by K. D. Sen. Singapore: World Scientific, 2002, page 293 (cited on page 288).

[670] K. Toyota, M. Ehara, and H. Nakatsuji. “Elimination of singularities in molecular orbital derivatives: Minimum orbital-deformation (MOD) method”. In: *Chem. Phys. Lett.* 356 (2002), pages 1–6 (cited on page 288).

[671] K. Toyota, I. Mayumi, M. Ehara, M. J. Frisch, and H. Nakatsuji. “Singularity-free analytical energy gradients for the SAC/SAC-CI method: Coupled perturbed minimum orbital-deformation (CPMOD) approach”. In: *Chem. Phys. Lett.* 367 (2003), pages 730–36 (cited on page 288).

[672] H. Nakatsuji, T. Miyahara, and R. Fukuda. “SAC(symmetry adapted cluster)/SAC-CI(configuration interaction) methodology extended to giant molecular systems: ring molecular crystals”. In: *J. Chem. Phys.* 126 (2007), pages 084104-1-18 (cited on page 288).

[673] R. Fukuda and H. Nakatsuji. “Formulation and implementation of direct algorithm for the symmetry adapted cluster and symmetry adapted cluster-configuration interaction method”. In: *J. Chem. Phys.* 128 (2008), page 094105 (cited on pages 288, 290).

[674] N. Nakatani, J. Hasegawa, and H. Nakatsuji. “Red Light in Chemiluminescence and Yellow-green Light in Bioluminescence: Color-tuning Mechanism of Firefly, Photinus pyralis, studied by the SAC-CI method”. In: *J. Am. Chem. Soc.* 129 (2007), pages 8756–8765 (cited on page 288).

[675] K. Fujimoto, J. Hasegawa, and H. Nakatsuji. “Color Tuning Mechanism of Human Red, Green and Blue Cone Pigments: SAC-CI Theoretical Study”. In: *Bull. Chem. Soc. Japan* 82 (2009), pages 1140–1148 (cited on page 288).

[676] T. Miyahara, H. Nakatsuji, and H. Sugiyama. “Helical Structure and Circular Dichroism Spectra of DNA: A Theoretical Study”. In: *J. Phys. Chem. A.* 117 (2013), page 42 (cited on page 288).

[677] T. Miyahara and H. Nakatsuji. “Conformational Dependence of the Circular Dichroism Spectrum of α-Hydroxyphenylacetic Acid: A ChiraSac Study”. In: *J. Phys. Chem. A* 117 (2013), pages 14065–14074 (cited on page 288).

[678] M. Ehara and J. Hasegawa H. Nakatsuji. “SAC-CI Method Applied to Molecular Spectroscopy”. In: *Theory and Applications of Computational Chemistry: The First 40 Years, A Volume of Technical and Historical Perspectives*. Edited by C. E. Dykstra, G. Frenking, K. S. Kim, and G. E. Scuseria. Oxford: Elsevier, 2005, pages 1099–1141 (cited on page 288).

[679] K. N. Kudin, G. E. Scuseria, and E. Cancès. “A black-box self-consistent field convergence algorithm: One step closer”. In: *J. Chem. Phys.* 116 (2002), pages 8255–61 (cited on page 295).

[680] H. B. Schlegel and J. J. McDouall. In: *Computational Advances in Organic Chemistry*. Edited by C. Ögretir and I. G. Csizmadia. The Netherlands: Kluwer Academic, 1991, pages 167–85 (cited on pages 295, 322).

[681] P. Pulay. “Improved SCF convergence acceleration”. In: *J. Comp. Chem.* 3 (1982), pages 556–60 (cited on page 295).

[682] A. Rabuck and G. E. Scuseria. “Improving self-consistent field convergence by varying occupation numbers”. In: *J. Chem. Phys.* 110 (1999), pages 695–700 (cited on page 295).

[683] G. B. Bacskay. “A Quadratically Convergent Hartree-Fock (QC-SCF) Method. Application to Closed Systems”. In: *Chem. Phys.* 61 (1981), pages 385–404 (cited on page 296).

