CHAPTER I - Introduction

Input Philosophy

Input to PC GAMESS may be in upper or lower case. There are three types of input groups in PC GAMESS:

1. A pseudo-namelist, free format, keyword driven group. Almost all input groups fall into this first category.

2. A free format group which does not use keywords. The only examples of this category are $DATA, $ECP, $POINTS, and $STONE.

3. Formatted data. This data is never typed by the user, but rather is generated in the correct format by some earlier PC GAMESS run.

All input groups begin with a $ sign in column 2, followed by a name identifying that group. The group name should be the only item appearing on the input line for any group in category 2 or 3.

All input groups terminate with a $END. For any group in category 2 and 3, the $END must appear beginning in column 2, and thus is the only item on that input line.

Type 1 groups may have keyword input on the same line as the group name, and the $END may appear anywhere.

Because each group has a unique name, the groups may be given in any order desired. In fact, multiple occurrences of category 1 groups are permissible.

* * *

Most of the groups can be omitted if the program defaults are adequate. An exception is $DATA, which is always required. A typical free format $DATA group is

```
 $DATA
STO-3G test case for water
CNV      2

OXYGEN       8.0
    STO  3

HYDROGEN     1.0    -0.758       0.0     0.545
    STO  3

 $END
```

Here, position is important. For example, the atom name must be followed by the nuclear charge and then the x,y,z coordinates. Note that missing values will be read as zero, so that the oxygen is placed at the origin. The zero Y coordinate must be given for the hydrogen, so that the final number is taken as Z.

The free format scanner code used to read $DATA is adapted from the ALIS program, and is described in the documentation for the graphics programs which accompany PC GAMESS. Note that the characters ;>! mean something special to the free format scanner, and so use of these characters in $DATA and $ECP should probably be avoided.

Because the default type of calculation is a single point (geometry) closed shell SCF, the $DATA group shown is the only input required to do a RHF/STO-3G water calculation.

* * *

As mentioned, the most common type of input is a namelist-like, keyword driven, free format group. These groups must begin with the $ sign in column 2, but have no further format restrictions. You are not allowed to abbreviate the keywords, or any string value they might expect. They are terminated by a $END string, appearing anywhere. The groups may extend over more than one physical card. In fact, you can give a particular group more than once, as multiple occurrences will be found and processed. We can rewrite the STO-3G water calculation using the keyword groups $CONTRL and $BASIS as

```
 $CONTRL SCFTYP=RHF RUNTYP=ENERGY $END
 $BASIS  GBASIS=STO NGAUSS=3 $END
 $DATA
STO-3G TEST CASE FOR WATER
Cnv    2

Oxygen       8.0     0.0          0.0     0.0
Hydrogen     1.0    -0.758        0.0     0.545
 $END
```

Keywords may expect logical, integer, floating point, or string values. Group names and keywords never exceed 6 characters. String values assigned to keywords never exceed 8 characters. Spaces or commas may be used to separate items:

```
 $CONTRL MULT=3 SCFTYP=UHF,TIMLIM=30.0 $END
```

Floating point numbers need not include the decimal, and may be given in exponential form, i.e. TIMLIM=30, TIMLIM=3.E1, and TIMLIM=3.0D+01 are all equivalent.

Numerical values follow the FORTRAN variable name convention. All keywords which expect an integer value

begin with the letters I-N, and all keywords which expect a floating point value begin with A-H or O-Z. String or logical keywords may begin with any letter.

Some keyword variables are actually arrays. Array elements are entered by specifying the desired subscript:

```
 $SCF NO(1)=1 NO(2)=1 $END
```

When contiguous array elements are given this may be given in a shorter form:

```
 $SCF NO(1)=1,1 $END
```

When just one value is given to the first element of an array, the subscript may be omitted:

```
 $SCF NO=1 NO(2)=1 $END
```

Logical variables can be .TRUE. or .FALSE. or .T. or .F. The periods are required. Logical variables may also be input as 1 or 0.

The program rewinds the input file before searching for the namelist group it needs. This means that the order in which the namelist groups are given is immaterial, and that comment cards may be placed between namelist groups.

Furthermore, the input file is read all the way through for each free-form namelist so multiple occurrences will be processed, although only the LAST occurrence of a variable will be accepted. Comment fields within a free-form namelist group are turned on and off by an exclamation point (!). Comments may also be placed after the $END's of free format namelist groups. Usually, comments are placed in between groups,

```
 $CONTRL SCFTYP=RHF RUNTYP=GRADIENT $END
--$CONTRL EXETYP=CHECK $END
 $DATA
molecule goes here...
```

The second $CONTRL is not read, because it does not have a blank and a $ in the first two columns. Here a careful user has executed a CHECK job, and is now running the real calculation. The CHECK card is now just a comment line.

* * *

The final form of input is the fixed format group. These groups must be given IN CAPITAL LETTERS only! This includes the beginning $NAME and closing $END cards, as well as the group contents. The formatted groups are $VEC, $HESS, $GRAD, $DIPDR, and $VIB. Each of these is produced by some earlier PC GAMESS run, in exactly the correct format for reuse. Thus, the format by which they are read is not documented in this manual.

* * *

Each group is described in the 'Input Description' chapter. Fixed format groups are indicated as such, and the conditions for which each group is required and/or relevant are stated.

* * *

Input Checking

Because some of the data in the input file may not be processed until well into a lengthy run, a facility to check the validity of the input has been provided. If EXETYP=CHECK is specified in the $CONTRL group, PC GAMESS will run without doing much real work so that all the input sections can be executed and the data checked for correct syntax and validity to the extent possible. The one-electron integrals are evaluated and the distinct row table is generated. Problems involving insufficient memory can be identified at this stage. To help avoid the inadvertent absence of data, which may result in the inappropriate use of default values, PC GAMESS will report the absence of any control group it tries to read in CHECK mode. This is of some value in determining which control groups are applicable to a particular problem.

The use of EXETYP=CHECK is HIGHLY recommended for the initial execution of a new problem.

Restart Capability

The program checks for CPU time, and will stop if time is running short. Restart data are printed and punched out automatically, so the run can be restarted where it left off.

At present all SCF modules will place the current orbitals on the punch file if the maximum number of iterations is reached. These orbitals may be used in conjunction with the GUESS=MOREAD option to restart the iterations where they quit. Also, if the TIMLIM option is used to specify a time limit just slighlty less than the job's batch time limit, PC GAMESS will halt if there is insufficient time to complete another full iteration, and the current orbitals will be punched.

When searching for equilibrium geometries or saddle points, if time runs short, or the maximum number of steps is exceeded, the updated hessian matrix is punched for restart. Optimization runs can also be restarted with the dictionary file. See $STATPT for details.

Force constant matrix runs can be restarted from cards. See the $VIB group for details.

```
*********************************
*                               *
*CHAPTER II - Input Description *
*                               *
*********************************
```

This manual describes the input options to PC GAMESS/ Firefly that are compatible with GAMESS (US) version 6 Jun 1999. Therefore, thus this document is based on the excellent GAMESS (US) documentation files version 6 Jun 1999.

This chapter is written in a reference, rather than tutorial fashion. However, there are frequent reminders that more information can be found on a particular input group, or type of calculation, in the 'Further Information' chapter of this manual.

The order of this chapter is chosen to approximate the order in which most people prepare their input ($CONTRL, $BASIS/$DATA, $GUESS, and so on). After that comes run type related input, then properties input, input for two different solvation models, integral related input, and finally CI/MCSCF input. The next page contains a list of most possible input groups, in the order in which they can be found in this chapter.

name	function

Molecule, basis, wavefunction specification:

$CONTRL	chemical control data
$SYSTEM	computer related control data
$D5	pure spherical functions control
$DFT	DFT control data
$BASIS	basis set
$DATA	molecule, basis set
$ZMAT	coded z-matrix
$LIBE	linear bend data
$SCF	HF-SCF wavefunction control
$SCFMI	SCF-MI input control data
$MP2	2nd order Moller-Plesset
$MP2GRD	MP2 gradient control
$MP3	3rd order Moller-Plesset
$MP4	4th order Moller-Plesset
$GUESS	initial orbital selection
$VEC	orbitals (formatted)

Potential energy surface options:

$STATPT	geometry search control
$TRUDGE	nongradient optimization
$TRURST	restart data for TRUDGE
$FORCE	hessian, normal coordinates
$CPHF	coupled-Hartree-Fock options
$HESS	force constant matrix (formatted)
$GRAD	gradient vector (formatted)

```
$DIPDR   dipole deriv. matrix  (formatted)
$VIB     HESSIAN restart data  (formatted)
$MASS    isotope selection
$IRC     intirisic reaction path
$GRADEX  gradient extremal method
$DRC     dynamic reaction path
$SURF    potential surface scan

Interpretation, properties:

$LOCAL   orbital localization control
$TWOEI   J,K integrals         (formatted)
$ELMOM   electrostatic moments
$ELPOT   electrostatic potential
$ELDENS  electron density
$ELFLDG  electric field/gradient
$POINTS  property calculation points
$GRID    property calculation mesh
$PDC     MEP fitting mesh
$MOLGRF  orbital plots
$STONE   distributed multipole analysis
$MOROKM  Morokuma energy decomposition
$FFCALC  finite field polarizabilities
$TDHF    time dependent HF NLO properties
$TDDFT   time dependent DFT NLO properties
$TDVEC   time-dependent orbitals              (formatted)


Solvation models:

$EFRAG   effective fragment potentials
$FRAGNAME specific named fragment pot.
$FRGRPL  inter-fragment repulsion
$PCM     polarizable continuum model
$PCMCAV  PCM cavity generation
$NEWCAV  PCM escaped charge cavity
$DISBS   PCM dispersion basis set
$DISREP  PCM dispersion/repulsion
$SCRF    self consistent reaction field

Integral and integral modification options:

$ECP     effective core potentials
$EFIELD  external electric field
$INTGRL  format for 2e- integrals
$FMM     quantum fast multipole method
$TRANS   integral transformation

MCSCF and CI wavefunctions, and their properties:

$CIINP   control of CI process
$CIS     CI singles control data
$CISVEC  CIS orbitals              (formatted)
$DET     determinantal full CI for MCSCF
$CIDET   determinantal full CI
$DRT     distinct row table for MCSCF
$CIDRT   distinct row table for CI
$MCSCF   parameters for MCSCF
$MCQDPT  multireference pert. Theory
$XMCQDPT extendend multireference pert. Theory
$MCQFIT  faster (X)MCQDPT code
```

```
$CISORT  integral sorting
$GUGDRT  DRT memory usage control
$GUGEM   Hamiltonian matrix formation
$GUGDIA  Hamiltonian eigenvalues/vectors
$GUGDM   1e- density matrix
$GUGDM2  2e- density matrix
$LAGRAN  CI lagrangian matrix
$TRFDM2  2e- density backtransformation
$TRANST  transition moments, spin-orbit
```

==========

```
$CONTRL group          (optional)

This is a free format group specifying global switches.

SCFTYP              together with MPLEVL or CITYP specifies
                    the wavefunction.  You may choose from

       = RHF        Restricted Hartree Fock calculation
                    (default)

       = UHF        Unrestricted Hartree Fock calculation

       = ROHF       Restricted open shell Hartree-Fock.
                    (high spin, see GVB for low spin)

       = GVB        Generalized valence bond wavefunction
                    or OCBSE type ROHF. (needs $SCF input)

       = MCSCF      Multiconfigurational SCF wavefunction
                    (this requires $DET or $DRT input)

       = NONE       indicates a single point computation,
                    rereading a converged SCF function.
                    This option requires that you select
                    CITYP=GUGA or ALDET, RUNTYP=ENERGY,
                    TRANSITN, or SPINORBT, and GUESS=MOREAD.

MPLEVL =            chooses Moller-Plesset perturbation
                    theory level, after the SCF.
                    See $MP2 and $MCQDPT input groups.
       = 0          skips the MP computation (default)
       = 2          performs a second order energy
                    correction.  MP2 is implemented only
                    for RHF, UHF, ROHF, and MCSCF wave
                    functions.  Gradients are available
                    only for RHF, so for the others you
                    may pick from RUNTYP=ENERGY, TRUDGE,
                    SURFACE, or FFIELD only.
       = 3          performs a third order energy
                    correction.
       = 4          performs a fourth order energy
                    correction.


CITYP  =            chooses CI computation after the SCF,
```

for any SCFTYP except UHF.

= NONE skips the CI. (default)

= CIS single excitations from a SCFTYP=RHF reference, only. This is for excited states, with analytic nuclear gradients available. See the $CIS input group.

= GUGA runs the Unitary Group CI package, which requires $CIDRT input. Gradients are available only for RHF, so for other SCFTYPs, you may choose only RUNTYP=ENERGY, TRUDGE, SURFACE, FFIELD, TRANSITN, or SPINORBT.

= ALDET runs the Ames Laboratory determinant full CI package, requiring $CIDET input. RUNTYP=ENERGY only.

Obviously, at most one of MPLEVL or CITYP may be chosen.

RUNTYP specifies the type of computation, for example at a single geometry point:

= ENERGY Molecular energy. (default)

= GRADIENT Molecular energy plus gradient.

= HESSIAN Molecular energy plus gradient plus second derivatives, including harmonic harmonic vibrational analysis. See the $FORCE and $CPHF input groups.

multiple geometry options:

= OPTIMIZE Optimize the molecular geometry using analytic energy gradients. See $STATPT.

= TRUDGE Non-gradient total energy minimization. See groups $TRUDGE and $TRURST.

= SADPOINT Locate saddle point (transition state). See the $STATPT group.

= IRC Follow intrinsic reaction coordinate. See the $IRC group.

= GRADEXTR Trace gradient extremal. See the $GRADEX group.

= DRC Follow dynamic reaction coordinate. See the $DRC group.

= SURFACE Scan linear cross sections of the potential energy surface. See $SURF.

= RSURFACE Relaxed potential energy scans. See $SURF

single geometry property options:

= PROP Properties will be calculated. A $DATA deck and converged $VEC group should be input. Optionally, orbital localization can be done. See $ELPOT, etc.

= MOROKUMA Performs monomer energy decomposition. See the $MOROKM group.

= TRANSITN Compute radiative transition moment. See the $TRANST group.

= SPINORBT Compute spin-orbit coupling.

```
                See the $TRANST group.
      = FFIELD  applies finite electric fields, most
                commonly to extract polarizabilities.
                See the $FFCALC group.
      = TDHF    analytic computation of time dependent
                polarizabilities.  See the $TDHF group.

    * * * * * * * * * * * * * * * * * * * * * * * * * *
    Note that RUNTYPs involving the energy gradient,
    which are GRADIENT, HESSIAN, OPTIMIZE, SADPOINT,
    IRC, GRADEXTR, and DRC, cannot be used for any
    CI or MP2 computation, except when SCFTYP=RHF.
    * * * * * * * * * * * * * * * * * * * * * * * * * *


EXETYP = RUN      Actually do the run. (default)
       = CHECK    Wavefunction and energy will not be
                  evaluated.  This lets you speedily
                  check input and memory requirements.
                  See the overview section for details.
       = DEBUG    Massive amounts of output are printed,
                  useful only if you hate trees.
       = QFMM     obtain timing statistics of the various
                  QFMM stages during SCF.
       = routine  Maximum output is generated by the
                  routine named.  Check the source for
                  the routines this applies to.


MAXIT  =          Maximum number of SCF iteration cycles.
                  Pertains only to RHF, UHF, ROHF, or
                  GVB runs.  See also MAXIT in $MCSCF.
                  (default = 30)

                * * * * * * *

ICHARG =          Molecular charge.  (default=0, neutral)

MULT   =          Multiplicity of the electronic state
       = 1        singlet (default)
       = 2,3,...  doublet, triplet, and so on.

   ICHARG and MULT are used directly for RHF, UHF, ROHF.
   For GVB, these are implicit in the $SCF input, while
   for MCSCF or CI, these are implicit in $DRT/$CIDRT or
   $DET/$CIDET input.  You must still give them correctly.

                * * * * * * *

ECP    =          effective core potential control.
       = NONE     all electron calculation (default).
       = READ     read the potentials in $ECP group.
       = SBKJC    use Stevens, Basch, Krauss, Jasien,
                  Cundari potentials for all heavy
                  atoms (Li-Rn are available).
       = HW       use Hay, Wadt potentials for all the
                  heavy atoms (Na-Xe are available).

                * * * * * * *
```

* * * the next three control molecular geometry * * *

```
COORD  = choice for molecular geometry in $DATA.
       = UNIQUE    only the symmetry unique atoms will be
                   given, in Cartesian coords (default).
       = HINT      only the symmetry unique atoms will be
                   given, in Hilderbrandt style internals.
       = CART      Cartesian coordinates will be input.
                   Please read the warning just below!!!
       = ZMT       GAUSSIAN style internals will be input.
       = ZMTMPC    MOPAC style internals will be input.
       = FRAGONLY  means no part of the system is treated
                   by ab initio means, hence $DATA is not
                   given.  The system is specified by $EFRAG.
```

Note that the CART, ZMT, ZMTMPC choices require input of all atoms in the molecule. These three also orient the molecule, and then determine which atoms are unique. The reorientation is very likely to change the order of the atoms from what you input. When the point group contains a 3-fold or higher rotation axis, the degenerate moments of inertia often cause problems choosing correct symmetry unique axes, in which case you must use COORD=UNIQUE rather than Z-matrices.

Warning: The reorientation into principal axes is done only for atomic coordinates, and is not applied to the axis dependent data in the following groups: $VEC, $HESS, $GRAD, $DIPDR, $VIB, nor Cartesian coords of effective fragments in $EFRAG. COORD=UNIQUE avoids reorientation, and thus is the safest way to read these.

Note that the choices CART, ZMT, ZMTMPC require the use of a $BASIS group to define the basis set. The first two choices might or might not use $BASIS, as you wish.

```
UNITS  = distance units, any angles must be in degrees.
       = ANGS      Angstroms (default)
       = BOHR      Bohr atomic units
```

```
NZVAR  = 0  Use Cartesian coordinates (default).
       = M  If COORD=ZMT or ZMTMPC and a $ZMAT is not given:
            the internal coordinates will be those defining
            the molecule in $DATA.  In this case, $DATA must
            not contain any dummy atoms.  M is usually 3N-6,
            or 3N-5 for linear.
       = M  For other COORD choices, or if $ZMAT is given:
            the internal coordinates will be those defined
            in $ZMAT.  This allows more sophisticated
            internal coordinate choices.  M is ordinarily
            3N-6 (3N-5), unless $ZMAT has linear bends.
```

NZVAR refers mainly to the coordinates used by OPTIMIZE or SADPOINT runs, but may also print the internal's values for other run types. You can use internals to define the molecule, but Cartesians during optimizations!

```
LOCAL  =          controls orbital localization.
       = NONE     Skip localization (default).
```

= BOYS Do Foster-Boys localization.
= RUEDNBRG Do Edmiston-Ruedenberg localization.
= POP Do Pipek-Mezey population localization.
See the $LOCAL group. Localization
does not work for SCFTYP=GVB or CITYP.

d5 = 0 Requests cartesian basis sets (default)
= 1 Allows the use of pure spherical basis sets.
In this instance, $d5 input group is read in
and parsed. Note that the current implementation
of d5 option is not compatible with non-standard
molecular input frames (custom orientations of axes).

DFTTYP = Functional

pure exchange functionals (no correlation)
LDA exchange:
= SLATER Slater exchange, no correlation
= LSDA the same as SLATER

GGA exchange:
= B88 Becke 1988 exchange, no correlation
= XPW91 Perdew-Wang 1991 exchange, no correlation
= GILL96 Gill 1996 exchange, no correlation
= XPBE96 Perdew-Burke-Ernzerhof 1996 exchange, no correlation
= OPTX Handy-Cohen 2001 OPTX exchange, no correlation

pure correlation functionals (Hartree-Fock exchange)
LDA correlation:
= VWN1RPA Hartree-Fock exchange, VWN formula 1 RPA LDA
correlation
= VWN5 Hartree-Fock exchange, VWN formula 5 LDA correlation
= PW91LDA Hartree-Fock exchange, Perdew-Wang 1991 LDA
correlation

GGA correlation:
= LYP Hartree-Fock exchange, Lee-Yang-Parr 1988 correlation
= CPBE96 Hartree-Fock exchange, Perdew-Burke-Ernzerhof 1996
nonlocal +Perdew-Wang 1991 LDA correlation
= CPW91 Hartree-Fock exchange, Perdew 1991 nonlocal +
Perdew-Wang 1991 LDA correlation

exchange-correlation functionals
= SLYP Slater exchange, Lee-Yang-Parr 1988 correlation
= BLYP Becke 1988 exchange, Lee-Yang-Parr 1988 correlation
= GLYP Gill 1996 exchange, Lee-Yang-Parr 1988 correlation
= XLYP Extended exchange functional (a combination of Slater,
Becke 88, and Perdew-Wang 91 exchange functionals) of
Xu and Goddard III, Lee-Yang-Parr 1988 correlation
= OLYP OPTX exchange, Lee-Yang-Parr 1988 correlation (
= SVWN1RPA Slater exchange, VWN formula 1 RPA correlation
= BVWN1RPA Becke 1988 exchange, VWN formula 1 RPA correlation
= SVWN5 Slater exchange, VWN formula 5 correlation
= BVWN5 Becke 1988 exchange, VWN formula 5 correlation
= PBE96 Perdew-Burke-Ernzerhof 1996 exchange,
Perdew-Burke-Ernzerhof 1996 nonlocal +
Perdew-Wang 1991 LDA correlation
= PBEPW91 Perdew-Burke-Ernzerhof 1996 exchange,

Perdew 1991 nonlocal + Perdew-Wang 1991 LDA
correlation
= PW91 Perdew-Wang 1991 exchange,
Perdew 1991 nonlocal + Perdew-Wang 1991 LDA
correlation

hybrid functionals
= B3LYP1 B3LYP as implemented in NWCHEM and GAUSSIAN 98,
using VWN formula 1 RPA correlation
= B3LYP5 B3LYP as implemented in GAMESS (US),
using VWN formula 5 correlation
= B3LYP either B3LYP1 or B3LYP5 depending on the value of
B3LYP keyword in the $DFT group, see later.
= X3LYP Extended exchange functional (a combination of Slater,
Becke 88, and Perdew-Wang 91 exchange functionals) of
Xu and Goddard III + Hartree-Fock exchange,
= O3LYP1 Slater + OPTX + Hartree-Fock exchange,
VWN formula 1 RPA + Lee-Yang-Parr 1988 correlation
= O3LYP5 Slater + OPTX + Hartree-Fock exchange,
VWN formula 5 + Lee-Yang-Parr 1988 correlation
= O3LYP The synonym of O3LYP5
NOTE: FOR ALL O3LYP FAMILY FUNCTIONALS,
SEE THE DESCRIPTION OF THE O3LYP OPTION BELOW!
= BHHLYP Becke 1988 + Hartree-Fock exchange, Lee-Yang-Parr 1988
correlation
= PBE0 Perdew-Burke-Ernzerhof 1996 + Hartree-Fock exchange,
Perdew-Burke-Ernzerhof 1996 nonlocal +
Perdew-Wang 1991 LDA correlation
= PBE1PW91 Perdew-Burke-Ernzerhof 1996 + Hartree-Fock exchange,
Perdew 1991 nonlocal + Perdew-Wang 1991 LDA
correlation
= B3PW91 Slater + Becke 1988 + Hartree-Fock exchange,
Perdew 1991 nonlocal + Perdew-Wang 1991 LDA
correlation

* * * interfaces to other programs * * *

MOLPLT = flag that produces an input deck for a molecule
drawing program distributed with GAMESS.
(default is .FALSE.)

PLTORB = flag that produces an input deck for an orbital
plotting program distributed with GAMESS.
(default is .FALSE.)

AIMPAC = flag to create an input deck for Bader's atoms
in molecules properties code. (default=.FALSE.)
For information about this program, contact
Richard F.W. Bader
Dept. of Chemistry
McMaster University
Hamilton, Ontario L8S-4M1 Canada
bader@sscvax.cis.mcmaster.ca

RPAC = flag to create the input files for Bouman and
Hansen's RPAC electronic excitation and NMR
shieldings program. RPAC works only with
RHF wavefunctions. (inactive)

```
FRIEND = string to prepare input to other quantum
         programs, choose from
       = HONDO    for HONDO 8.2
       = MELDF    for MELDF
       = GAMESSUK for GAMESS (UK Daresbury version)
       = GAUSSIAN for Gaussian 9x
       = ALL      for all of the above
```

PLTORB, MOLPLT, and AIMPAC decks are written to file PUNCH at the end of the job. The two binary disk files output by RPAC are written at the end of the job. Thus all of these correspond to the final geometry encountered during the job.

In contrast, selecting FRIEND turns the job into a CHECK run only, no matter how you set EXETYP. Thus the geometry is that encountered in $DATA. The input is added to the PUNCH file, and may require some (usually minimal) massaging.

PLTORB and MOLPLT are written even for EXETYP=CHECK. AIMPAC requires at least RUNTYP=PROP. RPAC requires at least RUNTYP=ENERGY, and you must take action to save the binary files AOINTS and WORK15.

* * * computation control switches * * *

For the most part, the default is the only sensible value, and unless you are sure of what you are doing, these probably should not be touched.

```
NPRINT =           Print/punch control flag
                   See also EXETYP for debug info.
                   (options -7 to 5 are primarily debug)
       = -7        Extra printing from Boys localization.
       = -6        debug for geometry searches
       = -5        minimal output
       = -4        print 2e-contribution to gradient.
       = -3        print 1e-contribution to gradient.
       = -2        normal printing, no punch file
       =  1        extra printing for basis,symmetry,ZMAT
       =  2        extra printing for MO guess routines
       =  3        print out property and 1e- integrals
       =  4        print out 2e- integrals
       =  5        print out SCF data for each cycle.
                   (Fock and density matrices, current MOs
       =  6        same as 7, but wider 132 columns output.
                   This option isn't perfect.
       =  7        normal printing and punching (default)
       =  8        more printout than 7. The extra output
                   is (AO) Mulliken and overlap population
                   analysis, eigenvalues, Lagrangians, ...
       =  9        everything in 8 plus Lowdin population
                   analysis, final density matrix.

NOSYM  = 0     the symmetry specified in $DATA is used
               as much as possible in integrals, SCF,
               gradients, etc.  (this is the default)
       = 1     the symmetry specified in the $DATA group
```

is used to build the molecule, then
symmetry is not used again. Some GVB
or MCSCF runs (those without a totally
symmetric charge density) require you
request no symmetry.

INTTYP = POPLE use fast Pople-Hehre routines for sp integral
blocks, and HONDO Rys polynomial code for
all other integrals. (default)
= HONDO use HONDO/Rys integrals for all integrals.
This option produces slightly more accurate
integrals but is also slower.

When diffuse functions are used, the inaccuracy in
Pople/Hehre sp integrals shows up as inaccurate
LCAO coefficients in virtual orbitals. This means
the error in SCF (meaning RHF to MCSCF) energies is
expected to be about 5d-8 Hartree, but the error in
computations that OCCUPY the virtual orbitals may
be much larger. We have seen an energy error of
1d-4 in an MP2 energy when diffuse functions were
used. We recommend that all MP2 or CI jobs with
diffuse functions select INTTYP=HONDO.

FSTINT=.TRUE./.FALSE. Enables (default)/disables the use of the new
direct SCF code. For use with MCSCF algorithms,
either $mcscf SOSCF=.t or FOCAS=.t. may be used,
but fullnr method is not supported.

REORDR=.TRUE./.FALSE. Enables (default)/disables shells reordering
for even better direct SCF performance.

GENCON=.TRUE./.FALSE. Enables (default)/disables the use of the
special version of the fastints code designed
for general contraction (GC) type basis sets.
It is mainly intended to dramatically speedup
calculations involving large GC-type basis sets
like ANO basis sets by Roos et al (the example of pure GC
basis sets), and to some degree cc-pVXZ basis sets (which
are only partially of GC type), and many others. The code
is very efficient, but requires some additional amount of
memory, as well as has minor addition computational
overhead for setup. It can result in slightly different
energies than the standard fastints code using the same
value of ICUT and ITOL parameters, and does not improve
performance for pure segmented contraction basis sets at
all. This is why the gencon code automatically disables
itself if the basis set is not of GC type. It has no
effect at present on QFMM calculations. For use with MCSCF
algorithms, either $mcscf SOSCF=.t or FOCAS=.t. may be
used, but fullnr method is not supported.

NORMF = 0 normalize the basis functions (default)
= 1 no normalization

NORMP = 0 input contraction coefficients refer to
normalized Gaussian primitives. (default)
= 1 the opposite.

```
ITOL   =       primitive cutoff factor (default=20)
       = n     products of primitives whose exponential
               factor is less than 10**(-n) are skipped.

ICUT   = n     integrals less than 10.0**(-n) are not
               saved on disk. (default = 9)

WIDE = True.   Wide format for vectors in $VEC groups for both INPUT
               and PUNCH files Default is .False. .
     = False.  Normal format (which is standard for GAMESS).

            * * * restart options * * *

IREST  =       restart control options
               (for OPTIMIZE run restarts, see $STATPT)
               Note that this option is unreliable!
       = -1    reuse dictionary file from previous run,
               useful with GEOM=DAF and/or GUESS=MOSAVED.
               Otherwise, this option is the same as 0.
       = 0     normal run (default)
       = 1     2e restart (1-e integrals and MOs saved)
       = 2     SCF restart (1-,2-e integrls and MOs saved). Now allows
               one to restart with the AOINTS file only, without
               DICTNRY. 1-e integrals will be recomputed if the old
               DICTNRY file does not exist.
       = 3     1e gradient restart
       = 4     2e gradient restart

GEOM   =       select where to obtain molecular geometry
       = INPUT from $DATA input (default for IREST=0)
       = DAF   read from DICTNRY file (default otherwise)

    As noted in the first chapter, binary file restart is
    not a well tested option!
============================================================

$SYSTEM group         (optional)

    This group provides global control information for
your computer's operation.  This is system related input,
and will not seem particularly chemical to you!

TIMLIM =  time limit, in minutes.  Set to about 95 percent
          of the time limit given to the batch job so that
          PC Gamess/Firefly can stop itself gently.
          (default=600.0)

MWORDS =  the maximum replicated memory which your job can
          use, on every node.  This is given in units of
          1,000,000 words (as oppsed to 1024*1024 words),
          where a word is always a 64 bit quantity.  Most
          systems allocate this memory at run time, but
          some more primitive systems may have an upper
          limit chosen at compile time.  (default=1)
          In case finer control over the memory is needed,
          this value can be given in units of words by
          using the keyword MEMORY instead of MWORDS.

KDIAG  =    diagonalization control switch
       = 0  selects use of very stable and fast diagonalization routine
```

which requires large amount of extra memory. (default)
= -1 selects potentially less stable but even faster diagonalization routine which uses less memory than kdiag=0
= -2 selects a combination of two above mentioned methods which can be more stable than kdiag=-1, is usually as fast as kdiag=-1 but requires as much memory as kdiag=0
= 1 use EVVRSP diagonalization. This may be more accurate than Gamess (US) KDIAG=0.
= 2 use GIVEIS diagonalization (not as fast or reliable as EVVRSP)
= 3 use JACOBI diagonalization (this is the slowest method)

The Fastdiag dynamic library (fastdiag.dll in Windows PC GAMESS distribution and fastdiag.ex in Linux version) contains fast optimized modern algorithms for symmetric matrix diagonalization and inversion and is intended to improve the performance of initial guess generation, DIIS extrapolation, as well as some other computationally-intensive steps. You should put it into the same folder as the PC GAMESS executables (Windows users) and in your working/scratch directory (Linux users). It should be renamed to be all lower-case (Linux users).

Note that fast diagonalization routines uses extra memory which is _not_ taken into account during check runs at present. Thus it is recommended to reserve some additional amount of memory for diagonalization routines. For example, if you have a job with 1000 basis functions, it is a good idea to add ca. 2.5-3MW of memory for diagonalization purposes. Otherwise, the slower built-in routine will be called if the amount of memory is not enough to use the fast routines.

Finally, it should be noted that fastdiag is not compatible with and is not used by the generic Pentium PC GAMESS version.

NOJAC = n never use Jacobi diagonalization for matrices of size n x n and above. Recommended for "large-scale" CI and MCSCF calculations (i.e. systems having large number of basis functions (e.g., 1500) and relatively small active spaces (e.g., 12 electrons/12 orbitals).

FASTF = False. Normal operation .
= True. Turns on the usage of fast (non-fortran) sequential file access routines. This increases the overall performance, and allows PC GAMESS to handle large (> 2Gb) sequential access files under Windows NT.

SAFMEM The size of the safety memory pool (in double precision words), which is used by several parts of PC GAMESS in the case when there is not enough dynamic memory to perform current operation. In some cases, the PC GAMESS tries to use slightly more memory than the amount available. Then, increasing SAFMEM can help.

LDAR The default value of LDAR parameter (the size of direct access file record) for $INTGRL, $GUGDM2, $TRFDM2, $CISORT, and $MCSCF groups. Default is 2045

TRUNCF
= .False. The existing files of non-zero length are not truncated when being (re)opened for writing. This may be useful in some cases, for (almost) all the needed disk space is permanently (pre)allocated at the beginning of your PC GAMESS run, which prevents other disk-intensive programs

from causing PC Gamess/Firefly to die in future if the free disk space is exhausted. It has an effect only if FASTF is set to True.

= .True. The files are truncated while being (re)opened for writing. This is the default behavior, which results in the faster write operations. Default is .True.

WSCTL

= .False. Normal operation. The operating system controls the amount of physical memory (the size of the working set) used by the PC GAMESS.

= .True. The PC GAMESS tries to allocate the working set as large as the amount of memory needed for the current operation. Roughly speaking, the process working set is the amount of physical (not virtual!) memory that is allotted to the process. Thus, this strategy usually considerably reduces paging in the case of memory-demanded jobs. On the other hand, this can leave less physical memory available to other processes and to the operating system itself. See also FAQ for information concerning the possible problems (and how to solve them). It has an effect when running under WinNT only. Default is .True.

MAXWS The upper limit for the size of the PC GAMESS working set, in DP words. The PC GAMESS will never try to allocate working set that is larger than this amount. The value of 0 means the automatic selection of an appropriate limit. Namely, the limit will be set to the value smaller by 16 MB than the amount of the physical memory installed (or to the half of the physical memory, if the latter value is larger. It has an effect only when running under WinNT, and WSCTL is set to True. Default is zero.

BLAS3

= sleep Additional threads that were created by BLAS level 3 routines, are suspended when not in use. (default)

= nosleep Additional threads are permanently active. Use this option only if you encounter some unexpected SMP-related problems with the default settings. It is supported only by SMP-enabled PC GAMESS version running in SMP environment.

DECOMM

= .False. Old-style memory manager behavior. Use this option only if you encounter some unexpected problems with the default settings.

= .True. Turns on the advanced memory management feature. The advanced memory management is enabled by default. It is supported under NT, Win95/98, and OS/2.

FLUSH

= .False. Disables the usage of file cache flush operation. It is recommended to disable flushing only in the two following situations:
if the total size of all PC GAMESS working files is less than the available size of the OS file cache;
if the hardware RAID volume is used as the scratch storage.

= .True. Enables the file cache flush operation. PC GAMESS will flush the cache buffers of its working files into disk, when necessary. File flushing is enabled by default.It is supported under NT, Win95/98, and OS/2.

L2SIZE The size of fast L2 cache per CPU, in Kbytes. This information is used by several parts of the PC GAMESS (e.g., by MP3/MP4 code) in order to select the computational strategy optimal for the specific hardware the PC GAMESS is running on. At present, the default value is 0 for all Pentium-optimized PC GAMESS versions, and 512 for Win32-specific Pentium Pro/II/III-optimized version. Change the defaults appropriately for your hardware configuration.

$D5 group (optional)

d5=.true.(default)/.false.
f7=.true.(default)/.false.
g9=.true.(default)/.false.

The default values of d5, f7,and g9 keywords of the $d5 group are all .true., i.e., to use pure spherical d, f, and g functions. The $d5 group has no effect if d5 flag of the $contrl group is not set, which is the default settings. Thus if you prefer to usecartesian basis set, you shouldn't do anything special. If you want to use pure spherical basis, set $contrl d5=1 $end. If you like to use Cartesian d functions but spherical f and g harmonics, you should specify:

```
$contrl d5=.t. $end
$d5 d5=.f. $end
```

$DFT group (optional)

NRAD (integer) the default number of radial points per atom used. Default value is 63.

NRDATM (integer array, dimension 128) An array to change the actual value of NRAD on per element number basis. For example, setting NRDATM(6)=128 will apply the radial grid of 128 points to all carbon atoms. The entry NRDATM(128) is used for dummy atoms with no charge. The default values are all equal to NRAD.

LMAX (integer) the default order of Lebedev angular grid to be used. The default value is 29. Valid values are:

LMAX	Number of points per radial shell
3	6
5	14
7	26

9	38
11	50
13	74
15	86
17	110
19	146
21	170
23	194
25	230
27	266
29	302
31	350
35	434
41	590
47	770
53	974
59	1202
65	1454
71	1730
77	2030
83	2354
89	2702
95	3074
101	3470
107	3890
113	4334
119	4802
125	5294
131	5810

LMXATM (integer array, dimension 128) An array to change the actual value of LMAX on per element number basis. For example, setting NRDATM(1)=25 will apply the angular grid of 25th order to all hydrogen atoms. The entry LMXATM(128) is used for dummy atoms with no charge. The default values are all equal to LMAX.

ANGPRN (logical) flag to activate the angular grid pruning as function of radius, uses the scheme similar to proposed by Murray, Handy and Laming (MHL).
Default is .true. Setting it to false will slow down the calculations but will result to the more accurate results.

KAP (double precision) The parameter Ktheta used for angular pruning. The default value is 5.0 as recommended by MHL. Increasing this value to, say, 10.0 will improve precision by the cost of performance. This parameter has no effect if angular pruning is not used.

RADPRN (logical) flag to activate the radial grid pruning.
Default is .true. Setting it to false will slow down calculations to some degree but may improve precision.
Note that to get correct results you must explicitly disable radial pruning if your system has any basisless atoms!

RMXATM (integer array, dimension 128). An array to change the atomic cutoff radii used during radial pruning on per element number basis. For example, setting RMXATM(8)=6 will set the effective radius of all oxygen atoms to 6 Angstrom. The entry RMXATM(128) is used for dummy atoms with no charge. The default values are probably much larger than is actually necessary and depend on the basis set used.

CUTOFF (double precision). Contributions to the DFT Fock matrix due to batch of angular points, which are smaller than CUTOFF, are ignored. The default value is 1.0d-10.

CUTAO (double precision). If the absolute numerical value of AO is smaller than CUTAO, it will be set to zero during calculations. The default value is 1.0d-10.

CUTWGT (double precision). If the absolute value of weight associated with grid point is less than CUTWGT, this point will not be taken into account during DFT calculations. The default value is 1.0d-20.

CUTORB (double precision). Contributions to the DFT Fock matrix due to batch of orbitals, which are smaller than CUTORB, are ignored. The default value is 1.0d-15.

CUTGG1,
CUTGG2,
CUTGG3,
CUTGG4 (double precision). Various cutoffs used during calculation of grid weights derivatives contributions to the molecular gradients. The default values are 1.0d-13, 1.0d-13, 1.0d-13, and 1.0d-30 correspondingly and are probably too strict.

B3LYP (symbolic). Selects the default implementation of B3LYP functional. The default value is NWCHEM which makes B3LYP to be synonym of B3LYP1. Setting this value to GAMESS will change B3LYP to be the same as B3LYP5.

O3LYP (symbolic). Setting it to GAUSSIAN changes the weight of non-local exchange in all O3LYP functionals so that the resulting functional will be identical to one used in Gaussian 03 Rev D.01 and above. The default value is DEFAULT which makes code compatible with O3LYP implementations in other programs. Note there are some ambiguities in O3LYP-related papers which do not allow to decide which implementation (default or Gaussian) is correct. These two implementations should be considered as just two different xc functionals.