[684] R. Seeger and J. A. Pople. “Self-Consistent Molecular Orbital Methods. 16. Numerically stable direct energy minimization procedures for solution of Hartree-Fock equations”. In: *J. Chem. Phys.* 65 (1976), pages 265–71 (cited on page 296).

[685] S. Miertuš̌, E. Scrocco, and J. Tomasi. “Electrostatic Interaction of a Solute with a Continuum. A Direct Utilization of ab initio Molecular Potentials for the Prevision of Solvent Effects”. In: *Chem. Phys.* 55 (1981), pages 117–29 (cited on pages 299, 382).

[686] S. Miertuš̌ and J. Tomasi. “Approximate Evaluations of the Electrostatic Free Energy and Internal Energy Changes in Solution Processes”. In: *Chem. Phys.* 65 (1982), pages 239–45 (cited on pages 299, 382).

[687] J. L. Pascual-Ahuir, E. Silla, and I. Tuñón. “GEPOL: An improved description of molecular-surfaces. 3. A new algorithm for the computation of a solvent-excluding surface”. In: *J. Comp. Chem.* 15 (1994), pages 1127–38 (cited on page 299).

[688] M. Cossi, V. Barone, R. Cammi, and J. Tomasi. “Ab initio study of solvated molecules: A new implementation of the polarizable continuum model”. In: *Chem. Phys. Lett.* 255 (1996), pages 327–35 (cited on pages 299, 382).

[689] V. Barone, M. Cossi, and J. Tomasi. “A new definition of cavities for the computation of solvation free energies by the polarizable continuum model”. In: *J. Chem. Phys.* 107 (1997), pages 3210–21 (cited on page 299).

[690] B. Mennucci and J. Tomasi. “Continuum solvation models: A new approach to the problem of solute’s charge distribution and cavity boundaries”. In: *J. Chem. Phys.* 106 (1997), pages 5151–58 (cited on page 299).

[691] V. Barone and M. Cossi. “Quantum calculation of molecular energies and energy gradients in solution by a conductor solvent model”. In: *J. Phys. Chem. A* 102 (1998), pages 1995–2001 (cited on pages 299, 301).

[692] M. Cossi, V. Barone, B. Mennucci, and J. Tomasi. “Ab initio study of ionic solutions by a polarizable continuum dielectric model”. In: *Chem. Phys. Lett.* 286 (1998), pages 253–60 (cited on page 299).

[693] V. Barone, M. Cossi, and J. Tomasi. “Geometry optimization of molecular structures in solution by the polarizable continuum model”. In: *J. Comp. Chem.* 19 (1998), pages 404–17 (cited on page 299).

[694] R. Cammi, B. Mennucci, and J. Tomasi. “Second-order Møller-Plesset analytical derivatives for the polarizable continuum model using the relaxed density approach”. In: *J. Phys. Chem. A* 103 (1999), pages 9100–08 (cited on page 299).

[695] M. Cossi, V. Barone, and M. A. Robb. “A direct procedure for the evaluation of solvent effects in MC-SCF calculations”. In: *J. Chem. Phys.* 111 (1999), pages 5295–302 (cited on pages 299, 300).

[696] J. Tomasi, B. Mennucci, and E. Cancès. “The IEF version of the PCM solvation method: An overview of a new method addressed to study molecular solutes at the QM ab initio level”. In: *J. Mol. Struct. (Theochem)* 464 (1999), pages 211–26 (cited on page 299).

[697] M. Cossi and V. Barone. “Solvent effect on vertical electronic transitions by the polarizable continuum model”. In: *J. Chem. Phys.* 112 (2000), pages 2427–35 (cited on page 299).

[698] M. Cossi and V. Barone. “Time-dependent density functional theory for molecules in liquid solutions”. In: *J. Chem. Phys.* 115 (2001), pages 4708–17 (cited on pages 299, 300).

[699] M. Cossi, N. Rega, G. Scalmani, and V. Barone. “Polarizable dielectric model of solvation with inclusion of charge penetration effects”. In: *J. Chem. Phys.* 114 (2001), pages 5691–701 (cited on page 299).