Below is the sample input file.

```
! GAMESS (US) STYLE B3LYP OPTIMIZATION+HESSIAN CALCULATION OF WATER MOLECULE
 $CONTRL SCFTYP=RHF DFTTYP=B3LYP5 runtyp=optimize $END
 $SYSTEM TIMLIM=3000 MEMORY=3000000 $END
 $BASIS  GBASIS=n31 ngauss=6 NDFUNC=1 $END
 $statpt hssend=.t. nprt=-2 $end
 $force nvib=2 $end
 $DATA
H2O
CNV 2

 O           8.0   0.0000000000   0.0000000000   0.7205815395
 H           1.0   0.0000000000   0.7565140024   0.1397092302
 $END
```

```
============================================================

$BASIS group          (optional)

    This group allows certain standard basis sets to be
easily given.  If this group is omitted, the basis set
must be given instead in the $DATA group.

GBASIS =        Name of the Gaussian basis set.
       = MINI - Huzinaga's 3 gaussian minimal basis set.
                Available H-Rn.
       = MIDI - Huzinaga's 21 split valence basis set.
                Available H-Rn.
       = STO  - Pople's STO-NG minimal basis set.
                Available H-Xe, for NGAUSS=2,3,4,5,6.
       = N21  - Pople's N-21G split valence basis set.
                Available H-Xe, for NGAUSS=3.
                Available H-Ar, for NGAUSS=6.
       = N31  - Pople's N-31G split valence basis set.
                Available H-Ne,P-Cl for NGAUSS=4.
                Available H-He,C-F for NGAUSS=5.
                Available H-Ar, for NGAUSS=6.
                For Ga-Kr, N31 selects the BC basis.
       = N311 - Pople's "triple split" N-311G basis set.
                Available H-Ne, for NGAUSS=6.
                Selecting N311 implies MC for Na-Ar.
       = DZV  - "double zeta valence" basis set.
                a synonym for DH for H,Li,Be-Ne,Al-Cl.
                (14s,9p,3d)/[5s,3p,1d] for K-Ca.
                (14s,11p,5d/[6s,4p,1d] for Ga-Kr.
       = DH   - Dunning/Hay "double zeta" basis set.
                (3s)/[2s] for H.
                (9s,4p)/[3s,2p] for Li.
                (9s,5p)/[3s,2p] for Be-Ne.
                (11s,7p)/[6s,4p] for Al-Cl.
       = TZV  - "triple zeta valence" basis set.
                (5s)/[3s] for H.
                (10s,3p)/[4s,3p] for Li.
                (10s,6p)/[5s,3p] for Be-Ne.
                a synonym for MC for Na-Ar.
                (14s,9p)/[8s,4p] for K-Ca.
                (14s,11p,6d)/[10s,8p,3d] for Sc-Zn.
       = MC   - McLean/Chandler "triple split" basis.
                (12s,9p)/[6s,5p] for Na-Ar.
                Selecting MC implies 6-311G for H-Ne.


     * * * the next two are ECP bases only * * *

GBASIS = SBKJC- Stevens/Basch/Krauss/Jasien/Cundari
                valence basis set, for Li-Rn.  This choice
                implies an unscaled -31G basis for H-He.
       = HW   - Hay/Wadt valence basis.
                This is a -21 split, available Na-Xe,
                except for the transition metals.
                This implies a 3-21G basis for H-Ne.

     * * * semiempirical basis sets * * *

          The elements for which these exist can be found
          in the 'Further information' chapter of this
```

manual. If you pick one of these, all other data in this group is ignored. Semi-empirical runs actually use valence-only STO bases, not GTOs.

GBASIS = MNDO - selects MNDO model hamiltonian

= AM1 - selects AM1 model hamiltonian

= PM3 - selects PM3 model hamiltonian

NGAUSS = the number of Gaussians (N). This parameter pertains only to GBASIS=STO, N21, N31, or N311.

NDFUNC = number of heavy atom polarization functions to be used. These are usually d functions, except for MINI/MIDI. The term "heavy" means Na on up when GBASIS=STO, HW, or N21, and from Li on up otherwise. The value may not exceed 3. The variable POLAR selects the actual exponents to be used, see also SPLIT2 and SPLIT3. (default=0)

NFFUNC = number of heavy atom f type polarization functions to be used on Li-Cl. This may only be input as 0 or 1. (default=0)

NPFUNC = number of light atom, p type polarization functions to be used on H-He. This may not exceed 3, see also POLAR. (default=0)

DIFFSP = flag to add diffuse sp (L) shell to heavy atoms. Heavy means Li-F, Na-Cl, Ga-Br, In-I, Tl-At. The default is .FALSE.

DIFFS = flag to add diffuse s shell to hydrogens. The default is .FALSE.

Warning: if you use diffuse functions, please read INTTYP in the $CONTRL group for numerical concerns.

EXTFIL = a flag to read basis sets from an external file, defined by GBASIS=name, rather than from a $DATA group. (default=.false.)

The GBASIS “name” keyword must be an 8 or less character string, obviously not using any internally stored names. Every atom must be defined in the external file by a line giving the chemical symbol, and this chosen string. Following this header line, give the basis in free format $DATA style, containing only S, P, D, F, G, and L shells, and terminating each atom by the usual blank line. The external file may have several families of bases in the same file, identified by different GBASIS strings.
By default, PC GAMESS looks for the file called “basis name”.lib in the directory where input file resides. If the -b < path to basisfile > option is given, then it looks for file called "path to basisfile" first. If it does not exist, it then tries to find file called basis.lib in the directory "path to basisfile".

POLAR = exponent of polarization functions

= POPLE (default for all other cases)

= POPN311 (default for GBASIS=N311, MC)
= DUNNING (default for GBASIS=DH, DZV)
= HUZINAGA (default for GBASIS=MINI, MIDI)
= HONDO7 (default for GBASIS=TZV)

SPLIT2 = an array of splitting factors used when NDFUNC or NPFUNC is 2. Default=2.0,0.5

SPLIT3 = an array of splitting factors used when NDFUNC or NPFUNC is 3. Default=4.00,1.00,0.25

==========

The splitting factors are from the Pople school, and are probably too far apart. See for example the Binning and Curtiss paper. For example, the SPLIT2 value will usually cause an INCREASE over the 1d energy at the HF level for hydrocarbons.

The actual exponents used for polarization functions, as well as for diffuse sp or s shells, are described in the 'Further information' section of this manual. This section also describes the sp part of the basis set chosen by GBASIS fully, with all references cited.

Note that PC Gamess/Firefly always punches a full $DATA group. Thus, if $BASIS does not quite cover the basis you want, you obtain this full $DATA group from EXETYP=CHECK, and then can change polarization exponents, add Rydbergs, etc.

==========

$DATA group (required)

This group describes the global molecular data such as point group symmetry, nuclear coordinates, and possibly the basis set. It consists of a series of free format card images.

-1- TITLE a single descriptive title card.

-2- GROUP, NAXIS

GROUP is the Schoenflies symbol of the symmetry group, you may choose from
C1, CS, CI, CN, S2N, CNH, CNV, DN, DNH, DND,
T, TH, TD, O, OH.

NAXIS is the order of the highest rotation axis, and must be given when the name of the group contains an N. For example, "Cnv 2" is C2v. "S2n 3" means S6.

For linear molecules, choose either CNV or DNH, and enter NAXIS as 4. Enter atoms as DNH with NAXIS=2. If the electronic state of either is degenerate, check the note

about the effect of symmetry in the electronic state
in the Further information chapter.

In order to use PC Gamess/Firefly effectively, you must be able
to recognize the point group name for your molecule. This
presupposes a knowledge of group theory at about the level
of Cotton's "Group Theory", Chapter 3.

Armed with only the name of the group, PC Gamess/Firefly is able
to exploit the molecular symmetry throughout almost all of
the program, and thus save a great deal of computer time.
PC Gamess/Firefly does not require that you know very much else about
group theory, although a deeper knowledge (character
tables, irreducible representations, term symbols, and so
on) is useful when dealing with the more sophisticated
wavefunctions.

Cards -3- and -4- are quite complicated, and are rarely
given. A *SINGLE* blank card may replace both cards -3-
and -4-, to select the 'master frame', which is defined on
the next page. If you choose to enter a blank card, skip
to the bottom of the next page.

Note!
If the point group is C1 (no symmetry), skip over cards
-3- and -4- (which means no blank card).

-3- X1, Y1, Z1, X2, Y2, Z2

For C1 group, there is no card -3- or -4-.
For CI group, give one point, the center of inversion.
For CS group, any two points in the symmetry plane.
For axial groups, any two points on the principal axis.
For tetrahedral groups, any two points on a two-fold axis.
For octahedral groups, any two points on a four-fold axis.

-4- X3, Y3, Z3, DIRECT

third point, and a directional parameter.
For CS group, one point of the symmetry plane,
noncollinear with points 1 and 2.
For CI group, there is no card -4-.

For other groups, a generator sigma-v plane (if any) is
the (x,z) plane of the local frame (CNV point groups).

A generator sigma-h plane (if any) is the (x,y) plane of
the local frame (CNH and dihedral groups).

A generator C2 axis (if any) is the x-axis of the local
frame (dihedral groups).

The perpendicular to the principal axis passing through
the third point defines a direction called D1. If

DIRECT='PARALLEL', the x-axis of the local frame coincides with the direction D1. If DIRECT='NORMAL', the x-axis of the local frame is the common perpendicular to D1 and the principal axis, passing through the intersection point of these two lines. Thus D1 coincides in this case with the negative y axis.

The 'master frame' is just a standard orientation for the molecule. By default, the 'master frame' assumes that

1. z is the principal rotation axis (if any),
2. x is a perpendicular two-fold axis (if any),
3. xz is the sigma-v plane (if any), and
4. xy is the sigma-h plane (if any).

Use the lowest number rule that applies to your molecule.

Some examples of these rules:
Ammonia (C3v): the unique H lies in the XZ plane (R1,R3).
Ethane (D3d): the unique H lies in the YZ plane (R1,R2).
Methane (Td): the H lies in the XYZ direction (R2). Since there is more than one 3-fold, R1 does not apply.
HP=O (Cs): the mirror plane is the XY plane (R4).

In general, it is a poor idea to try to reorient the molecule. Certain sections of the program, such as the orbital symmetry assignment, do not know how to deal with cases where the 'master frame' has been changed.

Linear molecules (C4v or D4h) must lie along the z axis, so do not try to reorient linear molecules.

You can use EXETYP=CHECK to quickly find what atoms are generated, and in what order. This is typically necessary in order to use the general $ZMAT coordinates.

* * * *

Depending on your choice for COORD in $CONTROL,

```
if COORD=UNIQUE, follow card sequence U
if COORD=HINT,   follow card sequence U
if COORD=CART,   follow card sequence C
if COORD=ZMT,    follow card sequence G
if COORD=ZMTMPC, follow card sequence M
```

Card sequence U is the only one which allows you to define a completely general basis here in $DATA.

Recall that UNIT in $CONTRL determines the distance units.

-5U- Atom input. Only the symmetry unique atoms are

input, PC Gamess/Firefly will generate the symmetry equivalent atoms
according to the point group selected above.

```
   if COORD=UNIQUE   NAME, ZNUC, X, Y, Z
   ***************
```

```
NAME  = 10 character atomic name, used only for printout.
        Thus you can enter H or Hydrogen, or whatever.
ZNUC  = nuclear charge.  It is the nuclear charge which
        actually defines the atom's identity.
X,Y,Z = Cartesian coordinates.
```

```
   if COORD=HINT
   *************

   NAME,ZNUC,CONX,R,ALPHA,BETA,SIGN,POINT1,POINT2,POINT3
```

```
NAME = 10 character atomic name (used only for print out).
ZNUC = nuclear charge.
CONX = connection type, choose from
  'LC'   linear conn.              'CCPA' central conn.
  'PCC'  planar central conn.             with polar atom
  'NPCC' non-planar central conn.  'TCT'  terminal conn.
  'PTC'  planar terminal conn.            with torsion
R    = connection distance.
ALPHA= first connection angle
BETA = second connection angle
SIGN = connection sign, '+' or '-'
POINT1, POINT2, POINT3 =
     connection points, a serial number of a previously
     input atom, or one of 4 standard points: O,I,J,K
     (origin and unit points on axes of master frame).
     defaults:  POINT1='O', POINT2='I', POINT3='J'
```

ref- R.L. Hilderbrandt, J.Chem.Phys. 51, 1654 (1969).
You cannot understand HINT input without reading this.

Note that if ZNUC is negative, the internally stored
basis for ABS(ZNUC) is placed on this center, but the
calculation uses ZNUC=0 after this. This is useful
for basis set superposition error (BSSE) calculations.

```
* * * If you gave $BASIS, continue entering cards -5U-
      until all the unique atoms have been specified.
      When you are done, enter a " $END " card.
* * * If you did not, enter cards -6U-, -7U-, -8U-.
```

-6U- GBASIS, NGAUSS, (SCALF(i),i=1,4)

GBASIS has exactly the same meaning as in $BASIS. You may
choose from MINI, MIDI, STO, N21, N31, N311, DZV, DH, BC,
TZV, MC, SBKJC, or HW. In addition, you may choose S, P,
D, F, G, or L to enter an explicit basis set. Here, L
means both an s and p shell with a shared exponent.

NGAUSS is the number of Gaussians (N) in the Pople style
basis, or user input general basis. It has meaning only
for GBASIS=STO, N21, N31, or N311, and S,P,D,F,G, or L.

Up to four scale factors may be entered. If omitted, standard values are used. They are not documented as every GBASIS treats these differently. Read the source code if you need to know more. They are seldom given.

---

* * * If GBASIS is not S,P,D,F,G, or L, either add more shells by repeating card -6U-, or go on to -8U-.
* * * If GBASIS=S,P,D,F,G, or L, enter NGAUSS cards -7U-.

-7U- IG, ZETA, C1, C2

IG = a counter, IG takes values 1, 2, ..., NGAUSS.
ZETA = Gaussian exponent of the IG'th primitive.
C1 = Contraction coefficient for S,P,D,F,G shells, and for the s function of L shells.
C2 = Contraction coefficient for the p in L shells.

---

* * * For more shells on this atom, go back to card -6U-.
* * * If there are no more shells, go on to card -8U-.

-8U- A blank card ends the basis set for this atom.

Continue entering atoms with -5U- through -8U- until all are given, then terminate the group with a " $END " card.

--- this is the end of card sequence U ---

COORD=CART input:

-5C- Atom input.

Cartesian coordinates for all atoms must be entered. They may be arbitrarily rotated or translated, but must possess the actual point group symmetry. PC Gamess/Firefly will reorient the molecule into the 'master frame', and determine which atoms are the unique ones. Thus, the final order of the atoms may be different from what you enter here.

NAME, ZNUC, X, Y, Z

NAME = 10 character atomic name, used only for printout. Thus you can enter H or Hydrogen, or whatever.
ZNUC = nuclear charge. It is the nuclear charge which actually defines the atom's identity.
X,Y,Z = Cartesian coordinates.

Continue entering atoms with card -5C- until all are given, and then terminate the group with a " $END " card.

--- this is the end of card sequence C ---

COORD=ZMT input: (GAUSSIAN style internals)

-5G- ATOM

Only the name of the first atom is required.
See -8G- for a description of this information.

-6G- ATOM i1 BLENGTH

Only a name and a bond distance is required for atom 2.
See -8G- for a description of this information.

-7G- ATOM i1 BLENGTH i2 ALPHA

Only a name, distance, and angle are required for atom 3.
See -8G- for a description of this information.

-8G- ATOM i1 BLENGTH i2 ALPHA i3 BETA i4

ATOM is the chemical symbol of this atom. It can be followed by numbers, if desired, for example Si3. The chemical symbol implies the nuclear charge.
i1 defines the connectivity of the following bond.
BLENGTH is the bond length "this atom-atom i1".
i2 defines the connectivity of the following angle.
ALPHA is the angle "this atom-atom i1-atom i2".
i3 defines the connectivity of the following angle.
BETA is either the dihedral angle "this atom-atom i1-atom i2-atom i3", or perhaps a second bond angle "this atom-atom i1-atom i3".
i4 defines the nature of BETA,
If BETA is a dihedral angle, i4=0 (default).
If BETA is a second bond angle, i4=+/-1.
(sign specifies one of two possible directions).

- Repeat -8G- for atoms 4, 5, ...
- The use of ghost atoms is possible, by using X or BQ for the chemical symbol. Ghost atoms preclude the option of an automatic generation of $ZMAT.
- The connectivity i1, i2, i3 may be given as integers, 1, 2, 3, 4, 5,... or as strings which match one of the ATOMs. In this case, numbers must be added to the ATOM strings to ensure uniqueness!

- In -6G- to -8G-, symbolic strings may be given in place of numeric values for BLENGTH, ALPHA, and BETA. The same string may be repeated, which is handy in enforcing symmetry. If the string is preceeded by a minus sign, the numeric value which will be used is the opposite, of course. Any mixture of numeric data and symbols may be given. If any strings were given

in -6G- to -8G-, you must provide cards -9G- and
-10G-, otherwise you may terminate the group now with
a " $END " card.

-9G- A blank line terminates the Z-matrix section.

-10G- STRING VALUE

STRING is a symbolic string used in the Z-matrix.
VALUE is the numeric value to substitute for that string.

Continue entering -10G- until all STRINGs are defined.
Note that any blank card encountered while reading -10G-
will be ignored. PC Gamess/Firefly regards all STRINGs as variables
(constraints are sometimes applied in $STATPT). It is not
necessary to place constraints to preserve point group
symmetry, as PC Gamess/Firefly will never lower the symmetry from
that given at -2-. When you have given all STRINGs a
VALUE, terminate the group with a " $END " card.

--- this is the end of card sequence G ---

* * * *

The documentation for sequence G above and sequence M
below presumes you are reasonably familiar with the input
to GAUSSIAN or MOPAC. It is probably too terse to be
understood very well if you are unfamiliar with these. A
good tutorial on both styles of Z-matrix input can be
found in Tim Clark's book "A Handbook of Computational
Chemistry", published by John Wiley & Sons, 1985.

Both Z-matrix input styles must generate a molecule
which possesses the symmetry you requested at -2-. If
not, your job will be terminated automatically.

COORD=ZMTMPC input: (MOPAC style internals)

-5M- ATOM

Only the name of the first atom is required.
See -8M- for a description of this information.

-6M- ATOM BLENGTH

Only a name and a bond distance is required for atom 2.
See -8M- for a description of this information.

-7M-   ATOM BLENGTH j1 ALPHA j2

Only a bond distance from atom 2, and an angle with repect to atom 1 is required for atom 3. If you prefer to hook atom 3 to atom 1, you must give connectivity as in -8M-. See -8M- for a description of this information.

-8M-   ATOM BLENGTH j1 ALPHA j2 BETA j3 i1 i2 i3

ATOM, BLENGTH, ALPHA, BETA, i1, i2 and i3 are as described at -8G-. However, BLENGTH, ALPHA, and BETA must be given as numerical values only. In addition, BETA is always a dihedral angle. i1, i2, i3 must be integers only.

The j1, j2 and j3 integers, used in MOPAC to signal optimization of parameters, must be supplied but are ignored here. You may give them as 0, for example.

Continue entering atoms 3, 4, 5, ... with -8M- cards until all are given, and then terminate the group by giving a " $END " card.

--- this is the end of card sequence M ---

==========

This is the end of $DATA!

If you have any doubt about what molecule and basis set you are defining, or what order the atoms will be generated in, simply execute an EXETYP=CHECK job to find out!

==========

$ZMAT group (required if NZVAR is nonzero in $CONTRL)

This group lets you define the internal coordinates in which the gradient geometry search is carried out. These need not be the same as the internal coordinates used in $DATA. The coordinates may be simple Z-matrix types, delocalized coordinates, or natural internal coordinates.

You must input a total of M=3N-6 internal coordinates (M=3N-5 for linear molecules). NZVAR in $CONTRL can be less than M IF AND ONLY IF you are using linear bends. It is also possible to input more than M coordinates if they are used to form exactly M linear combinations for new internals. These may be symmetry coordinates or natural internal coordinates. If NZVAR > M, you must input IJS and SIJ below to form M new coordinates. See DECOMP in $FORCE for the only circumstance in which you may enter a larger NZVAR without giving SIJ and IJS.