[700] M. Cossi, G. Scalmani, N. Rega, and V. Barone. “New developments in the polarizable continuum model for quantum mechanical and classical calculations on molecules in solution”. In: *J. Chem. Phys.* 117 (2002), pages 43–54 (cited on pages 299, 382).

[701] G. Scalmani and M. J. Frisch. “Continuous surface charge polarizable continuum models of solvation. I. General formalism”. In: *J. Chem. Phys.* 132 (2010), page 114110 (cited on pages 299, 301).

[702] F. Lipparini, G. Scalmani, B. Mennucci, E. Cances, M. Caricato, and M. J. Frisch. “A variational formulation of the polarizable continuum model”. In: *J. Chem. Phys.* 133 (2010), page 014106 (cited on pages 299, 387).

[703] J. Tomasi, B. Mennucci, and R. Cammi. “Quantum mechanical continuum solvation models”. In: *Chem. Rev.* 105 (2005), pages 2999–3093 (cited on page 299).

[704] D. M. Chipman. “Reaction field treatment of charge penetration”. In: *J. Chem. Phys.* 112 (2000), pages 5558–65 (cited on page 299).

[705] E. Cancès and B. Mennucci. “Comment on “Reaction field treatment of charge penetration””. In: *J. Chem. Phys.* 114 (2001), pages 4744–45 (cited on page 299).

[706] A. V. Marenich, C. J. Cramer, and D. G. Truhlar. “Universal solvation model based on solute electron density and a continuum model of the solvent defined by the bulk dielectric constant and atomic surface tensions”. In: *J. Phys. Chem. B* 113 (2009), pages 6378–96 (cited on pages 299, 301, 311).

[707] J. B. Foresman, T. A. Keith, K. B. Wiberg, J. Snoonian, and M. J. Frisch. “Solvent Effects 5. The Influence of Cavity Shape, Truncation of Electrostatics, and Electron Correlation on ab initio Reaction Field Calculations”. In: *J. Phys. Chem.* 100 (1996), pages 16098–104 (cited on page 299).

[708] J. G. Kirkwood. “Theory of Solutions of Molecules Containing Widely Separated Charges with Special Application to Zwitterions”. In: *J. Chem. Phys.* 2 (1934), page 351 (cited on page 299).

[709] L. Onsager. “Electric Moments of Molecules in Liquids”. In: *J. Am. Chem. Soc.* 58 (1936), pages 1486–93 (cited on page 299).

[710] M. W. Wong, M. J. Frisch, and K. B. Wiberg. “Solvent Effects 1. The Mediation of Electrostatic Effects by Solvents”. In: *J. Am. Chem. Soc.* 113 (1991), pages 4776–82 (cited on page 299).

[711] M. W. Wong, K. B. Wiberg, and M. J. Frisch. “Hartree-Fock Second Derivatives and Electric Field Properties in a Solvent Reaction Field - Theory and Application”. In: *J. Chem. Phys.* 95 (1991), pages 8991–98 (cited on page 299).

[712] M. W. Wong, K. B. Wiberg, and M. J. Frisch. “Solvent Effects 2. Medium Effect on the Structure, Energy, Charge Density, and Vibrational Frequencies of Sulfamic Acid”. In: *J. Am. Chem. Soc.* 114 (1992), pages 523–29 (cited on page 299).

[713] M. W. Wong, K. B. Wiberg, and M. J. Frisch. “Solvent Effects 3. Tautomeric Equilibria of Formamide and 2-Pyridone in the Gas Phase and Solution: An ab initio SCRF Study”. In: *J. Am. Chem. Soc.* 114 (1992), pages 1645–52 (cited on page 299).

[714] D. M. York and M. Karplus. “Smooth solvation potential based on the conductor-like screening model”. In: *J. Phys. Chem. A* 103 (1999), pages 11060–79 (cited on page 299).

[715] R. Improta, V. Barone, G. Scalmani, and M. J. Frisch. “A state-specific polarizable continuum model time dependent density functional method for excited state calculations in solution”. In: *J. Chem. Phys.* 125 (2006), 054103:1–9 (cited on pages 299, 300, 302).