**** IZMAT defines simple internal coordinates ****

IZMAT is an array of integers defining each coordinate.
The general form for each internal coordinate is
      code number,I,J,K,L,M,N

IZMAT =1 followed by two atom numbers. (I-J bond length)
      =2 followed by three numbers. (I-J-K bond angle)
      =3 followed by four numbers. (dihedral angle)
         Torsion angle between planes I-J-K and J-K-L.
      =4 followed by four atom numbers. (atom-plane)
         Out-of-plane angle from bond I-J to plane J-K-L.
      =5 followed by three numbers. (I-J-K linear bend)
         Counts as 2 coordinates for the degenerate bend,
         normally J is the center atom. See $LIBE.
      =6 followed by five atom numbers. (dihedral angle)
         Dihedral angle between planes I-J-K and K-L-M.
      =7 followed by six atom numbers. (ghost torsion)
         Let A be the midpoint between atoms I and J, and
         B be the midpoint between atoms M and N. This
         coordinate is the dihedral angle A-K-L-B. The
         atoms I,J and/or M,N may be the same atom number.
         (If I=J AND M=N, this is a conventional torsion).
         Examples: N2H4, or, with one common pair, H2POH.

Example - a nonlinear triatomic, atom 2 in the middle:
      $ZMAT IZMAT(1)=1,1,2, 2,1,2,3, 1,2,3 $END
This sets up two bonds and the angle between them.
The blanks between each coordinate definition are
not necessary, but improve readability mightily.

     **** the next define delocalized coordinates ****

DLC   is a flag to request delocalized coordinates.
       (default is .FALSE.)

AUTO   is a flag to generate all redundant coordinates,
       automatically. The DLC space will consist of all
       non-redundant combinations of these which can be
       found. The list of redundant coordinates will
       consist of bonds, angles, and torsions only.
       (default is .FALSE.)

NONVDW is an array of atom pairs which are to be joined
       by a bond, but might be skipped by the routine
       that automatically includes all distances shorter
       than the sum of van der Waals radii. Any angles
       and torsions associated with the new bond(s) are
       also automatically included.

The format for IXZMAT, IRZMAT, IFZMAT is that of IZMAT:

IXZMAT is an extra array of simple internal coordinates
       which you want to have added to the list generated
       by AUTO. Unlike NONVDW, IXZMAT will add only the
       coordinate(s) you specify.

IRZMAT is an array of simple internal coordinates which
       you would like to remove from the AUTO list of
       redundant coordinates. It is sometimes necessary
       to remove a torsion if other torsions around a bond

are being frozen, to obtain a nonsingular G matrix.

IFZMAT is an array of simple internal coordinates which
you would like to freeze. See also FVALUE below.
Note that IFZMAT/FVALUE work only with DLC, see the
IFREEZ option in $STATPT to freeze coordinates if
you wish to freeze simple or natural coordinates.

FVALUE is an array of values to which the internal
coordinates should be constrained. It is not
necessary to input $DATA such that the initial
values match these desired final values, but it is
helpful if the initial values are not too far away.

AUTOFV is a flag to generate fvalue array automatically. Any user input
of fvalue is thus ignored. This option is especially useful for
relaxed PES scans in DLC. Default is .true.

DLCTOL is the threshold used at several stages during DLC generation, as
well as to impose geometrical constrains. Normally you do not
need to alter it, however, in some cases setting it to, say, 1d-7
can help generating required number of linearly independent DLC
coordinates. Default is 1d-5.

IFRZAT is the array of atom numbers you want to freeze during geometry
optimization using DLC. More precisely, the distances between all
chosen atoms will be frozen, however, they will be allowed to
rotate and translate as the single, united group (rigid body).
Currently no more than 16 atoms may be frozen

**** SIJ,IJS define natural internal coordinates ****

SIJ is a transformation matrix of dimension NZVAR x M,
used to transform the NZVAR internal coordinates in
IZMAT into M new internal coordinates. SIJ is a
sparse matrix, so only the non-zero elements are
given, by using the IJS array described below.
The columns of SIJ will be normalized by PC Gamess/Firefly.
(Default: SIJ = I, unit matrix)

IJS is an array of pairs of indices, giving the row and
column index of the entries in SIJ.

example - if the above triatomic is water, using
IJS(1) = 1,1, 3,1, 1,2, 3,2, 2,3
SIJ(1) = 1.0, 1.0, 1.0,-1.0, 1.0

gives the matrix S= 1.0 1.0 0.0
0.0 0.0 1.0
1.0 -1.0 0.0

which defines the symmetric stretch, asymmetric stretch,
and bend of water.

references for natural internal coordinates:
P.Pulay, G.Fogarasi, F.Pang, J.E.Boggs
J.Am.Chem.Soc. 101, 2550-2560(1979)
G.Fogarasi, X.Zhou, P.W.Taylor, P.Pulay
J.Am.Chem.Soc. 114, 8191-8201(1992)
reference for delocalized coordinates:
J.Baker, A. Kessi, B.Delley

J.Chem.Phys. 105, 192-212(1996)

==========

$LIBE group (required if linear bends are used in $ZMAT)

A degenerate linear bend occurs in two orthogonal planes, which are specified with the help of a point A. The first bend occurs in a plane containing the atoms I,J,K and the user input point A. The second bend is in the plane perpendicular to this, and containing I,J,K. One such point must be given for each pair of bends used.

APTS(1)= x1,y1,z1,x2,y2,z2,... for linear bends 1,2,...

Note that each linear bend serves as two coordinates, so that if you enter 2 linear bends (HCCH, for example), the correct value of NZVAR is M-2, where M=3N-6 or 3N-5, as appropriate.

==========

$SCF group relevant if SCFTYP = RHF, UHF, or ROHF, required if SCFTYP = GVB)

This group of parameters provides additional control over the RHF, UHF, ROHF, or GVB SCF steps. It must be used for GVB open shell or perfect pairing wavefunctions.

DIRSCF = a flag to activate a direct SCF calculation, which is implemented for all the Hartree-Fock type wavefunctions: RHF, ROHF, UHF, and GVB. This keyword also selects direct MP2 computation. The default of .FALSE. stores integrals on disk storage for a conventional SCF calculation.

FDIFF = a flag to compute only the change in the Fock matrices since the previous iteration, rather than recomputing all two electron contributions. This saves much CPU time in the later iterations. This pertains only to direct SCF, and has a default of .TRUE. This option is implemented only for the RHF, ROHF, UHF cases.

Cases with many diffuse functions in the basis set sometimes oscillate at the end, rather than converging. Turning this parameter off will normally give convergence.

---- The next flags affect convergence rates.

EXTRAP = controls Pople extrapolation of the Fock matrix.
DAMP = controls Davidson damping of the Fock matrix.
SHIFT = controls level shifting of the Fock matrix.
RSTRCT = controls restriction of orbital interchanges.
DIIS = controls Pulay's DIIS interpolation.
SOSCF = controls second order SCF orbital optimization.
(default=.TRUE. for RHF, Abelian group ROHF, GVB)
(default=.FALSE. for UHF, non-Abelian group ROHF)
DEM = controls direct energy minimization, which is implemented only for RHF.

defaults for	EXTRAP	DAMP	SHIFT	RSTRCT	DIIS	SOSCF
ab initio:	T	F	F	F	T	T/F
semiempirical:	T	F	F	F	F	F

The above parameters are implemented for all SCF wavefunction types, except that DIIS will work for GVB only for those cases with NPAIR=0 or NPAIR=1. If both DIIS and SOSCF are chosen, SOSCF is stronger than DIIS, and so DIIS will not be used.

Once either DIIS or SOSCF are initiated, any other accelerator in effect is put in abeyance.

---- These parameters fine tune the various convergers.

NCONV = n SCF density convergence criteria.
Convergence is reached when the density change between two consecutive SCF cycles is less than 10.0**(-n) in absolute value. One more cycle is executed after reaching convergence. Less accuracy in NCONV gives questionable gradients. (default is n=5, except CI or MP2 gradients n=6)

SOGTOL = second order gradient tolerance. SOSCF will be initiated when the orbital gradient falls below this threshold. (default=0.25 au)

ETHRSH = energy error threshold for initiating DIIS. The DIIS error is the largest element of e=FDS-SDF. Increasing ETHRSH forces DIIS on sooner. (default = 0.5 Hartree)

MAXDII = Maximum size of the DIIS linear equations, so that at most MAXDII-1 Fock matrices are used in the interpolation. (default=10)

DEMCUT = Direct energy minimization will not be done once the density matrix change falls below this threshold. (Default=0.5)

DMPCUT = Damping factor lower bound cutoff. The damping damping factor will not be allowed to drop below this value. (default=0.0)
note: The damping factor need not be zero to achieve valid convergence (see Hsu, Davidson, and Pitzer, J.Chem.Phys., 65, 609 (1976), see especially the section on convergence control), but it should not be astronomical either.

* *
For more info on the convergence methods,
see the 'Further information' section.
* *

----- miscellaneous options -----

UHFNOS = flag controlling generation of the natural orbitals of a UHF function. (default=.FALSE.)

MVOQ = 0 Skip MVO generation (default)
= n Form modified virtual orbitals, using a cation with n electrons removed. Implemented for RHF, ROHF, and GVB. If necessary to reach a closed shell cation, the program might remove n+1 electrons. Typically, n will be about 6.

NPUNCH = SCF punch option
= 0 do not punch out the final orbitals
= 1 punch out the occupied orbitals
= 2 punch out occupied and virtual orbitals
The default is NPUNCH = 2.

----- options for virial scaling -----

VTSCAL = A flag to request that the virial theorem be satisfied. An analysis of the total energy as an exact sum of orbital kinetic energies is printed. The default is .FALSE.

This option is implemented for RHF, UHF, and ROHF, for RUNTYP=ENERGY, OPTIMIZE, or SADPOINT. Related input is as follows:

SCALF = initial exponent scale factor when VTSCAL is in use, useful when restarting. The default is 1.0.

MAXVT = maximum number of iterations (at a single geometry) to satisfy the energy virial theorem. The default is 20.

VTCONV = convergence criterion for the VT, which is satisfied when 2<T> + <V> + R x dE/dR is less than VTCONV. The default is 1.0D-6 Hartree.

For more information on this option, which is most economically employed during a geometry search, see M.Lehd and F.Jensen, J.Comput.Chem. 12, 1089-1096(1991).

The next parameters define the GVB wavefunction. Note that ALPHA and BETA also have meaning for ROHF. See also MULT in the $CONTRL group. The GVB wavefunction assumes orbitals are in the order core, open, pairs.

NCO = The number of closed shell orbitals. The default almost certainly should be changed! (default=0).

NSETO = The number of sets of open shells in the function. Maximum of 10. (default=0)

NO = An array giving the degeneracy of each open shell set. Give NSETO values. (default=0,0,0,...).

NPAIR = The number of geminal pairs in the -GVB- function. Maximum of 12. The default

corresponds to open shell SCF (default=0).

CICOEF = An array of ordered pairs of CI coefficients for the -GVB- pairs. For example, a two pair case for water, say, might be CICOEF(1)=0.95,-0.05,0.95,-0.05. If not normalized, as in the default, they will be. This parameter is useful in restarting a GVB run, with the current CI coefficients. (default = 0.90,-0.20,0.90,-0.20,...)

COUPLE = A switch controlling the input of F, ALPHA, and BETA. The default is to use internally stored values for these variables. Note ALPHA and BETA can be given for -ROHF-, as well as -GVB-. (Default=.FALSE.)

F = An vector of fractional occupations.

ALPHA = An array of A coupling coefficients given in lower triangular order.

BETA = An array of B coupling coefficients given in lower triangular order.

Note: The default for F, ALPHA, and BETA depends on the state chosen. Defaults for the most commonly occuring cases are internally stored.

* * * * * * * * * * * * * * * * * * *
For more discussion of GVB/ROHF input
see the 'Further information' section
* * * * * * * * * * * * * * * * * * *

==========

$SCFMI group (optional, relevant if SCFTYP=RHF)

The SCF-MI method is a modification of the Roothaan equations that avoids basis set superposition error (BSSE) in intermolecular interaction calculations, by expanding each monomer's orbitals using only its own basis set. Thus, the resulting orbitals are not orthogonal. The presence of a $SCFMI group in the input triggers the use of this option.

The implementation is limited to two monomers, treated at the RHF level. The energy, gradient, and therefore numerical hessian are available. The SCF step may be run in direct SCF mode. The first 4 parameters must be given. All atoms of monomer A must be given in $DATA before the atoms of monomer B.

NA = number of doubly occupied MOs on fragment A.
NB = number of doubly occupied MOs on fragment B.
MA = number of basis functions on fragment A.
MB = number of basis functions on fragment B.

```
ITER      = maximum number of SCF-MI cycles, overriding
            the usual MAXIT value.  (default is 50).

DTOL      = SCF-MI density convergence criteria.
            (default is 1.0d-10)

ALPHA     = possible level shift parameter.
            (default is 0.0, meaning shifting is not used)

IOPT      =   prints additional debug information.
          = 0 standard outout (default)
          = 1 print for each SCF-MI cycle MOs, overlap
              between the MOs, CPU times.
          = 2 print some extra informations in secular
              systems solution.

MSHIFT    = debugging option that permits to shift all
            the memory pointer of the SCF-MI section
            of code of the quantity MSHIFT (default is 0).

============================================================

   "Modification of Roothan Equations to Exclude BSSE
       from Molecular Interaction Calculations"
    E. Gianinetti, M. Raimondi, E. Tornaghi
    Int. J. Quantum Chem. 60, 157 (1996)

    A. Famulari, E. Gianinetti, M. Raimondi, and M. Sironi
    Int. J. Quantum Chem. (1997), submitted.

============================================================

$MP2 group   (relevant to SCFTYP=RHF,UHF,ROHF if MPLEVL=2)

     Controls 2nd order Moller-Plesset perturbation runs,
if requested by MPLEVL in $CONTRL.  See also the DIRSCF
keyword in $SCF to select direct MP2.  MP2 is implemented
for RHF, high spin ROHF, or UHF wavefunctions.  Analytic
gradients and the first order correction to the wave-
function (i.e. properties) are available only for RHF.
The $MP2 group is usually not given.  See also $MCQDPT.


METHOD = 2     Selects old style sequential UHF/ROHF MP2 energy integral
               transformation.
       = -2     Activates a new sequential UHF/ROHF MP2 energy integral
               transformation code that requires approximately a half of
               memory used in method 2. Default is -2.
METHOD = 1     Selects the usage of the new RHF/ROHF/UHF MP2 energy
               program. This program is intended to handle large systems
                (e.g., 500 AOs or more). It is direct, very fast, and
               requires much less memory as compared to other MP2
               methods. It is definitely the method that is to be chosen
               for the large jobs. On the other hand, it requires the 2-
               electron AO integrals to be reevaluated four times. Thus,
               there can be a considerable overhead for the small jobs.
               This is the only reason why this method is not turned on
               by default.

NCORE = n  Omits the first n occupied orbitals from the
           calculation.  The default for n is the number
```

of chemical core orbitals.

MP2PRP= a flag to turn on property computation for RHF MP2 jobs with RUNTYP=ENERGY. This is appreciably more expensive than just evaluating the 2nd order energy correction alone, so the default is .FALSE. Properties are always computed during gradient runs, when they are an almost free byproduct.

LMOMP2= a flag to analyze the closed shell MP2 energy in terms of localized orbitals. Any type of localized orbital may be used. This option is implemented only for RHF, and its selection forces use of the METHOD=3 transformation. The default is .FALSE.

OSPT= selects open shell spin-restricted perturbation. This parameter applies only when SCFTYP=ROHF. Please see the 'Further information' chapter for more information about this choice.
= ZAPT picks Z-averaged perturbation theory. (default)
= RMP picks RMP (aka ROHF-MBPT) perturbation theory.

CUTOFF= transformed integral retention threshold, the default is 1.0d-9.

GRDMET = 1 Selects old style MP2 gradient calculations
= 2 Activates the new MP2 gradient code, which is generally much faster. (default)

DM2MET = -3,-2,-1,0,1,2,3 Selects one of seven(!) possible methods of MP2 DM2 calculations during MP2 gradient run. Methods -3, -2, and -1 are precisely the same as methods 3, 2, and 1, respectively, but they use an asynchronous disk I/O to eliminate I/O latency and to reduce the total execution time; for this reason, these methods are not supported under DOS. Method 0 is definitely the slowest one; which one is the fastest depends on the amount of the physical memory installed. Methods 2(-2) require the least CPU time, but probably not the least total time. Methods 3(-3) need the least memory. By default, method 1 is chosen, which seems to be a quite reasonable compromise. You can experiment with this option to find optimal settings for your particular task and environment. Relevant only when GRDMET=2. Default is 1.

MEMGRD = The maximum number of words of the memory to be used in the MP2 gradient calculation. It controls only the last stage of calculations, namely, the evaluation of the gradient integrals themselves. Zero means the usage of all available memory. Relevant only when GRDMET=2. The default value is 0.

MXCPIT = Controls the maximum number of AO CPHF iterations during the MP2 gradient runs. The default value is 100.

CPTOL = Controls the maximum allowed RMS error of the approximate solution of CPHF equations during MP2 gradient runs. The default value is 10-10.

MXRECL = The maximum record size for direct access file used during

the MP2 energy (method=3) and MP2 gradient runs. The default maximum size is 65536 DP words.

IOPARS = Specifies the set of specific I/O optimization flags for the direct access files (DAF) used during MP2/MP2 GRADIENT runs. Actually, this is the 2-digit octal value. The least significant digit controls the general DAF writing operations, while the most significant one controls the behavior of 'READ-THEN-WRITE-TO-THE-SAME-LOCATION' operation. Each digit can have one of the following values:

* 0 - no particular I/O optimization;
* 1 - no particular I/O optimization, file cache write through mode;
* 2 - optimizes file I/O for the sequential access;
* 3 - optimizes file I/O for the sequential access, file cache write through mode;
* 4 - optimizes file I/O for the random access;
* 5 - optimizes file I/O for the random access, file cache write through mode;
* 6 - optimizes file I/O for the random access with some locality;
* 7 - optimizes file I/O for the random access with some locality, file cache write through mode;

The default value for this option is 74 in the case of non-negative DM2MET, and 75 otherwise. If you have a plenty of RAM, or fast RAID, or your job creates DAF of a (very) small size, try to set IOPARS to 64 and see if the I/O speed changes. The optimal value may also depend on the OS used.

DIRECT = .False. Selects the semidirect evaluation of the AO integrals during METHOD=1 MP2 calculations. (default)
= .True. Selects the fully direct evaluation of AO integrals during METHOD=1 MP2 calculations. This somewhat reduces the amount of disk space needed, by the cost of the additional CPU time (which may be considerable). Relevant only for the METHOD=1 case.

PACKAO =..False. Normal operation.
= .True. Activates packing of the AO integrals during METHOD=1 semidirect MP2 calculations, thus (slightly) reducing the amount of disk space needed. Relevant only for the METHOD=1 semidirect case. Default is .True.

MNRECL = The minimum allowed record size (in terms of 12-byte elements) for the direct access file used during METHOD=1 MP2 calculations. Reducing the MNRECL value increases the job execution time to some degree, but decreases the amount of memory needed for calculations. It is not recommended to set MNRECL below 500. Relevant only for the METHOD=1 case. Default is 20000.

SVDISK = .False. This allows the METHOD=1 ROHF MP2 program to use (in some rare cases) additional disk space in order to reduce the job CPU time. (default)
= .True. Disables the extra disk space usage. Relevant only for the METHOD=1 ROHF MP2 case.

SPARSE =.False. This options controls how the matrix-matrix multiplication is performed during the MP2 METHOD=1 runs. False means the usage of standard BLAS level 3 routines (the only strategy, which was implemented in the PC GAMESS 4.4 and earlier). This is the best choice when the list of two-electron integrals is dense enough, i.e., when the total number of nonzero integrals is less than the maximum of N4/8 only several times (up to 3-5). Relevant only for the METHOD=1 MP2 case.