[716] R. Improta, G. Scalmani, M. J. Frisch, and V. Barone. “Toward effective and reliable fluorescence energies in solution by a new State Specific Polarizable Continuum Model Time Dependent Density Functional Theory Approach”. In: *J. Chem. Phys.* 127 (2007), 074504:1–9 (cited on pages 299, 300, 302).

[717] G. Scalmani, M. J. Frisch, B. Mennucci, J. Tomasi, R. Cammi, and V. Barone. “Geometries and properties of excited states in the gas phase and in solution: Theory and application of a time-dependent density functional theory polarizable continuum model”. In: *J. Chem. Phys.* 124 (2006), 094107:1–15 (cited on pages 300, 325).

[718] M. Caricato, B. Mennucci, J. Tomasi, F. Ingrosso, R. Cammi, S. Corni, and G. Scalmani. “Formation and relaxation of excited states in solution: A new time dependent polarizable continuum model based on time dependent density functional theory”. In: *J. Chem. Phys.* 124 (2006), page 124520 (cited on page 302).

[719] T. Vreven, B. Mennucci, C. O. da Silva, K. Morokuma, and J. Tomasi. “The ONIOM-PCM method: Combining the hybrid molecular orbital method and the polarizable continuum model for solvation. Application to the geometry and properties of a merocyanine in solution”. In: *J. Chem. Phys.* 115 (2001), pages 62–72 (cited on page 303).

[720] S. J. Mo, T. Vreven, B. Mennucci, K. Morokuma, and J. Tomasi. “Theoretical study of the SN2 reaction of Cl-(H2O) + CH3Cl using our own N-layered integrated molecular orbital and molecular mechanics polarizable continuum model method (ONIOM-PCM)”. In: *Theor. Chem. Acc.* 111 (2003), pages 154–61 (cited on page 303).

[721] F. Florsi and J. Tomasi. “Evaluation of the dispersion contribution to the solvation energy. A simple computational model in the continuum approximation”. In: *J. Comp. Chem.* 10 (1989), page 616 (cited on page 311).

[722] F. Florsi, J. Tomasi, and J. L. Pascual-Ahuir. “Dispersion and repulsion contributions to the solvation energy: Refinements to a simple computational model in the continuum approximation”. In: *J. Comp. Chem.* 12 (1991), page 784 (cited on page 311).

[723] R. A. Pierotti. “A scaled particle theory of aqueous and nonaqueous solutions”. In: *Chem. Rev.* 76 (1976), page 717 (cited on page 311).

[724] R. Cammi, C. Cappelli, S. Corni, and J. Tomasi. “On the calculation of infrared intensities in solution within the polarizable continuum model”. In: *J. Phys. Chem. A* 104 (2000), pages 9874–79 (cited on page 312).

[725] M. J. Frisch, G. Scalmani, T. Vreven, and G. Zheng. “Analytic second derivatives for semiempirical models based on MNDO”. In: *Mol. Phys.* 107 (2009), pages 881–887 (cited on page 315).

[726] W. Thiel and A. A. Voityuk. “Extension of MNDO to d orbitals: Parameters and results for the second-row elements and for the zinc group”. In: *J. Phys. Chem.* 100 (1996), pages 616–26 (cited on page 315).

[727] W. Thiel and A. A. Voityuk. “Extension of the MNDO formalism to d orbitals: Integral approximations and preliminary numerical results”. In: *Theor. Chem. Acc.* 81 (1992), pages 391–404 (cited on page 315).

[728] M. J. S. Dewar, E. G. Zoebisch, and E. F. Healy. “AM1: A New General Purpose Quantum Mechanical Molecular Model”. In: *J. Am. Chem. Soc.* 107 (1985), pages 3902–09 (cited on page 315).

[729] M. J. S. Dewar, C. Jie, and E. G. Zoebisch. “AM1 calculations for compounds containing boron”. In: *Organometallics* 7 (1988), pages 513–21 (cited on page 315).