= .True. Selects the usage of special matrix-matrix multiplication routines. This strategy is optimal for sparse lists of two-electron integrals. By default, the PC GAMESS tries to choose the most appropriate strategy automatically. Note, the program choice is in some situations not the best one. This is especially true for SMP systems, or for the systems with the slow access to the main memory

NWORD = controls memory usage. The default uses all available memory. (default=0)

METHOD= n selects transformation method, 2 being the segmented transformation, and 3 being a more conventional two phase bin sort implementation. 3 requires more disk, but less memory. The default is to attempt method 2 first, and method 3 second.

AOINTS= defines AO integral storage during conventional integral transformations, during parallel runs.
DUP stores duplicated AO lists on each node, and is the default for parallel computers with slow interprocessor communication, e.g. ethernet.
DIST distributes the AO integral file across all nodes, and is the default for parallel computers with high speed communications.

Main characteristics of the MP2 implementation in PcGamess/Firefly:

Memory requirements scale as approximately N2.
The other MP2 programs currently implemented in the GAMESS scale as at least N3 for the segmented transformation and as A N2 for the alternative integral transformation. Here, A and N are the number of active orbitals and the total number of MOs, respectively.

Disk requirements scale as A2N2. They scale as A2(N-A)2 for the alternative integral transformation. The segmented transformation does not use temporary disk storage. The disk I/O is used, but to a very limited degree. Therefore, the CPU utilization is usually 90% or even better.

The CPU utilization is usually less than 50% for other MP2 transformation methods working in the conventional mode, while, in the direct mode, there is a very serious overhead because of the multiple reevaluation of 2-electron integrals.

Asymptotically, the FLOPs count is about a half or even better as compared to other MP2 energy transformation methods.

It uses SMP more efficiently.

On the other hand, it requires the 2-electron AO integrals to be reevaluated four times. This cost is fixed and does not depend on the details of the MP2 calculation performed.

==========

$MP2GRD group (requested by $MP2 GRDMET=1 or GRDMET=2)

New semidirect MP2 properties/gradient code was developed to speed up MP2 gradient and related jobs, to seriously reduce memory demands and to allow efficient parallel execution. It is based on FASTINTS code and runs in parallel using P2P interface. It has very modest memory demands, namely, asymptotically they are proportional to Nocc ?(Nocc + Ncore) ?Nvir, where Nocc is the number of active (not counting frozen core) occupied orbitals, Ncore is the number of frozen core occupied orbitals, and Nvir is the number of virtual orbitals. Intrinsically, it is very fast and has very good parallel scalability, with most of the memory demands reducing linearly with the number of used computing nodes. At present, it is limited to RMP2 case only, but will be extended for ZAPT2 in the future.

New MP2 gradient code uses disk I/O and (large) temporary files to store half-transformed 2-electron integrals and non-separable part of MP2 2-particle density matrix (DM2) and thus can be disk I/O intensive. When running in parallel, the contents of these files is evenly distributed across all nodes. If one is running this code in parallel on SMP or multicore system, it is recommended to assign a separate physical disk to each of the parallel PC GAMESS processes.

The new code is controlled by two keywords of the $MP2 group and by several additional keywords of $MP2GRD group. New code is triggered on by the presence of METHOD=1 keyword in $MP2 group and its behavior is further controlled by either $MP2 GRDMET=1 $END or $MP2 GRDMET=2 $END (the default) switch. GRDMET=2 uses less memory and is usually faster, but for some combinations of computer systems and particular tasks GRDMET=1 can be the preferred option.

The fine tuning of the new properties/gradient module is possible via the following keywords of the $MP2GRD group:

CACHE (logical, default value depends on the particular task). If turned on, reduces disk reads during second MP2 gradient 2-electron integral half-transformation by using some extra memory.

ASYNC (logical, default is true). If turned on, activates asynchronous disk I/O during 2-e gradient computation by using some extra memory

ASYSNC (logical, default is false). When turned on, activates asynchronous "sync" (e.g., "disk flush") calls.

FUSED (logical, default is true). Changes the ordering of loops so that the disk I/O becomes less irregular.

DBLBF (logical, default is true). Enables double buffering and asynchronous I/O for disk writes during second MP2 gradient 2-electron integral half-transformation. Causes minor memory overhead. See also the discussion of iofflgs(30)=1 option below.

GLBLSN (logical, default is false). Enables global synchronization of disk I/O during second MP2 gradient 2-electron integral half-transformation when running in parallel.

MXI (integer, default is min(Nocc,256)). The maximum number of occupied

orbitals to be processed in one slice during first and second integral half-transformation. Seriously affects amount of memory required by both first and second half-transformation. Cannot be larger than 256 by design.

MXPQ (integer, default is 225). The maximum number of AO orbital pairs to be processed in one slice during second 2-e integral half-transformation and 2-e gradient computation. Seriously affects amount of memory required by both second half-transformation and gradient code itself

RSIZE1 (integer, default value is 32768). Record size (in elements) used to store half-transformed 2-e integrals. Seriously affects amount of memory required by the first half-transformation, as well as disk I/O performance.

RSIZE2 (integer, default value is 32768). Record size (in elements) used to store half-transformed DM2 elements. Seriously affects amount of memory required by the second half-transformation, as well as disk I/O performance.

NBUFS (integer, default value is calculated to be close to the optimum). Number of buffers used for double buffered asynchronous I/O (see DBLBF option).

TOL1 (double precision, default value is 10-9). Threshold to store half-transformed 2-e integrals.

TOL2 (double precision, default value is 10-9). Threshold to store half-transformed DM2 elements.

RANDOM (double precision, default value is 1/3). The random seed to initialize random number generator used while distributing AO shell pairs over nodes.

==========

$MP3 group (relevant to SCFTYP=RHF,UHF,ROHF if MPLEVL=2)

$MP4 group (relevant to SCFTYP=RHF,UHF,ROHF if MPLEVL=2)

NCORE = Omits first NCORE occupied orbitals from the calculation. The default for NCORE is the number of chemical core orbitals.

NWORD = Controls the memory usage during MP3/MP4 calculations.The default value for NWORD is 0 which means that all available memory is used.

CUTOFF = This parameter controls the precision, to which the contributions to the MP3 and MP4(D) energies due to the external exchange operator, are calculated. The default value for CUTOFF is 10-9. For MP3 energy, it is guaranteed that the absolute value of the cumulative error will be less than the value of CUTOFF. For the MP4(D) part, the situation is usually the same, but, unlike MP3 energy, this is not always true, especially for the basis sets with partial linear dependence.
Due to this reason, it is recommended to use stronger CUTOFF (e.g., 10-15), if the high-precision MP4 energy is required.

This will increase the precision by the cost of additional CPU time.

SMPMET = Selects one of the three SMP-based strategies available for the calculation of the external exchange operator contribution to the MP4 energy (this option has no effect in the case of MP3 calculations).
= 1 Method 1 is probably the best one, as it has very good SMP scaling properties. On the other hand, its memory requirements are proportional to the number of CPUs used. (default)
= 2 Method 2 does not require extra memory, but can be used only if the number of CPUs is even. Furtermore, its SMP scaling is generally worse than that of Method 1.
= 3 Method 3 also does not require extra memory. It usually has the worst SMP scaling properties.

SDTQ = .False. Performs MP4-SDQ calculation.
= .True. Performs MP4-SDTQ (i.e., full MP4) calculation. Default is False, i.e., only MP4-SDQ energy is calculated.

==========

$GUESS group (optional, relevant for all SCFTYP's)

This group controls the selection of initial molecular orbitals.

GUESS = Selects type of initial orbital guess.
= HUCKEL Carry out an extended Huckel calculation using a Huzinaga MINI basis set, and project this onto the current basis. This is implemented for atoms up to Rn, and will work for any all electron or ECP basis set. (default for most runs)
= HCORE Diagonalize the one electron Hamiltonian to obtain the initial guess orbitals. This method is applicable to any basis set, but does not work as well as the HUCKEL guess.
= MOREAD Read in formatted vectors punched by an earlier run. This requires a $VEC group, and you MUST pay attention to NORB below.
= MOSAVED (default for restarts) The initial orbitals are read from the DICTNRY file of the earlier run.
= SKIP Bypass initial orbital selection. The initial orbitals and density matrix are assumed to be in the DICTNRY file.

All GUESS types except 'SKIP' permit reordering of the orbitals, carry out an orthonormalization of the orbitals, and generate the correct initial density matrix. The initial density matrix cannot be generated for -CI- and -MCSCF-, so property restarts for these wavefunctions will require 'SKIP' which is an otherwise seldom used option. Note that correct computation of a -GVB- density matrix requires CICOEF in $SCF. Another possible use for 'SKIP' is to speed up a EXETYP=CHECK job, or a RUNTYP=HESSIAN job where the hessian is supplied.

PRTMO = a flag to control printing of the initial guess.
(default=.FALSE.)

$VEC

NORB = The number of orbitals to be read in the $VEC group. This applies only to GUESS=MOREAD.

For -RHF-, -UHF-, -ROHF-, and -GVB-, NORB defaults to the number of occupied orbitals. NORB must be given for -CI- and -MCSCF-. For -UHF-, if NORB is not given, only the occupied alpha and beta orbitals should be given, back to back. Otherwise, both alpha and beta orbitals must consist of NORB vectors.
NORB may be larger than the number of occupied MOs, if you wish to read in the virtual orbitals. If NORB is less than the number of atomic orbitals, the remaining orbitals are generated as the orthogonal complement to those read.

NORDER = Orbital reordering switch.
= 0 No reordering (default)
= 1 Reorder according to IORDER and JORDER.

IORDER = Reordering instructions.
Input to this array gives the new molecular orbital order. For example, IORDER(3)=4,3 will interchange orbitals 3 and 4, while leaving the other MOs in the original order. This parameter applies to all orbitals (alpha and beta) except for -UHF-, where it only affects the alpha MOs. (default is IORDER(i)=i)

JORDER = Reordering instructions.
Same as IORDER, but for the beta MOs of -UHF-.

TOLZ = level below which MO coefficients will be set to zero. (default=0)

TOLE = level at which MO coefficients will be equated. This is a relative level, coefficients are set equal if one agrees in magnitude to TOLE times the other. (default=0)

SYMDEN = call a routine to project the totally symmetric component out of a density matrix. This may be useful if the HUCKEL or HCORE give orbitals with inexact symmetry. Since the density matrix is not idempotent, this can generate a non-variational energy on the first iteration. For the same reason, this should never be used with orbitals of MOREAD quality. (default=.FALSE.)

MIX = rotate the alpha and beta HOMO and LUMO orbitals so as to generate inequivalent alpha and beta orbital spaces. This pertains to UHF singlets only. This may require use of NOSYM=1 in $CONTRL depending on your situation. (default=.FALSE.)

KDIAG = diagonalization control switch

= 0 selects use of very stable and fast diagonalization routine which requires large amount of extra memory.
= -1 selects potentially less stable but even faster diagonalization routine which uses less memory than kdiag=0
= -2 selects a combination of two above mentioned methods which can be more stable than kdiag=-1, is usually as fast as kdiag=-1 but requires as much memory as kdiag=0
= 1 use EVVRSP diagonalization. This may be more accurate than Gamess (US) KDIAG=0.
= 2 use GIVEIS diagonalization (not as fast or reliable as EVVRSP)
= 3 use JACOBI diagonalization (this is the slowest method) (DEFAULT) This default is not sensible, but assures compatibility with Gamess (US). It is generally recommended to set kdiag=0 in the $guess group in all the PC GAMESS input files, especially if the number of basis functions is large enough.

The Fastdiag dynamic library (fastdiag.dll in Windows PC GAMESS distribution and fastdiag.ex in Linux version) contains fast optimized modern algorithms for symmetric matrix diagonalization and inversion and is intended to improve the performance of initial guess generation, DIIS extrapolation, as well as some other computationally-intensive steps. You should put it into the same folder as the PC GAMESS executables (Windows users) and in your working/scratch directory (Linux users). It should be renamed to be all lower-case (Linux users).

Note that fast diagonalization routines uses extra memory which is _not_ taken into account during check runs at present. Thus it is recommended to reserve some additional amount of memory for diagonalization routines. For example, if you have a job with 1000 basis functions, it is a good idea to add ca. 2.5-3MW of memory for diagonalization purposes. Otherwise, the slower built-in routine will be called if the amount of memory is not enough to use the fast routines.
Finally, it should be noted that fastdiag is not compatible with and is not used by the generic Pentium PC GAMESS version.

==========

$VEC group (optional, relevant for all SCFTYP's)
(required if GUESS=MOREAD)

This group consists of formatted vectors, as written onto file PUNCH in a previous run. It is considered good form to retain the titling comment cards punched before the $VEC card, as a reminder to yourself of the origin of the orbitals.

For Morokuma decompositions, the names of this group are $VEC1, $VEC2, ... for each monomer, computed in the identical orientation as the supermolecule. For transition moment or spin-orbit coupling runs, orbitals for states one and possibly two are $VEC1 and $VEC2.

==========

$STATPT group (optional, for RUNTYP=OPTIMIZE or SADPOINT)

This group controls the search for stationary points. Note that NZVAR in $CONTRL determines if the geometry

search is conducted in Cartesian or internal coordinates.

METHOD = optimization algorithm selection. Pick from

NR Straight Newton-Raphson iterate. This will attempt to locate the nearest stationary point, which may be of any order. There is no steplength control. RUNTYP can be either OPTIMIZE or SADPOINT

RFO Rational Function Optimization. This is one of the augmented Hessian techniques where the shift parameter(s) is(are) chosen by a rational function approximation to the PES. For SADPOINT searches it involves two shift parameters. If the calculated stepsize is larger than DXMAX the step is simply scaled down to size.

QA Quadratic Approximation. This is another version of an augmented Hessian technique where the shift parameter is chosen such that the steplength is equal to DXMAX. It is completely equivalent to the TRIM method. (default)

SCHLEGEL The quasi-NR optimizer by Schlegel.

CONOPT, CONstrained OPTimization. An algorithm which can be used for locating TSs. The starting geometry MUST be a minimum! The algorithm tries to push the geometry uphill along a chosen Hessian mode (IFOLOW) by a series of optimizations on hyperspheres of increasingly larger radii. Note that there currently are no restart capabilitites for this method, not even manually.

GDIIS Geometry optimisation using Direct Inversion in the Iterative Subspace (highly recommended)

OPTTOL = gradient convergence tolerance, in Hartree/Bohr. Convergence of a geometry search requires the largest component of the gradient to be less than OPTTOL, and the root mean square gradient less than 1/3 of OPTTOL. (default=0.0001)

NSTEP = maximum number of steps to take. Restart data is punched if NSTEP is exceeded. (default=20)

NOREG = number of steps before delocalized coordinates are regenerated. Selecting a low value for NOREG is advised when your DLC based optimization aborts after a few steps due to failure to generate good delocalized coordinates.

--- the next four control the step size ---

DXMAX = initial trust radius of the step, in Bohr. For METHOD=RFO, QA, or SCHLEGEL, steps will

be scaled down to this value, if necessary.
(default=0.3 for OPTIMIZE and 0.2 for SADPOINT)
For METHOD=NR, DXMAX is inoperative.
For METHOD=CONOPT, DXMAX is the step along the
previous two points to increment the hypersphere
radius between constrained optimizations.
(default=0.1)

the next three apply only to METHOD=RFO or QA:

TRUPD = a flag to allow the trust radius to change as
the geometry search proceeds. (default=.TRUE.)

TRMAX = maximum permissible value of the trust radius.
(default=0.5 for OPTIMIZE and 0.3 for SADPOINT)

TRMIN = minimum permissible value of the trust radius.
(default=0.05)

--- the next three control mode following ---

IFOLOW = Mode selection switch, for RUNTYP=SADPOINT.
For METHOD=RFO or QA, the mode along which the
energy is maximized, other modes are minimized.
Usually refered to as "eigenvector following".
For METHOD=SCHLEGEL, the mode whose eigenvalue
is (or will be made) negative. All other
curvatures will be made positive.
For METHOD=CONOPT, the mode along which the
geometry is initially perturbed from the minima.
(default is 1)
In Cartesian coordinates, this variable doesn't
count the six translation and rotation degrees.
Note that the "modes" aren't from mass-weighting.

STPT = flag to indicate whether the initial geometry
is considered a stationary point. If .true.
the initial geometry will be perturbed by
a step along the IFOLOW normal mode with
stepsize STSTEP. (default=.false.)
The positive direction is taken as the one where
the largest component of the Hessian mode is
positive. If there are more than one largest
component (symmetry), the first is taken as
positive.
Note that STPT=.TRUE. has little meaning with
HESS=GUESS as there will be many degenerate
eigenvalues.

STSTEP = Stepsize for jumping off a stationary point.
Using values of 0.05 or more may work better.
(default=0.01)

IFREEZ = array of internal coordinates to freeze.
For example, IFREEZ(1)=1,3 freezes the two bond
lengths in the $ZMAT example, while optimizing
the angle. You cannot freeze Cartesian coords.

--- The next two control the hessian matrix quality ---

HESS = selects the initial hessian matrix.
= GUESS chooses a positive definite diagonal hessian. (default for RUNTYP=OPTIMIZE)
= READ causes the hessian to be read from a $HESS group. (default for RUNTYP=SADPOINT)
= RDAB reads only the ab initio part of the hessian, and approximates the effective fragment blocks.
= RDALL reads the full hessian, then converts any fragment blocks to 6x6 T+R shape. (this option is seldom used).
= CALC causes the hessian to be computed, see the $FORCE group.

IHREP = the number of steps before the hessian is recomputed. If given as 0, the hessian will be computed only at the initial geometry if you choose HESS=CALC, and never again. If nonzero, the hessian is recalculated every IHREP steps, with the update formula used on other steps. (default=0)

--- the next two control the amount of output ---
Let 0 mean the initial geometry, L mean the last geometry, and all mean every geometry.
Let INTR mean the internuclear distance matrix.
Let HESS mean the approximation to the hessian.
Note that a directly calculated hessian matrix will always be punched, NPUN refers only to the updated hessians used by the quasi-Newton step.

NPRT = 1 Print INTR at all, orbitals at all
0 Print INTR at all, orbitals at 0+L (default)
-1 Print INTR at all, orbitals never
-2 Print INTR at 0+L, orbitals never

NPUN = 3 Punch all orbitals and HESS at all
2 Punch all orbitals at all
1 same as 0, plus punch HESS at all
0 Punch all orbitals at 0+L, otherwise only occupied orbitals (default)
-1 Punch occ orbitals at 0+L only
-2 Never punch orbitals

HSSEND = a flag to control automatic hessian evaluation at the end of a successful geometry search. (default=.FALSE.)

MOVIE = a flag to create a series of structural data which can be show as a movie by the MacIntosh program Chem3D. The data is written to the file IRCDATA. (default=.FALSE.)

---- the following parameters are quite specialized ----

PURIFY = a flag to help eliminate the rotational and

translational degrees of freedom from the
initial hessian (and possibly initial gradient).
This is much like the variable of the same name
in $FORCE, and will be relevant only if internal
coordinates are in use. (default=.FALSE.)

PROJCT = a flag to eliminate translation and rotational
degrees of freedom from Cartesian optimizations.
The default is .TRUE. since this normally will
reduce the number of steps, except that this
variable is set false when POSITION=FIXED is
used during EFP runs.

ITBMAT = number of micro-iterations used to compute the
step in Cartesians which corresponds to the
desired step in internals. The default is 5.

UPHESS = SKIP do not update Hessian (not recommended)
BFGS default for OPTIMIZE using RFO or QA
POWELL default for OPTIMIZE using NR or CONOPT
POWELL default for SADPOINT
MSP mixed Murtagh-Sargent/Powell update
SCHLEGEL only choice for METHOD=SCHLEGEL

RESTAR = Enables restart of an optimization run. This
can only be used with IREST .ne. 0 in $CONTRL.
Use of this variable is discouraged.

---- NNEG, RMIN, RMAX, RLIM apply only to SCHLEGEL ----

NNEG = The number of negative eigenvalues the force
constant matrix should have. If necessary the
smallest eigenvalues will be reversed. The
default is 0 for RUNTYP=OPTIMIZE, and 1 for
RUNTYP=SADPOINT.

RMIN = Minimum distance threshold. Points whose root
mean square distance from the current point is
less than RMIN are discarded. (default=0.0015)

RMAX = Maximum distance threshold. Points whose root
mean square distance from the current point is
greater than RMAX are discarded. (default=0.1)

RLIM = Linear dependence threshold. Vectors from the
current point to the previous points must not
be colinear. (default=0.07)

==========

* *
See the 'Further information' chapter for
some help with OPTIMIZE and SADPOINT runs
* *

==========

$TRUDGE group (optional, required for RUNTYP=TRUDGE)

This group defines the parameters for a non-gradient optimization of exponents or the geometry. The TRUDGE package is a modified version of the same code from Michel Dupuis' HONDO 7.0 system, origially written by H.F.King. Presently the program allows for the optimization of 10 parameters.

Exponent optimization works only for uncontracted primitives, without enforcing any constraints. Two non-symmetry equivalent H atoms would have their p function exponents optimized separately, and so would two symmetry equivalent atoms! A clear case of GIGO.

Geometry optimization works only in HINT internal coordinates (see $CONTRL and $DATA groups). The total energy of all types of SCF wavefunctions can be optimized, although this would be extremely stupid as gradient methods are far more efficient. The main utility is for open shell MP2 or CI geometry optimizations, which may not be done in any other way with PC Gamess/Firefly.

OPTMIZ = a flag to select optimization of either geometry or exponents of primitive gaussian functions.
= BASIS for basis set optimization.
= GEOMETRY for geometry optimization (default). This means minima search only, there is no saddle point capability.

NPAR = number of parameters to be optimized.

IEX = defines the parameters to be optimized.

If OPTMIZ=BASIS, IEX declares the serial number of the Gaussian primitives for which the exponents will be optimized.

If OPTMIZ=GEOMETRY, IEX define the pointers to the HINT internal coordinates which will be optimized. (Note that not all internal coordinates have to be optimized.) The pointers to the internal coordinates are defined as: (the number of atom on the input list)*10 + (the number of internal coordinate for that atom). For each atom, the HINT internal coordinates are numbered as 1, 2, and 3 for BOND, ALPHA, and BETA, respectively.

P = Defines the initial values of the parameters to be optimized. You can use this to reset values given in $DATA. If omitted, the $DATA values are used. If given here, geometric data must be in Angstroms and degrees.

A complete example is a TCSCF multireference 6-31G geometry optimization for methylene,

```
 $CONTRL SCFTYP=GVB CITYP=GUGA RUNTYP=TRUDGE
         COORD=HINT $END
```

```
 $BASIS  GBASIS=N31 NGAUSS=6 $END
 $DATA
Methylene TCSCF+CISD geometry optimization
Cnv 2

C    6.     LC  0.00  0.0  0.00  -  O  K
H    1.    PCC  1.00  53.  0.00  +  O  K  I
 $END
 $SCF    NCO=3 NPAIR=1 $END
 $TRUDGE OPTMIZ=GEOMETRY  NPAR=2
         IEX(1)=21,22   P(1)=1.08 $END
 $CIDRT  GROUP=C2V SOCI=.TRUE. NFZC=1 NDOC=3 NVAL=1
         NEXT=-1 $END
```

using GVB-PP(1), or TCSCF orbitals in the CI. The starting bond length is reset to 1.09, while the initial angle will be 106 (twice 53). Result after 17 steps is R=1.1283056, half-angle=51.83377, with a CI energy of -38.9407538472

Note that you may optimize the geometry for an excited CI state, just specify

```
          $GUGDIA   NSTATE=5  $END
          $GUGDM    IROOT=3   $END
```

to find the equilibrium geometry of the third state (of five total states) of the symmetry implied by your $CIDRT.

==========

$TRURST group (optional, relevant for RUNTYP=TRUDGE)

This group specifies restart parameters for TRUDGE runs and accuracy thresholds.

KSTART indicates the conjugate gradient direction in which the optimization will proceed. (default = -1)

-1 .... indicates that this is a non-restart run.
 0 .... corresponds to a restart run.

FNOISE accuracy of function values.
Variation smaller than FNOISE are not considered to be significant (Def. 0.0005)

TOLF accuracy required of the function (Def. 0.001)

TOLR accuracy required of conjugate directions (Def. 0.05)

For geometry optimization, the values which give better results (closer to the ones obtained with gradient methods) are: TOLF=0.0001, TOLR=0.001, FNOISE=0.00001

==========

$FORCE group (optional, relevant for RUNTYP=HESSIAN,OPTIMIZE,SADPOINT)

This group controls the computation of the hessian matrix (the energy second derivative tensor, also known as the force constant matrix), and an optional harmonic vibrational analysis. This can be a very time consuming calculation. However, given the force constant matrix, the vibrational analysis for an isotopically substituted molecule is very cheap. Related input is HESS= in $STATPT, and the $MASS, $HESS, $GRAD, $DIPDR, $VIB groups.

METHOD = chooses the computational method.
= ANALYTIC is implemented only for SCFTYPs RHF, ROHF, and GVB (when NPAIR is 0 or 1). This is the default for these cases.
= NUMERIC is the default for all other cases: UHF, MCSCF, and all MP2 or CI runs.

RDHESS = a flag to read the hessian from a $HESS group, rather than computing it. This variable pertains only to RUNTYP=HESSIAN. See also HESS= in the $STATPT group. (default is .FALSE.)

PURIFY = controls cleanup
Given a $ZMAT, the hessian and dipole derivative tensor can be "purified" by transforming from Cartesians to internals and back to Cartesians. This effectively zeros the frequencies of the translation and rotation "modes", along with their IR intensities. The purified quantities are punched out. Purification does change the Hessian slightly, frequencies at a stationary point can change by a wave number or so. The change is bigger at non-stationary points. (default=.FALSE. if $ZMAT is given)

PRTIFC = prints the internal coordinate force constants. You MUST have defined a $ZMAT group to use this. (Default=.FALSE.)

--- the next four apply only to METHOD=NUMERIC ----

NVIB = Number of displacements in each Cartesian direction for force field computation.
= 1 Move one VIBSIZ unit in each positive Cartesian direction. This requires 3N+1 evaluations of the wavefunction, energy, and gradient, where N is the number of SYMMETRY UNIQUE atoms given in $DATA. (default)
= 2 Move one VIBSIZ unit in the positive direction and one VIBSIZ unit in the negative direction. This requires 6N+1 evaluations of the wavefunction and gradient, and gives a small improvement in accuracy. In particular, the frequencies will change from NVIB=1 results by no more than 10-100 wavenumbers, and usually much less. However, the normal modes will be more nearly symmetry adapted, and the residual rotational and translational "frequencies" will be much closer to zero.

VIBSIZ = Displacement size (in Bohrs). Default=0.01

Let 0 mean the Vib0 geometry, and
D mean all the displaced geometries

NPRT = 1 Print orbitals at 0 and D
= 0 Print orbitals at 0 only (default)

NPUN = 2 Punch all orbitals at 0 and D
= 1 Punch all orbitals at 0 and occupied orbs at D
= 0 Punch all orbitals at 0 only (default)

----- the rest control normal coordinate analysis ----

VIBANL = flag to activate vibrational analysis.
(the default is .TRUE. for RUNTYP=HESSIAN, and
otherwise is .FALSE.)

SCLFAC = scale factor for vibrational frequencies, used
in calculating the zero point vibrational energy.
Some workers correct for the usual overestimate
in SCF frequencies by a factor 0.89. ZPE or other
methods might employ other factors, see A.P.Scott,
L.Radom J.Phys.Chem. 100, 16502-16513 (1996).
The output always prints unscaled frequencies, so
this value is used only during the thermochemical
analysis. (Default is 1.0)

TEMP = an array of up to ten temperatures at which the
thermochemistry should be printed out. The
default is a single temperature, 298.15 K. To
use absolute zero, input 0.001 degrees.

FREQ = an array of vibrational frequencies. If the
frequencies are given here, the hessian matrix
is not computed or read. You enter any imaginary
frequencies as negative numbers, omit the
zero frequencies corresponding to translation
and rotation, and enter all true vibrational
frequencies. Thermodynamic properties will be
printed, nothing else is done by the run.

PRTSCN = flag to print contribution of each vibrational

mode to the entropy. (Default is .FALSE.)

DECOMP = activates internal coordinate analysis.
Vibrational frequencies will be decomposed into
"intrinsic frequencies", by the method of

J.A.Boatz and M.S.Gordon, J.Phys.Chem., 93,
1819-1826(1989). If set .TRUE., the $ZMAT group
may define more than 3N-6 (3N-5) coordinates.
(default=.FALSE.)

PROJCT = controls the projection of the hessian matrix.

The projection technique is described by W.H.Miller, N.C.Handy, J.E.Adams in J. Chem. Phys. 1980, 72, 99-112. At stationary points, the projection simply eliminates rotational and translational contaminants. At points with non-zero gradients, the projection also ensures that one of the vibrational modes will point along the gradient, so that there are a total of 7 zero frequencies. The other 3N-7 modes are constrained to be orthogonal to the gradient. Because the projection has such a large effect on the hessian, the hessian punched is the one BEFORE projection. For the same reason, the default is .FALSE. to skip the projection, which is mainly of interest in dynamical calculations.

==========

$CPHF group (relevant for analytic RUNTYP=HESSIAN)

This group controls the solution of the response equations, also known as coupled Hartree-Fock.

POLAR = a flag to request computation of the static polarizability, alpha. Because this property needs 3 additional response vectors, beyond those needed for the hessian, the default is to skip the property. (default = .FALSE.)

NWORD = controls memory usage for this step. The default uses all available memory. (default=0)

MXCPIT = When solving CPHF equations iteratively, MXCPIT controls the maximum allowed number of iterations.The default value is 50.

CPTOL = When solving CPHF equations iteratively, CPTOL controls the maximum allowed relative error of the approximate solution. Default value is 10^-5

==========

$HESS group (relevant for RUNTYP=HESSIAN if RDHESS=.TRUE.)
(relevant for RUNTYP=IRC if FREQ,CMODE not given)
(relevant for RUNTYP=OPTIMIZE,SADPOINT if HESS=READ)

Formatted force constant matrix (FCM), i.e. hessian matrix. This data is punched out by a RUNTYP=HESSIAN job, in the correct format for subsequent runs. The first card in the group must be a title card.

A $HESS group is always punched in Cartesians. It will be transformed into internal coordinate space if a

geometry search uses internals. It will be mass weighted (according to $MASS) for IRC and frequency runs.

The initial FCM is updated during the course of a geometry optimization or saddle point search, and will be punched if a run exhausts its time limit. This allows restarts where the job leaves off. You may want to read this FCM back into the program for your restart, or you may prefer to regenerate a new initial hessian. In any

case, this updated hessian is absolutely not suitable for frequency prediction!

==========

$GRAD group (relevant for RUNTYP=OPTIMIZE or SADPOINT)
(relevant for RUNTYP=HESSIAN when RDHESS=.TRUE.)

Formatted gradient vector at the $DATA geometry. This data is read in the same format it was punched out.

For RUNTYP=HESSIAN, this information is used to determine if you are at a stationary point, and possibly for projection. If omitted, the program pretends the gradient is zero, and otherwise proceeds normally.

For geometry searches, this information (if known) can be read into the program so that the first step can be taken instantly.

==========

$DIPDR group (relevant for RUNTYP=HESSIAN if RDHESS=.T.)

Formatted dipole derivative tensor, punched in a previous RUNTYP=HESSIAN job. If this group is omitted, then a vibrational analysis will be unable to predict the IR intensities, but the run can otherwise proceed.

==========

$VIB group (relevant for RUNTYP=HESSIAN, METHOD=NUMERIC)

Formatted card image -restart- data. This data is read in the format it was punched by a previous HESSIAN job to the file IRCDATA. Just add a " $END" card, and if the final gradient was punched as zero, delete the last set of data. Normally, IREST in $CONTRL will NOT be used in conjunction with a HESSIAN restart. The mere presence of this deck triggers the restart from cards. This deck can also be used to turn a single point differencing run into double differencing, as well as recovering from time limits, or other bombouts.

==========

$MASS group (relevant for RUNTYP=HESSIAN, IRC, or DRC)

This group permits isotopic substitution during the computation of mass weighted Cartesian coordinates. Of course, the masses affect the frequencies and normal modes of vibration.

AMASS = An array giving the atomic masses, in amu. The default is to use the mass of the most abundant isotope. Masses through element 104 are stored.

example - $MASS AMASS(3)=2.0140 $END
will make the third atom in the molecule a deuterium.

==========

$IRC group (relevant for RUNTYP=IRC)

This group governs the location of the intrinsic reaction coordinate, a steepest descent path in mass weighted corrdinates, that connects the saddle point to reactants and products.

----- there are five integration methods chosen by PACE.

PACE = GS2 selects the Gonzalez-Schlegel second order method. This is the default method. Related input is:

- GCUT cutoff for the norm of the mass-weighted gradient tangent (the default is chosen in the range from 0.00005 to 0.00020, depending on the value for STRIDE chosen below.
- RCUT cutoff for Cartesian RMS displacement vector. (the default is chosen in the range 0.0005 to 0.0020 Bohr, depending on the value for STRIDE)
- ACUT maximum angle from end points for linear interpolation (default=5 degrees)
- MXOPT maximum number of contrained optimization steps for each IRC point (default=20)
- IHUPD is the hessian update formula. 1 means Powell, 2 means BFGS (default=2)
- GA is a gradient from the previous IRC point, and is used when restarting.
- OPTTOL is a gradient cutoff used to determine if the IRC is approaching a minimum. It has the same meaning as the variable in $STATPT. (default=0.0001)

PACE = LINEAR selects linear gradient following (Euler's method). Related input is:

- STABLZ switches on Ishida/Morokuma/Komornicki reaction path stabilization. The default is .TRUE.
- DELTA initial step size along the unit bisector, if STABLZ is on. Default=0.025 Bohr.
- ELBOW is the collinearity threshold above which the stabilization is skipped. If the mass weighted gradients at QB and QC are almost collinear, the reaction path is deemed to be curving very little, and stabilization isn't needed. The default is 175.0 degrees. To always perform stabilization, input 180.0.
- READQB,EB,GBNORM,GB are energy and gradient data already known at the current IRC point. If it happens that a run with STABLZ on decides to skip stabilization because of ELBOW, this data will be punched to speed the restart.

PACE = QUAD selects quadratic gradient following. Related input is:

```
  SAB    distance to previous point on the IRC.
  GA     gradient vector at that historical point.


PACE = AMPC4  selects the fourth order Adams-Moulton
              variable step predictor-corrector.
              Related input is:

  GA0,GA1,GA2 which are gradients at previous points.


PACE = RK4    selects the 4th order Runge-Kutta variable
              step method.  There is no related input.



----- The next two are used by all PACE choices -----

STRIDE = Determines how far apart points on the reaction
         path will be.  STRIDE is used to calculate the
         step taken, according to the PACE you choose.
         The default is good for the GS2 method, which is
         very robust.  Other methods should request much
         smaller step sizes, such as 0.10 or even 0.05.
         (default = 0.30 sqrt(amu)-Bohr)
NPOINT = The number of IRC points to be located in this
         run. The default is to find only the next point.
         (default = 1)



----- The next two let you choose your output volume -----

    Let F mean the first IRC point found in this run,
    and L mean the final IRC point of this run.
    Let INTR mean the internuclear distance matrix.

NPRT   =  1  Print INTR at all, orbitals at all IRC points
          0  Print INTR at all, orbitals at F+L (default)
         -1  Print INTR at all, orbitals never
         -2  Print INTR at F+L, orbitals never

NPUN   =  1  Punch all orbitals at all IRC points
          0  Punch all orbitals at F+L, only occupied
             orbitals at IRC points between (default)
         -1  Punch all orbitals at F+L only
         -2  Never punch orbitals



----- The next two tally the reaction path results.  The
      defaults are appropriate for starting from a saddle
      point, restart values are automatically punched out.

NEXTPT = The number of the next point to be computed.
STOTAL = Total distance along the reaction path to next
         IRC point, in mass weighted Cartesian space.


----- The following controls jumping off the saddle point.
```

If you give a $HESS group, FREQ and CMODE will be generated automatically.

SADDLE = A logical variable telling if the coordinates given in the $DATA deck are at a saddle point (.TRUE.) or some other point lying on the IRC (.FALSE.). If SADDLE is true, either a $HESS group or else FREQ and CMODE must be given. (default = .FALSE.) Related input is:

TSENGY = A logical variable controlling whether the energy and wavefunction are evaluated at the transition state coordinates given in $DATA. Since you already know the energy from the transition state search and force field runs, the default is .F.
FORWRD = A logical variable controlling the direction to proceed away from a saddle point. The forward direction is defined as the direction in which the largest magnitude component of the imaginary normal mode is positive. (default =.TRUE.)
EVIB = Desired decrease in energy when following the imaginary normal mode away from a saddle point. (default=0.0005 Hartree)
FREQ = The magnitude of the imaginary frequency, given in cm**-1.
CMODE = An array of the components of the normal mode whose frequency is imaginary, in Cartesian coordinates. Be careful with the signs!

You must give FREQ and CMODE if you don't give a $HESS group, when SADDLE=.TRUE. The option of giving these two variables instead of a $HESS does not apply to the GS2 method, which must have a hessian input, even for restarts. Note also that EVIB is ignored by GS2 runs.

==========

* * * * * * * * * * * * * * * * * * *
For hints about IRC tracking, see
the 'Further information' section.
* * * * * * * * * * * * * * * * * * *

==========

$GRADEX group (optional, for RUNTYP=GRADEXTR)

This group controls the gradient extremal following algorithm. The GEs leave stationary points parallel to each of the normal modes of the hessian. Sometimes a GE leaving a minimum will find a transition state, and thus provides us with a way of finding that saddle point. GEs have many unusual mathematical properties, and you should be aware that they normally differ a great deal from IRCs.

The search will always be performed in cartesian coordinates, but internal coordinates along the way may be printed by the usual specification of NZVAR and $ZMAT.

METHOD = algorithm selection.
SR A predictor-corrector method due to Sun and Ruedenberg (default).
JJH A method due to Jorgensen, Jensen and Helgaker.

NSTEP = maximum number of predictor steps to take.

(default=50)

DPRED = the stepsize for the predictor step.
(default = 0.10)

STPT = a flag to indicate whether the initial geometry is considered a stationary point. If .TRUE., the geometry will be perturbed by STSTEP along the IFOLOW normal mode.
(default = .TRUE.)

STSTEP = the stepsize for jumping away from a stationary point. (default = 0.01)

IFOLOW = Mode selection option. (default is 1)
If STPT=.TRUE., the intial geometry will be perturbed by STSTEP along the IFOLOW normal mode. Note that IFOLOW can be positive or negative, depending on the direction the normal mode should be followed in. The positive direction is defined as the one where the largest component of the Hessian eigenvector is positive.

If STPT=.FALSE. the sign of IFOLOW determines which direction the GE is followed in. A positive value will follow the GE in the uphill direction. The value of IFOLOW should be set to the Hessian mode which is parallel to the gradient to avoid miscellaneous warning messages.

GOFRST = a flag to indicate whether the algorithm should attempt to locate a stationary point. If .TRUE., a straight NR search is performed once the NR step length drops below SNRMAX. 10 NR step are othen allowed, a value which cannot be changed.
(default = .TRUE.)

SNRMAX = upper limit for switching to straight NR search for stationary point location.
(default = 0.10 or DPRED, whichever is smallest)

OPTTOL = gradient convergence tolerance, in Hartree/Bohr.
Used for optimizing to a stationary point.
Convergence of a geometry search requires the

rms gradient to be less than OPTTOL.
(default=0.0001)

HESS = selection of the initial hessian matrix, if STPT=.TRUE.

= READ causes the hessian to be read from a $HESS
group.
= CALC causes the hessian to be computed. (default)

---- parameters on this page apply only to METHOD=SR ----

DELCOR = the corrector step should be smaller than this
value before the next predictor step is taken.
(default = 0.001)

MYSTEP = maximum number of micro iteration allowed to
bring the corrector step length below DELCOR.
(default=20)

SNUMH = stepsize used in the numerical differentiation
of the Hessian to produce third derivatives.
(default = 0.0001)

HSDFDB = flag to select determination of third derivatives.
At the current geometry we need the gradient, the
Hessian, and the partial third derivative matrix
in the gradient direction.

If .TRUE., the gradient is calculated at the
current geometry, and two Hessians are calculated
at SNUMH distance to each side in the gradient
direction. The Hessian at the geometry is formed
as the average of the two displaced Hessians.