[730] M. J. S. Dewar and K. M. Merz Jr. “AM1 parameters for zinc”. In: *Organometallics* 7 (1988), pages 522–24 (cited on page 315).

[731] M. J. S. Dewar and C. Jie. “AM1 parameters for phosphorus”. In: *J. Mol. Struct. (Theochem)* 187 (1989), page 1 (cited on page 315).

[732] M. J. S. Dewar and Y.-C. Yuan. “AM1 parameters for sulfur”. In: *Inorganic Chem.* 29 (1990), pages 3881–90 (cited on page 315).

[733] M. J. S. Dewar and A. J. Holder. “AM1 parameters for aluminum”. In: *Organometallics* 9 (1990), pages 508–11 (cited on page 315).

[734] E. Anders, R. Koch, and P. Freunscht. “Optimization and application of lithium parameters for PM3”. In: *J. Comp. Chem.* 14 (1993), pages 1301–12 (cited on page 315).

[735] J. J. P. Stewart. “Optimization of parameters for semiempirical methods. I. Method”. In: *J. Comp. Chem.* 10 (1989), pages 209–20 (cited on page 315).

[736] J. J. P. Stewart. “Optimization of parameters for semiempirical methods. II. Applications”. In: *J. Comp. Chem.* 10 (1989), pages 221–64 (cited on page 315).

[737] J. J. P. Stewart. “Optimization of parameters for semiempirical methods. V. Modification of NDDO approximations and application to 70 elements”. In: *J. Mol. Model.* 13 (2007), pages 1173–213 (cited on page 315).

[738] I. Tubert-Brohman, C. R. W. Guimarães, and W. L. Jorgensen. “Extension of the PDDG/PM3 Semiempirical Molecular Orbit Method to Sulfur, Silicon, and Phosphorus”. In: *J. Chem. Theory and Comput.* 1 (2005), pages 817–23 (cited on page 315).

[739] M. P. Repasky, J. Chandrasekhar, and W. L. Jorgensen. “PDDG/PM3 and PDDG/MNDO: Improved semiempirical methods”. In: *J. Comp. Chem.* 23 (2002), pages 1601–22 (cited on page 315).

[740] I. Tubert-Brohman, C. R. W. Guimarães, M. P. Repasky, and W. L. Jorgensen. “Extension of the PDDG/PM3 and PDDG/MNDO Semiempirical Molecular Orbital Methods to the Halogens”. In: *J. Comp. Chem.* 25 (2004), pages 138–50 (cited on page 315).

[741] J. Tirado-Rives and W. L. Jorgensen. “Performance of B3LYP density functional methods for a large set of organic molecules”. In: *J. Chem. Theory and Comput.* 4 (2008), pages 297–306 (cited on page 315).

[742] K. Throssel and M. J. Frisch. Evaluation and Improvement of Semi-empirical methods I: PM7R8: A variant of PM7 with numerically stable hydrogen bonding corrections, *in prep.* (cited on page 315).

[743] J. J. P. Stewart. “Optimization of parameters for semiempirical methods VI: more modifications to the NDDO approximations and re-optimization of parameters”. In: *J. Molec. Modeling* 19 (2013), pages 1–32 (cited on page 315).

[744] A. D. Daniels, J. M. Millam, and G. E. Scuseria. “Semiempirical methods with conjugate gradient density matrix search to replace diagonalization for molecular systems containing thousands of atoms”. In: *J. Chem. Phys.* 107 (1997), pages 425–31 (cited on page 322).

[745] R. Seeger and J. A. Pople. “Self-Consistent Molecular Orbital Methods. 28. Constraints and Stability in Hartree-Fock Theory”. In: *J. Chem. Phys.* 66 (1977), pages 3045–50 (cited on page 322).

[746] R. Bauernschmitt and R. Ahlrichs. “Stability analysis for solutions of the closed shell Kohn-Sham equation”. In: *J. Chem. Phys.* 104 (1996), pages 9047–52 (cited on page 322).

[747] P. Cársky and E. Hubak. “Restricted Hartree-Fock and Unrestricted Hartree-Fock as reference states in many-body perturbation-theory: A critical comparison of the two approaches”. In: *Theor. Chem. Acc.* 80 (1991), pages 407–25 (cited on page 322).

[748] R. Bauernschmitt and R. Ahlrichs. “Treatment of electronic excitations within the adiabatic approximation of time dependent density functional theory”. In: *Chem. Phys. Lett.* 256 (1996), pages 454–64 (cited on page 325).

[749] M. E. Casida, C. Jamorski, K. C. Casida, and D. R. Salahub. “Molecular excitation energies to high-lying bound states from time-dependent density-functional response theory: Characterization and correction of the time-dependent local density approximation ionization threshold”. In: *J. Chem. Phys.* 108 (1998), pages 4439–49 (cited on page 325).

[750] R. E. Stratmann, G. E. Scuseria, and M. J. Frisch. “An efficient implementation of time-dependent density-functional theory for the calculation of excitation energies of large molecules”. In: *J. Chem. Phys.* 109 (1998), pages 8218–24 (cited on page 325).

[751] C. Van Caillie and R. D. Amos. “Geometric derivatives of excitation energies using SCF and DFT”. In: *Chem. Phys. Lett.* 308 (1999), pages 249–55 (cited on page 325).

[752] C. Van Caillie and R. D. Amos. “Geometric derivatives of density functional theory excitation energies using gradient-corrected functionals”. In: *Chem. Phys. Lett.* 317 (2000), pages 159–64 (cited on page 325).

[753] F. Furche and R. Ahlrichs. “Adiabatic time-dependent density functional methods for excited state properties”. In: *J. Chem. Phys.* 117 (2002), pages 7433–47 (cited on page 325).

[754] J. Liu and W. Liang. “Analytical Hessian of electronic excited states in time-dependent density functional theory with Tamm-Dancoff approximation”. In: *J. Chem. Phys.* 135 (2011), page 014113 (cited on page 325).

[755] J. Liu and W. Liang. “Analytical approach for the excited-state Hessian in time-dependent density functional theory: formalism, implementation and performance”. In: *J. Chem. Phys.* 135 (2011), page 184111 (cited on page 325).

[756] D. Williams-Young, G. Scalmani, F. Ding, M. J. Frisch, and X. Li. in prep (cited on page 325).

[757] C. Adamo and D. Jacquemin. “The calculations of excited-state properties with Time-Dependent Density Functional Theory”. In: *Chem. Soc. Rev.* 42 (2013), page 845 (cited on page 325).

[758] A. D. Laurent, C. Adamo, and D. Jacquemin. “Dye chemistry with time-dependent density functional theory”. In: *Phys. Chem. Chem. Phys.* 16 (2014), pages 14334–56 (cited on page 325).

[759] F. Trani, G. Scalmani, G. S. Zheng, I. Carnimeo, M. J. Frisch, and V. Barone. “Time-Dependent Density Functional Tight Binding: New Formulation and Benchmark of Excited States”. In: *J. Chem. Theory Comput.* 7 (2011), pages 3304–3313 (cited on page 325).

[760] T. Helgaker and P. Jørgensen. “An Electronic Hamiltonian for Origin Independent Calculations of Magnetic-Properties”. In: *J. Chem. Phys.* 95 (1991), pages 2595–601 (cited on page 325).

[761] K. L. Bak, P. Jørgensen, T. Helgaker, K. Ruud, and H. J. A. Jensen. “Gauge-Origin Independent Multiconfigurational Self-Consistent-Field Theory for Vibrational Circular-Dichroism”. In: *J. Chem. Phys.* 98 (1993), pages 8873–87 (cited on page 325).

[762] K. L. Bak, A. E. Hansen, K. Ruud, T. Helgaker, J. Olsen, and P. Jørgensen. “Ab Initio Calculation of Electronic Circular-Dichroism for trans-Cyclooctene Using London Atomic Orbitals”. In: *Theor. Chem. Acc.* 90 (1995), pages 441–58 (cited on page 325).