If .FALSE., both the gradient and Hessian are
calculated at the current geometry, and one
additional Hessian is calculated at SNUMH in the
gradient direction.

The default double-sided differentiation produces
a more accurate third derivative matrix, at the
cost of an additional wave function and gradient.
(default = .TRUE.)

* * * * * * * * * * * * * * * * * * *
See the 'Further information' chapter
for some help with GRADEXTR runs.
* * * * * * * * * * * * * * * * * * *

==========

$DRC group (relevant for RUNTYP=DRC)

This group governs the dynamical reaction coordinate, a classical trajectory method based on quantum chemical potential energy surfaces. In PC Gamess/Firefly these may be either ab initio or semi-empirical. Because the vibrational period of a normal mode with frequency 500 wavenumbers is 67 fs, a DRC needs to run for many steps in order to sample a representative portion of phase space. Almost all DRCs break molecular symmetry, so build your molecule with C1 symmetry in $DATA, or specify NOSYM=1 in $CONTRL.

Restart data can be found in the job's OUTPUT file, with important results summarized to the IRCDATA file.

NSTEP = The number of DRC points to be calculated, not including the initial point. (default = 1000)

DELTAT = is the time step. (default = 0.1 fs)

TOTIME = total duration of the DRC computed in a previous job, in fs. The default is the correct value when initiating a DRC. (default=0.0 fs)

* * *

In general, a DRC can be initiated anywhere, so $DATA might contain coordinates of the equilibrium geometry, or a nearby transition state, or something else. You must also supply an initial kinetic energy, and the direction of the initial velocity, for which there are a number of options:

EKIN = The initial kinetic energy (default = 0.0 kcal/mol) See also ENM, NVEL, and VIBLVL regarding alternate ways to specify the initial value.

VEL = an array of velocity components, in Bohr/fs. When NVEL is false, this is simply the direction of the velocity vector. Its magnitude will be automatically adjusted to match the desired initial kinetic energy, and it will be projected so that the translation of the center of mass is removed. Give in the order vx1, vy1, vz1, vx2, vy2, ...

NVEL = a flag to compute the initial kinetic energy from the input VEL using the sum of mass*VEL*VEL/2. This flag is usually selected only for restarts. (default=.FALSE.)

* * *

The next two allow the kinetic energy to be partitioned over all normal modes. The coordinates in $DATA are likely to be from a stationary point! You must also supply a $HESS group.

VIBLVL = a flag to turn this option on (default=.FALSE.)

VIBENG = an array of energies (in units of multiples of the hv of each mode) to be imparted along each normal mode. The default is to assign the zero point energy only, VIBENG(1)=0.5, 0.5, ..., 0.5. If given as a negative number, the initial direction of the velocity vector is along the reverse direction of the mode. "Reverse" means the phase of the normal mode is chosen such that

the largest magnitude component is a negative value. An example might be VIBENG(4)=2.5 to add two quanta to mode 4, along with zero point energy in all modes.

* * *

The next three pertain to initiating the DRC along a single normal mode of vibration. No kinetic energy is assigned to the other modes. You must also supply a $HESS group.

NNM = The number of the normal mode to which the initial kinetic energy is given. The absolute value of NNM must be in the range 1, 2, ..., 3N-6. If NNM is a positive/negative value, the initial velocity will lie in the forward/reverse direction of the mode. "Forward" means the largest component of the normal mode is a positive value. (default=0)

ENM = the initial kinetic energy given to mode NNM, in units of vibrational quanta hv, so the amount depends on mode NNM's vibrational frequency, v. If you prefer to impart an arbitrary initial kinetic energy to mode NNM, specify EKIN instead. (default = 0.0 quanta)

* * *

To summarize, there are five different ways to specify the DRC trajectory:

1. VEL vector with NVEL=.TRUE. This is difficult to specify at your initial point, and so this option is mainly used when restarting your trajectory. The restart information is always in this format.
2. VEL vector and EKIN with NVEL=.FALSE. This will give a desired amount of kinetic energy in the direction of the velocity vector.
3. VIBLVL and VIBENG selected, to give initial kinetic energy to all of the normal modes.
4. NNM and ENM to give quanta to a single normal mode.
5. NNM and EKIN to give arbitrary kinetic energy to a single normal mode.

* * *

The most common use of the next two is to analyze a trajectory with respect to the minimum energy geometry the trajectory is traveling around.

NMANAL = a flag to select mapping of the mass-weighted Cartesian DRC coordinates and velocity (conjugate momentum) in terms of normal modes step by step. If you choose this option, you must supply both

C0 and a $HESS group from the stationary point.
(default=.FALSE.)

C0 = an array of the coordinates of the stationary point (the coordinates in $DATA might well be some other coordinates). Give in the order x1,y1,z1,x2,y2,...

* * *

The next option applies to all input paths which read a hessian: NMANAL, NNM, or VIBLVL. After the translations and rotations have been dropped, the normal modes are renumbered 1, 2, ..., 3N-6.

HESSTS = a flag to say if the hessian corresponds to a transition state or a minimum. This parameter controls deletion of the translation and rotation degrees of freedom, i.e. the default is to drop the first six "modes", while setting this flag on drops modes 2 to 7 instead. (default=.FALSE.)

* * *

The final variables control the volume of output. Let F mean the first DRC point found in this run, and L mean the last DRC point of this run.

NPRTSM = summarize the DRC results every NPRTSM steps, to the file IRCDATA. (default = 1)

NPRT = 1 Print orbitals at all DRC points
0 Print orbitals at F+L (default)
-1 Never print orbitals

NPUN = 2 Punch all orbitals at all DRC points
1 Punch all orbitals at F+L, and occupied orbitals at DRC points between
0 Punch all orbitals at F+L only (default)
-1 Never punch orbitals

==========

References:

J.J.P.Stewart, L.P.Davis, L.W.Burggraf,
J.Comput.Chem. 8, 1117-1123 (1987)

S.A.Maluendes, M.Dupuis, J.Chem.Phys. 93, 5902-5911(1990)

T.Taketsugu, M.S.Gordon, J.Phys.Chem. 99, 8462-8471(1995)

T.Taketsugu, M.S.Gordon, J.Phys.Chem. 99, 14597-604(1995)

T.Taketsugu, M.S.Gordon, J.Chem.Phys. 103, 10042-9(1995)

T.Taketsugu, M.S.Gordon, J.Chem.Phys. 104, 2834-40(1996)

M.S.Gordon, G.Chaban, T.Taketsugu
J.Phys.Chem. 100, 11512-11525(1996)

==========

$SURF group (relevant for RUNTYP=SURFACE and RUNTYP=RSURFACE)

This group allows you to probe a potential energy surface along a small grid of points. Note that there is no option to vary angles, only distances. The scan can be made for any SCFTYP, or for the MP2 or CI surface.

IVEC1 = an array of two atoms, defining a coordinate from the first atom given, to the second.

IGRP1 = an array specifying a group of atoms, which must include the second atom given in IVEC1. The entire group will be translated (rigidly) along the vector IVEC1, relative to the first atom given in IVEC1.

ORIG1 = starting value of the coordinate, which may be positive or negative. Zero corresponds to the distance given in $DATA.

DISP1 = step size for the coordinate.

NDISP1 = number of steps to take for this coordinate.

There are no reasonable defaults for these keywords, so you should input all of them. ORIG1 and DISP1 should be given in Angstrom.

IVEC2, IGRP2, ORIG2, DISP2, NDISP2 = have the identical meaning as their "1" counterparts, and permit you to make a two dimensional map along two displacement coordinates. If the "2" data are not input, the surface map proceeds in only one dimension.

Note that properties are not computed at these points, other than the energy.

REUSE = .t. reuses approximate hessian information during relaxed scans. Failure to add this keyword may compromise the completion of your relaxed scan job.

NORMV is seldom used and means to renorm vectors defined via VECT1 or VECT2 to unity.

NSURF controls multiple PES construction for multistate jobs like state-averaged MCSCF.

VECT1 and
VECT2 are double precision arrays containing coefficients of the Cartesian or internal coordinates forming displacements to scan, with the numbering scheme the same as of IFREEZ array in $STATPT. The use of these keywors allows the programs to scan angles or dihedrals, rather than distances. VECT1 (and VECT2) runs should be used in conjunction with DISPi and NDISPi. ORIGi are also allowed. RSURFACE runs should use ifreez in $STATPT to manually freeze the same

Cartesian/internal coordinates or their combinations which are selected to scan in $surf via VECT1/VECT2. This is not required for non-relaxed surface runs.

==========

$LOCAL group (relevant for LOCAL=RUEDNBRG, BOYS, or POP)

This group allows input of additional data to control the localization methods. If no input is provided, the valence orbitals will be localized as much as possible, while still leaving the wavefunction invariant.

PRTLOC = a flag to control supplemental printout. The extra output is the rotation matrix to the localized orbitals, and, for the Boys method, the orbital centroids, for the Ruedenberg method, the coulomb and exchange matrices, for the population method, atomic populations. (default=.FALSE.)

MAXLOC = maximum number of localization cycles. This applies to BOYS or POP methods only. If the localization fails to converge, a different order of 2x2 pairwise rotations will be tried. (default=250)

CVGLOC = convergence criterion. The default provides LMO coefficients accurate to 6 figures. (default=1.0E-6)

SYMLOC = a flag to restrict localization so that orbitals of different symmetry types are not mixed. This option is not supported in all possible point groups. The purpose of this option is to give a better choice for the starting orbitals for GVB-PP or MCSCF runs, without destroying the orbital's symmetry. This option is compatible with each of the 3 methods of selecting the orbitals to be included. (default=.FALSE.)

These parameters select the orbitals which are to be included in the localization. You may select from FCORE, NOUTA/NOUTB, or NINA/NINB, but may choose only one of these.

FCORE = flag to freeze all the chemical core orbitals present. All the valence orbitals will be localized. (default=.TRUE.)

* * *

NOUTA = number of alpha orbitals to hold fixed in the localization. (default=0)

MOOUTA = an array of NOUTA elements giving the numbers of

the orbitals to hold fixed. For example, the input NOUTA=2 MOOUTA(1)=8,13 will freeze only orbitals 8 and 13. You must enter all the orbitals you want to freeze, including any cores. This variable has nothing to do with cows.

NOUTB = number of beta orbitals to hold fixed in -UHF- localizations. (default=0)

MOOUTB = same as MOOUTA, except that it applies to the beta orbitals, in -UHF- wavefunctions only.

* * *

NINA = number of alpha orbitals which are to be included in the localization. (default=0)

MOINA = an array of NINA elements giving the numbers of the orbitals to be included in the localization. Any orbitals not mentioned will be frozen.

NINB = number of -UHF- beta MOs in the localization. (default=0)

MOINB = same as MOINA, except that it applies to the beta orbitals, in -UHF- wavefunctions only.

N.B. Since Boys localization needs the dipole integrals, do not turn off dipole moment calculation in $ELMOM.

----- The following keywords are used for the localized charge distribution (LCD) energy decomposition.

EDCOMP = flag to turn on LCD energy decomposition. Note that this method is currently implemented for SCFTYP=RHF and ROHF and LOCAL=RUEDNBRG only. The SCF LCD forces all orbitals to be localized, overriding input on the previous page. See also LMOMP2 in the $MP2 group. (default = .FALSE.)

MOIDON = flag to turn on LMO identification and subsequent LMO reordering, and assign nuclear LCD automatically. (default = .FALSE.)

DIPDCM = flag for LCD molecular dipole decomposition. (default = .FALSE.)

QADDCM = flag for LCD molecular quadrupole decomposition. (default = .FALSE.)

POLDCM = flag to turn on LCD polarizability decomposition. This method is implemented for SCFTYP=RHF or ROHF and LOCAL=BOYS or RUEDNBRG. (default=.FALSE.)

POLANG = flag to choose units of localized polarizability output. The default is Angstroms**3, while false will give Bohr**3. (default=.TRUE.)

ZDO = flag for LCD analysis of a composite wave function, given in a $VEC group of a van der Waals complex,

within the zero differential overlap approximation.
The MOs are not orthonormalized and the inter-
molecular electron exchange energy is neglected.
In addition, the molecular overlap matrix is printed
out. This is a very specialized option.
(default = .FALSE.)

----- The remaining keywords can be used to define the
nuclear part of an LCD. They are usually used to
rectify mistakes in the automatic definition
made when MOIDON=.TRUE. The index defining the
LMO number then refers to the reordered list of LMOs.

NNUCMO = array giving the number of nuclei assigned to a
particular LMO.

IJMO = is an array of pairs of indices (I,J), giving
the row (nucleus I) and column (orbital J)
index of the entries in ZIJ and MOIJ.

MOIJ = arrays of integers K, assigning nucleus K as the
site of the Ith charge of LCD J.

ZIJ = array of floating point numbers assigning a
charge to the Ith charge of LCD J.

IPROT = array of integers K, defining nucleus K as a
proton.

DEPRNT = a flag for additional decomposition printing,
such as pair contributions to various energy
terms, and centroids of the Ruedenberg orbitals.
(default = .FALSE.)

==========

* * * * * * * * * * * * * * * * * * *
For hints about localizations, and
the LCD energy decomposition, see
the 'Further information' section.
* * * * * * * * * * * * * * * * * * *

==========

$TWOEI group (relevant for EDCOMP=.TRUE. in $LOCAL)

Formatted transformed two-electron Coulomb and Exchange integrals as punched during a LOCAL=RUEDNBRG run. If this group is present it will automaticall be read in during such a run and the two-electron integrals do not have to be re-transformed. This group is especially useful for EDCOMP=.TRUE. runs when the localization has to be repeated for different definitions of nuclear LCDs.

==========

$ELMOM group (not required)

This group controls electrostatic moments calculation.

```
IEMOM  = 0 - skip this property
         1 - calculate monopole and dipole (default)
         2 - also calculate quadrupole moments
         3 - also calculate octupole moments

WHERE  = COMASS   - center of mass (default)
         NUCLEI   - at each nucleus
         POINTS   - at points given in $POINTS.

OUTPUT = PUNCH, PAPER, or BOTH (default)

IEMINT = 0 - skip printing of integrals (default)
         1 - print dipole integrals
         2 - also print quadrupole integrals
         3 - also print octupole integrals
        -2 - print quadrupole integrals only
        -3 - print octupole integrals only
```

The quadrupole and octupole tensors on the printout are formed according to the definition of Buckingham. Caution: only the first nonvanishing term in the multipole charge expansion is independent of the coordinate origin chosen, which is normally the center of mass.

==========

$ELPOT group (not required)

This group controls electrostatic potential calculation.

```
IEPOT = 0 skip this property (default)
        1 calculate electric potential

WHERE  = COMASS   - center of mass
         NUCLEI   - at each nucleus (default)
         POINTS   - at points given in $POINTS
         GRID     - at grid given in $GRID
         PDC      - at points controlled by $PDC.
         CUBE     - grid given in $CUBE. This is the default if
                    $CUBE is given


OUTPUT = PUNCH, PAPER, or BOTH (default)
```

This property is the electrostatic potential V(a) felt by a test positive charge, due to the molecular charge density. A nucleus at the evaluation point is ignored. If this property is evaluated at the nuclei, it obeys the equation

sum on nuclei(a) $Z(a)*V(a) = 2*V(nn) + V(ne)$.

The electronic portion of this property is called the diamagnetic shielding.

==========

$ELDENS group (not required)

This group controls electron density calculation.

```
IEDEN  = 0 skip this property (default)
       = 1 compute the electron density.

MORB   = The molecular orbital whose electron density is
         to be computed.  If zero, the total density is
         computed.  (default=0)

WHERE  = COMASS    - center of mass
         NUCLEI    - at each nucleus (default)
         POINTS    - at points given in $POINTS
         GRID      - at grid given in $GRID
         CUBE      - grid given in $CUBE. This is the default if
                     $CUBE is given

OUTPUT = PUNCH, PAPER, or BOTH (default)

IEDINT = 0 - skip printing of integrals (default)
         1 - print the electron density integrals

SPIND =.f.(default)/.t.  - calculate spin density if available

DIFFD  =.f.(default)/.t. - calculate delta density

SKIPHF=.t.(default)/.f.  - skip (HF - initial guess) delta density
                           calculation

DERIVS(1)=0,0,0 (default) - any combination of up to three numbers 0,1 &
                            2. Defines the level of derivatives
                            required. 0 means density, 1 means the
                            density gradient (or its norm), 2 - density
                            Laplacian.

GRDFLD= .f.(default)/.t. - modifies density gradient calculations so
                           that the 3-component vector gradient field is
                           written (by default, the norm of the gradient
                           is written).

ICORBS - integer array - indices of alpha MOs (positive) and (if
         available) beta MOs/NOs (negative) for which to calculate the
         values of orbitals (not the density!). Most of other $cube-
         related options are disabled in the presence of ICORBS array.
         In the punch file, the inner loop of the printout is over
         different MOs, then over z coordinate, etc...

UHFNOS= .f.(default)/.t. modifies icorbs array entries so that negative
                         entries are treated as requests to calculate
                         UHF NOs, not UHF beta orbitals.
```

==========

$ELFLDG group   (not required)

This group controls electrostatic field and electric field gradient calculation.

```
IEFLD  = 0 - skip this property (default)
```

```
         1 - calculate field
         2 - calculate field and gradient

WHERE  = COMASS   - center of mass
         NUCLEI   - at each nucleus (default)
         POINTS   - at points given in $POINTS

OUTPUT = PUNCH, PAPER, or BOTH (default)

IEFINT = 0 - skip printing these integrals (default)
         1 - print electric field integrals
         2 - also print field gradient integrals
        -2 - print field gradient integrals only
```

The Hellman-Feynman force on a nucleus is the nuclear charge multiplied by the electric field at that nucleus. The electric field is the gradient of the electric potential, and the field gradient is the hessian of the electric potential. The components of the electric field gradient tensor are formed in the conventional way, i.e. see D.Neumann and J.W.Moskowitz.

==========

$POINTS group (not required)

This group is used to input points at which properties will be computed. This first card in the group must contain the string ANGS or BOHR, followed by an integer NPOINT, the number of points to be used. The next NPOINT cards are read in free format, containing the X, Y, and Z coordinates of each desired point.

==========

$GRID group (not required)

This group is used to input a grid (plane through the molecule) on which properties will be calculated.

```
ORIGIN(i) = coordinates of the lower left corner of
            the plot.
XVEC(i)   = coordinates of the lower right corner of
            the plot.
YVEC(i)   = coordinates of the upper left corner of
            the plot.
SIZE      = grid increment, default is 0.25.
UNITS     = units of the above four values, it can be
            either BOHR or ANGS (the default).
```

Note that XVEC and YVEC are not necessarily parallel to the X and Y axes, rather they are the axes which you desire to see plotted by the MEPMAP contouring program.

==========

```
         * * * * * * * * * * * * * * * * * * * *
         For conversion factors, and references
         see the 'Further information' section.
```

* *

==========

$PDC group (relevant if WHERE=PDC in $ELPOT)

This group determines the points at which to compute the electrostatic potential, for the purpose of fitting atomic charges to this potential. Constraints on the fit which determines these "potential determined charges" can include the conservation of charge, the dipole, and the quadrupole.

PTSEL = determines the points to be used, choose from
GEODESIC to use a set of points on several fused sphere van der Waals surfaces, with points selected using an algorithm due to Mark Spackman. The results are similar to those from the Kollman/Singh method, but are less rotation dependent. (default)
CONNOLLY to use a set of points on several fused sphere van der Waals surfaces, with points selected using an algorithm due to Michael Connolly. This is identical to the method used by Kollman & Singh (see below)
CHELPG to use a modified version of the CHELPG algorithm, which produces a symmetric grid of points for a symmetric molecule.

CONSTR = NONE - no fit is performed. The potential at the points is instead output according to OUTPUT in $ELPOT.
CHARGE - the sum of fitted atomic charges is constrained to reproduce the total molecular charge. (default)
DIPOLE - fitted charges are constrained to exactly reproduce the total charge and dipole.
QUPOLE - fitted charges are constrained to exactly reproduce the charge, dipole, and quadrupole.

Note: the number of constraints cannot exceed the number of parameters, which is the number of nuclei. Planar molecules afford fewer constraint equations, namedly two dipole constraints and three quadrupole constraints, instead of three and five, repectively.