[763] J. Olsen, K. L. Bak, K. Ruud, T. Helgaker, and P. Jørgensen. “Orbital Connections for Perturbation-Dependent Basis-Sets”. In: *Theor. Chem. Acc.* 90 (1995), pages 421–39 (cited on page 325).

[764] A. E. Hansen and K. L. Bak. “Ab initio calculations of electronic circular dichroism”. In: *ENANTIOMER* 4 (1999), pages 455–76 (cited on page 325).

[765] J. Autschbach, T. Ziegler, S. J. A. van Gisbergen, and E. J. Baerends. “Chiroptical properties from time-dependent density functional theory. I. Circular dichroism spectra of organic molecules”. In: *J. Chem. Phys.* 116 (2002), pages 6930–40 (cited on page 325).

[766] R. Send and F. Furche. “First-order nonadiabatic couplings from time-dependent hybrid density functional response theory: Consistent formalism, implementation and performance”. In: *J. Chem. Phys.* 132 (2010), page 044107 (cited on page 326).

[767] D. B. Lingerfelt, D. B. Williams-Young, A. Petrone, and X. Li. “Direct ab Initio (Meta-)Surface-Hopping Dynamics”. In: *J. Chem. Theory Comput.* 12 (2016), pages 935–945 (cited on page 326).

[768] W. Liang, S. A. Fischer, M. J. Frisch, and X. Li. “Energy-Specific Linear Response TDHF/TDDFT for Calculating High-Energy Excited States”. In: *J. Chem. Theory Comput.* 7 (2011), pages 3540–3547 (cited on page 326).

[769] P. J. Lestrange, P. D. Nguyen, and X. Li. “Calibration of Energy-Specific TDDFT for Modeling K-edge XAS Spectra of Light Elements”. In: *J. Chem. Theory Comput.* 11 (2015), pages 2994–2999 (cited on page 326).

[770] S. Parthiban and J. M. L. Martin. “Assessment of W1 and W2 theories for the computation of electron affinities, ionization potentials, heats of formation, and proton affinities”. In: *J. Chem. Phys.* 114 (2001), pages 6014–29 (cited on page 330).

[771] E. C. Barnes, G. A. Petersson, J. A. Montgomery Jr., M. J. Frisch, and J. M. L. Martin. “Unrestricted Coupled Cluster and Brueckner Doubles Variations of W1 Theory”. In: *J. Chem. Theor. Comput.* 5 (2009), page 2687 (cited on page 330).

[772] A. J. Austin, M. J. Frisch, J. A. Montgomery Jr., and G. A. Petersson. “An overlap criterion for selection of core orbitals”. In: *Theor. Chem. Acc.* 107 (2002), pages 180–86 (cited on page 332).

[773] J. E. Ridley and M. C. Zerner. “An Intermediate Neglect of Differential Overlap Technique for Spectroscopy: Pyrrole and the Azines”. In: *Theor. Chem. Acc.* 32 (1973), pages 111–34 (cited on page 334).

[774] J. E. Ridley and M. C. Zerner. “Triplet states via Intermediate Neglect of Differential Overlap: Benzene, Pyridine, and Diazines”. In: *Theor. Chem. Acc.* 42 (1976), pages 223–36 (cited on page 334).

[775] A. D. Bacon and M. C. Zerner. “An Intermediate Neglect of Differential Overlap Theory for Transition Metal Complexes: Fe, Co, and Cu Chlorides”. In: *Theor. Chem. Acc.* 53 (1979), pages 21–54 (cited on page 334).

[776] M. C. Zerner, G. H. Lowe, R. F. Kirchner, and U. T. Mueller-Westerhoff. "An Intermediate Neglect of Differential Overlap Technique for Spectroscopy of Transition-Metal Complexes. Ferrocene". In: *J. Am. Chem. Soc.* 102 (1980), pages 589–99 (cited on page 334).

[777] P. Corrêa de Mello, M. Hehenberger, and M. C. Zerner. "Converging SCF Calculations on Excited States". In: *Int. J. Quantum Chem.* 21 (1982), pages 251–59 (cited on page 334).

[778] W. P. Anderson, W. D. Edwards, and M. C. Zerner. "Calculated Spectra of Hydrated Ions of the First Transition-Metal Series". In: *Inorganic Chem.* 25 (1986), pages 2728–32 (cited on page 334).

[779] L. K. Hanson, J. Fajer, M. A. Thompson, and M. C. Zerner. "Electrochromic Effects of Charge Separation in Bacterial Photosynthesis - Theoretical Models". In: *J. Am. Chem. Soc.* 109 (1987), pages 4728–30 (cited on page 334).

[780] M. A. Thompson and M. C. Zerner. "A Theoretical Examination of the Electronic Structure and Spectroscopy of the Photosynthetic Reaction Center from Rhodopseudomonas viridis". In: *J. Am. Chem. Soc.* 113 (1991), pages 8210–15 (cited on page 334).

[781] M. C. Zerner. In: *Reviews of Computational Chemistry*. Edited by K. B. Lipkowitz and D. B. Boyd. Volume 2. New York: VCH Publishing, 1991, pages 313–66 (cited on page 334).

[782] P. M. W. Gill. In: *Advances in Quantum Chemistry*. Volume 25. San Diego, CA: Academic Press, 1994, pages 141–205 (cited on page 363).

[783] M. Dupuis, J. Rys, and H. F. King. "Evaluation of molecular integrals over Gaussian basis functions". In: *J. Chem. Phys.* 65 (1976), pages 111–16 (cited on page 363).

[784] H. F. King and M. Dupuis. "Numerical Integration Using Rys Polynomials". In: *J. Comp. Phys.* 21 (1976), pages 144–65 (cited on page 363).

[785] J. Rys, M. Dupuis, and H. F. King. "Computation of electron repulsion integrals using the Rys quadrature method". In: *J. Comp. Chem.* 4 (1983), pages 154–57 (cited on page 363).

[786] H. B. Schlegel, J. S. Binkley, and J. A. Pople. "First and Second Derivatives of Two Electron Integrals over Cartesian Gaussians using Rys Polynomials". In: *J. Chem. Phys.* 80 (1984), pages 1976–81 (cited on page 363).

[787] R. C. Raffenetti. "Pre-processing Two-Electron Integrals for Efficient Utilization in Many-Electron Self-Consistent Field Calculations". In: *Chem. Phys. Lett.* 20 (1973), pages 335–38 (cited on page 381).

[788] T. A. Halgren and W. N. Lipscomb. "The Synchronous Transit Method for Determining Reaction Pathways and Locating Transition States". In: *Chem. Phys. Lett.* 49 (1977), pages 225–32 (cited on page 381).

[789] R. Fletcher and M. J. D. Powell. "A Rapidly Convergent Descent Method for Minimization". In: *Comput. J.* 6 (1963), pages 163–68 (cited on page 381).

[790] B. A. Murtaugh and R. W. H. Sargent. "Computational Experience with Quadratically Convergent Minimization Methods". In: *Comput. J.* 13 (1970), pages 185–94 (cited on page 381).

[791] P. Császár and P. Pulay. "Geometry optimization by direct inversion in the iterative subspace". In: *J. Mol. Struct. (Theochem)* 114 (1984), pages 31–34 (cited on page 382).

[792] Ö. Farkas. PhD (CsC) thesis, Eötvös Loránd University and Hungarian Academy of Sciences, Budapest, 1995 (in Hungarian). (cited on page 382).

[793] Ö. Farkas and H. B. Schlegel. "Methods for optimizing large molecules. II. Quadratic search". In: *J. Chem. Phys.* 111 (1999), pages 10806–14 (cited on page 382).

[794] Ö. Farkas and H. B. Schlegel. "Methods for geometry optimization of large molecules. I. An $O(N^2)$ algorithm for solving systems of linear equations for the transformation of coordinates and forces". In: *J. Chem. Phys.* 109 (1998), pages 7100–04 (cited on page 382).

[795] M. J. Frisch, M. Head-Gordon, and J. A. Pople. “Direct analytic SCF second derivatives and electric field properties”. In: *Chem. Phys.* 141 (1990), pages 189–96 (cited on page 404).