* * * the next 5 pertain to PTSEL=GEODESIC or CONNOLLY * * *

VDWSCL = scale factor for the first shell of VDW spheres. The default of 1.4 seems to be an empirical best value. Values for VDW radii for most elements up to Z=36 are internally stored.

VDWINC = increment for successive shells (default = 0.2). The defaults for VDWSCL and VDWINC will result in points chosen on layers at 1.4, 1.6, 1.8 etc

times the VDW radii of the atoms.

LAYER = number of layers of points chosen on successive fused sphere VDW surfaces (default = 4)

NFREQ = flag for particular geodesic tesselation of points. Only relevant if PTSEL=GEODESIC. Options are:
(10*h + k) for {3,5+}h,k tesselations
-(10*h + k) for {5+,3}h,k tesselations
(of course both nh and nk must be less than 10, so NFREQ must lie within the range -99 to 99)
The default value is NFREQ=30 (=03)

PTDENS = density of points on the surface of each scaled VDW sphere (in points per square au). Only relevant if PTSEL=CONNOLLY. Default is 0.28 per au squared, which corresponds to 1.0 per square Angstrom, the default recommended by Kollman & Singh.

* * * the next two pertain to PTSEL=CHELPG * * *

RMAX = maximum distance from any point to the closest atom. (default=3.0 Angstroms)

DELR = distance between points on the grid. (default=0.8 Angstroms)

MAXPDC = an estimate of the total number of points whose electrostatic potential will be included in the fit. (default=10000)

* * *

CENTER = an array of coordinates at which the moments were computed.

DPOLE = the molecular dipole.

QPOLE = the molecular quadrupole.

PDUNIT = units for the above values. ANGS (default) will mean that the coordinates are in Angstroms, the dipole in Debye, and quadrupole in Buckinghams. BOHR implies atomic units for all 3.

Note: it is easier to compute the moments in the current run, by setting IEMOM to at least 2 in $ELMOM. However, you could fit experimental data, for example, by reading it in here.

==========

There is no unique way to define fitted atomic charges. Smaller numbers of points at which the electrostatic potential is fit, changes in VDW radii, asymmetric point location, etc. all affect the results. A useful bibliography is

U.C.Singh, P.A.Kollman, J.Comput.Chem. 5, 129-145(1984)
L.E.Chirlain, M.M.Francl, J.Comput.Chem. 8, 894-905(1987)

R.J.Woods, M.Khalil, W.Pell, S.H.Moffatt, V.H.Smith,
J.Comput.Chem. 11, 297-310(1990)
C.M.Breneman, K.B.Wiberg, J.Comput.Chem. 11, 361-373(1990)
K.M.Merz, J.Comput.Chem. 13, 749(1992)
M.A.Spackman, J.Comput.Chem. 17, 1-18(1996)

==========

$MOLGRF group (relevant only if you have MOLGRAPH)

This option provides an interface for viewing orbitals through a commercial package named MOLGRAPH, from Daikin Industries

GRID3D = a flag to generate 3D grid data.
(default is .false.).

TOTAL = a flag to generate a total density grid data.
"Total" means the sum of the orbital densities
given by NPLT array. (default is .false.).

MESH = numbers of grids. You can use different numbers
for three axes. (default is MESH(1)=21,21,21).

BOUND = boundary coordinates of a 3D graphical cell. The
default is that the cell is larger than the
molecular skeleton by 3 bohr in all directions.
E.g., BOUND(1)=xmin,xmax,ymin,ymax,zmin,zmax

NPLOTS = number of orbitals to be used to generate 3D grid
data. (default is NPLOTS=1).

NPLT = orbital IDs. The default is 1 orbital only, the
HOMO or SOMO. If the LOCAL option is given in
$CONTRL, localized orbital IDs should be given.
For example, NPLT(1)=n1,n2,n3,...

CHECK = debug option, printing some of the grid data.

==========

$STONE group (optional)

This group defines the expansion points for Stone's distributed multipole analysis (DMA) of the electrostatic potential.

The DMA takes the multipolar expansion of each overlap charge density defined by two gaussian primitives, and translates it from the center of charge of the overlap density to the nearest expansion point. Some references for the method are

Stone, Chem.Phys.Lett. 83, 233 (1981)
Price and Stone, Chem.Phys.Lett. 98, 419 (1983)
Buckingham and Fowler, J.Chem.Phys. 79, 6426 (1983)
Stone and Alderton, Mol.Phys. 56, 1047 (1985)

The existence of a $STONE group in the input is what triggers the analysis. Enter as many lines as you wish, in any order, terminated by a $END record.

ATOM i name, where

ATOM	is a keyword indicating that a particular atom is selected as an expansion center.
i	is the number of the atom
name	is an optional name for the atom. If not entered the name will be set to the name used in the $DATA input.

ATOMS	is a keyword selecting all nuclei in the molecule as expansion points. No other input on the line is necessary.

BONDS	is a keyword selecting all bond midpoints in the molecule as expansion points. No other input on the line is necessary.

BOND i j name, where

BOND	is a keyword indicating that a bond midpoint is selected as an expansion center.
i,j	are the indices of the atoms defining the bond, corresponding to two atoms in $DATA.
name	an optional name for the bond midpoint. If omitted, it is set to 'BOND'.

CMASS	is a keyword selecting the center of mass as an expansion point. No other input on the line is necessary.

POINT x y z name, where

POINT	is a keyword indicating that an arbitrary point is selected as an expansion point.
x,y,z	are the coordinates of the point, in Bohr.
name	is an optional name for the expansion point. If omitted, it is set to 'POINT'.

==========

The second and third moments on the printout can be

converted to Buckingham's tensors by formula 9 of
  A.D.Buckingham, Quart.Rev. 13, 183-214 (1959)
These can in turn be converted to spherical tensors
by the formulae in the appendix of
  S.L.Price, et al. Mol.Phys. 52, 987-1001 (1984)

==========

$MOROKM group (relevant for RUNTYP=MOROKUMA)

This group controls how the supermolecule input in the $DATA group is divided into two or more monomers. Both the supermolecule and its constituent monomers must be well described by RHF wavefunctions.

MOROKM = a flag to request Morokuma-Kitaura decomposition. (default is .TRUE.)

RVS = a flag to request "reduced variation space" decomposition. This differs from the Morokuma option, and one or the other or both may be requested in the same run. (default is .FALSE.)

BSSE = a flag to request basis set superposition error be computed. You must ensure that CTPSPL is selected. This option applies only to MOROKM decompositions, as a basis superposition error is automatically generated by the RVS scheme. This is not the full Boys counterpoise correction, as explained in the reference. (default is .FALSE.)

* * *

IATM = An array giving the number of atoms in each of the monomer. Up to ten monomers may be defined. Your input in $DATA must have all the atoms in the first monomer defined before the atoms in the second monomer, before the third monomer... The number of atoms belonging to the final monomer can be omitted. There is no sensible default for IATM, so don't omit it from your input.

ICHM = An array giving the charges of the each monomer. The charge of the final monomer may be omitted, as it is fixed by ICH in $CONTRL, which is the total charge of the supermolecule. The default is neutral monomers, ICHM(1)=0,0,0,...

EQUM = a flag to indicate all monomers are equivalent by symmetry (in addition to containing identical atoms). If so, which is not often true, then only the unique computations will be done. (default is .FALSE.)

CTPSPL = a flag to decompose the interaction energy into charge transfer plus polarization terms. This is most appropriate for weakly interacting monomers. (default is .TRUE.)

CTPLX = a flag to combine the CT and POL terms into a single term. If you select this, you might want to turn CTPSPL off to avoid the extra work that that decomposition entails, or you can analyze both ways in the same run (default=.FALSE.)

RDENG = a flag to enable restarting, by reading the lines containing "FINAL ENERGY" from a previous run. The $ENERGY group is single lines read under format A16,F20.10 containing the E, and a card $END to complete. The 16 chars = anything. (default is .FALSE.)

==========

The present implementation has some quirks:

1. The initial guess of the monomer orbitals is not controlled by $GUESS. The program first looks for a $VEC1, $VEC2, ... group for each monomer. If they are found, they will be MOREAD. If any of these are missing, the guess for that monomer will be constructed by HCORE. Check your monomer energies carefully! The initial guess orbitals for the supermolecule are formed by a block diagonal matrix of the monomer orbitals.
2. The use of symmetry is turned off internally.
3. There is no direct SCF option. File ORDINT will be a full C1 list of integrals. File AOINTS will contain whatever subset of these is needed for each particular decomposition step. So extra disk space is needed compared to RUNTYP=ENERGY.
4. This kind of run applies only to ab initio cases.
5. This kind of run will work in parallel.

References:

C.Coulson in "Hydrogen Bonding", D.Hadzi, H.W.Thompson, Eds., Pergamon Press, NY, 1957, pp 339-360.
C.Coulson Research, 10, 149-159 (1957).
K.Morokuma J.Chem.Phys. 55, 1236-44 (1971).
K.Kitaura, K.Morokuma Int.J.Quantum Chem. 10, 325 (1976).
K.Morokuma, K.Kitaura in "Chemical Applications of Electrostatic Potentials", P.Politzer,D.G.Truhlar, Eds. Plenum Press, NY, 1981, pp 215-242.
The method coded is the newer version described in the latter two papers. Note that the CT term is computed separately for each monomer, as described in the words below equation 16 of the 1981 paper, not simultaneously.

Reduced Variational Space:
W.J.Stevens, W.H.Fink, Chem.Phys.Lett. 139, 15-22(1987).

A comparison of the RVS and Morokuma decompositions can be found in the review article: "Wavefunctions and Chemical Bonding" M.S.Gordon, J.H.Jensen in "Encyclopedia of Computational Chemistry", volume 5, P.V.R.Schleyer, editor, John Wiley and Sons, Chichester, 1998.

BSSE during Morokuma decomposition:
R.Cammi, R.Bonaccorsi, J.Tomasi
Theoret.Chim.Acta 68, 271-283(1985).

The present implementation:
"Energy decomposition analysis for many-body interactions,
 and application to water complexes"
W.Chen, M.S.Gordon J.Phys.Chem. 100, 14316-14328(1996)

==========

$FFCALC group (relevant for RUNTYP=FFIELD)

This group permits the study of the influence of an applied electric field on the wavefunction. The most common finite field calculation applies a sequence of fields to extract the linear polarizability and first and second order hyperpolarizability. The method is general, and so works for all ab initio wavefunctions in PC Gamess/Firefly.

EFIELD = applied electric field strength
(default=0.001 a.u.)

IAXIS and JAXIS specify the orientation of the applied field. 1,2,3 mean x,y,z respectively. The default is IAXIS=3 and JAXIS=0.

If IAXIS=i and JAXIS=0, the program computes alpha(ii), beta(iii), and gamma(iiii) from the energy changes, and a few more components from the dipole changes. Five wavefunction evaluations are performed.

If IAXIS=i and JAXIS=j, the program computes the cross terms beta(ijj), beta(iij), and gamma(iijj) from the energy changes, and a few more components from the dipole changes. This requires nine evaluations of the wavefunction.

AOFF = a flag to permit evaluation of alpha(ij) when the dipole moment is not available. This is necessary only for MP2, and means the off-axial calculation will do 13, not 9 energy evaluations. Default=.FALSE.

SYM = a flag to specify when the fields to be applied along the IAXIS and/or JAXIS (or according to EONE below) do not break the molecular symmetry. Since most fields do break symmetry, the default is .FALSE.

ONEFLD = a flag to specify a single applied field calculation will be performed. Only the energy and dipole moment under this field are computed. If this option is selected, only SYM and EONE input is heeded. The default is .FALSE.

EONE = an array of the three x,y,z components of the single applied field.

There are notes on RUNTYP=FFIELD on the next page.

Finite field calculations require large basis sets, and extraordinary accuracy in the wavefunction. To converge the SCF to many digits is sometimes problematic, but we suggest you use the input to increase integral accuracy and wavefunction convergence, for example

```
$CONTRL ICUT=20 ITOL=30 INTTYP=HONDO $END
$SCF    NCONV=10 FDIFF=.FALSE. $END
```

In many cases, the applied fields will destroy the molecular symmetry. This means the integrals are calculated once with point group symmetry to do the initial field free wavefunction evaluation, and then again with point group symmetry turned off. If the fields applied do not destroy symmetry, you can avoid this second calculation of the integrals by SYM=.TRUE. This option also permits use of symmetry during the applied field wavefunction evaluations.

Examples of fields that do not break symmetry are a Z-axis field for an axial point group which is not centrosymmetric (i.e. C2v). However, a second field in the X or Y direction does break the C2v symmetry. Application of a Z-axis field for benzene breaks D6h symmetry. However, you could enter the group as C6v in $DATA while using D6h coordinates, and regain the prospect of using SYM=.TRUE. If you wanted to go on to apply a second field for benzene in the X direction, you might want to enter Cs in $DATA, which will necessitate the input of two more carbon and hydrogen atom, but recovers use of SYM=.TRUE.

Reference: H.A.Kurtz, J.J.P.Stewart, K.M.Dieter
J.Comput.Chem. 11, 82-87 (1990).

For analytic computation of static and also frequency dependent NLO proerties, for closed shell cases, see the $TDHF group.

==========

$TDHF group (relevant for SCFTYP=RHF if RUNTYP=TDHF)

This group permits the analytic calculation of various static and/or frequency dependent polarizabilities, with an emphasis on important NLO properties such as second and third harmonic generation. The method is programmed only for closed shell wavefunctions, at the semi-empirical or ab initio level. Ab initio calculations may be direct SCF, or parallel, if desired.

Because the Fock matrices computed during the time-dependent Hartree-Fock CPHF are not symmetric, you may not use symmetry. You must enter NOSYM=1 in $CONTRL!

For a more general numerical approach to the static

properties, see $FFCALC.

NFREQ = Number of frequencies to be used. (default=1)

FREQ   = An array of energy values in atomic units. For
         example: if NFREQ=3 then FREQ(1)=0.0,0.1,0.25.
         By default, only the static polarizabilities are
         computed. (default is freq(1)=0.0)

    The conversion factor from Hartree to wave
    numbers is 219,474.6, and the wavelength is
    given (in nm) by 45.56/FREQ.

MAXITA = Maximum number of iterations for an alpha
         computation. (default=100)

MAXITU = Maximum number of iterations in the second order
         correction calculation. This applies to iterative
         beta values and all gammas. (default=100)

ATOL   = Tolerance for convergence of first-order results.
         (default=1.0d-05)

BTOL   = Tolerance for convergence of second-order results.
         (default=1.0d-05)

RETDHF = a flag to choose starting points for iterative
         calculations from best previous results.
         (default=.true.)

* * * the following NLO properties are available * * *

BSHG   = Calculate beta for second harmonic generation.

BEOPE = Calculate beta for electrooptic Pockels effect.

BOR   = Calculate beta for optical rectification.

GTHG   = Calculate gamma for third harmonic generation.

GEFISH = Calculate gamma for electric-field induced
         second harmonic generation.

GIDRI = Calculate gamma for intensity dependent
         refractive index.

GOKE   = Calculate gamma for optical Kerr effect.

    These will be computed only if a nonzero energy is requested. The default for each flag is .TRUE., and they may be turned off individually by setting some .FALSE. Note however that the program determines the best way to calculate them. For example, if you wish to have the SHG results but no gamma results are needed, the SHG beta will be computed in a non-iterative way from alpha(w) and alpha(2w). However if you request the computation of the THG gamma, the second order U(w,w) results are needed and an iterative SHG calculation will be performed whether you request it or not, as it is a required intermediate.

Reference:

S.P.Karna, M.Dupuis J.Comput.Chem. 12, 487-504 (1991).
P.Korambath, H.A.Kurtz, in "Nonlinear Optical Materials",
ACS Symposium Series 628, S.P.Karna and A.T.Yeates, Eds.
pp 133-144, Washington DC, 1996.

Review: D.P.Shelton, J.E.Rice, Chem.Rev. 94, 3-29(1994).

Starting from the PC GAMESS v. 7.0 build # 3448, TDHF and DFT code were extended to allow calculation of static and dynamic (hyper) polarizabilities using runtyp=tdhf.
For DFT, alpha values are exact, while beta and gamma are only approximate at present, as second-order (and higher) exchange-correlation kernels are not properly taken into account. This will be fixed when TDDFT gradient code will be incorporated into the future PC GAMESS versions.

The rest of this section is devoted to CITYP=TDHF runs for TDHF (a.k.a. RPA)excitation energies.

Current implementation allows the use of only RHF references, but can pick up both singlet and triplet excited states. Nuclear gradients are not yet programmed. Properties are available using "unrelaxed" density. Due to efficiency considerations, TDHF is programmed for SAPS (spin-adapted antisymmetrized product) basis only, so you cannot get both singlet and triplet states at once.

NCORE = n Omits the first n occupied alpha and beta orbitals from the calculation. The default for n is the number of chemical core orbitals.

NSTATE = Number of states to be found (excluding the ground state).

ISTATE = State for which properties and/or gradient will be calculated. Only one state can be chosen.

MULT = Multiplicity (1 or 3) of the singly excited SAPS (the reference is necessarily single RHF).

DIAGZN = Hamiltonian diagonalization method.
= DAVID use Davidson diagonalization. (default)
= FULL construct the full matrix in memory and diagonalize, thus determining all states
(not recommended except for small cases).

NGSVEC = Dimension of the Hamiltonian submatrix that is diagonalized to form the initial CI vectors. The default is the greater of NSTATE*2 and 10.

MXVEC = Maximum number of expansion basis vectors in the iterative subspace during Davidson iterations, before the expansion basis is truncated. The default is the larger of 8*NSTATE and NGSVEC.

NDAVIT = Maximum number of Davidson iterations. Default=50.

DAVCVG = Convergence criterion for Davidson eigenvectors. Eigenvector accuracy is proportional to DAVCVG, while the energy accuracy is proportional to its

square. The default is 3.0E-05.

RDTDVC = Flag to read TDHF vectors from a $TDVEC group in the input file. Default is .FALSE.

MNMEOP = Flag to force the use of the minimal amount of memory during the Davidson iterations. This is for debug purposes. The default is .FALSE.

MAXGC = maximum allowed number of trial vectors to be routed through GENCON engine, default is 1. If the number of trial vectors is greater than MAXGC, only FASTINTS will be used. The reason is that for moderately contracted GC basis sets like cc-pVXZ, gencon is faster than fastints only for relatively small number of trial vectors (this is by gencon design). On the other hand, for ANO-like basis sets, it is always better to set MAXGC to be equal the number of initial guess vectors, as fastints will be much slower.

PRTTOL = threshold for TDHF csf printout and also for states symmetry determination. Default is 0.05.

ISTSYM = symmetry of states of interest. Default is zero, i.e., does not use any symmetry during calculations. Setting this to the desired index of irrep (according to PC Gamess/Firefly numbering) will solve only for the states of the desired symmetry and exploiting full (including non-abelian) symmetry of molecule, thus significantly reducing computation time.

The state-tracking feature of the PC GAMESS' TDHF code can be activated by selecting negative value of istate in the $TDHF group. It is intended for geometry optimization of the excited states in the case of root flipping.

Note that oscillator strengths printed in the TDHF summary table are calculated using transition dipoles length form only.

Below is the sample input file:

```
 $CONTRL SCFTYP=RHF CITYP=TDHF $END
 $SYSTEM TIMLIM=3000 MEMORY=3000000 $END
 $BASIS  GBASIS=n31 ngauss=6 NDFUNC=1 $END
 $TDHF NSTATE=3 ISTSYM=0 ISTATE=1 $END
 $DATA
H2O
CNV 2

 O           8.0   0.0000000000   0.0000000000   0.7205815395
 H           1.0   0.0000000000   0.7565140024   0.1397092302
 $END
```

==========

$TDVEC group required if RDTDVC in $TDHF is chosen

This is formatted data generated by a previous TDHF run, to
be read back in as starting vectors.

$TDDFT group required when CITYP=TDDFT
(note that CITYP=TDDFT requires
DFTTYP to be set in $CONTRL)

Current implementation allows the use of only R-DFT references,
but can pick up both singlet and triplet excited states.
Nuclear gradients are not yet programmed (will be in the PC GAMESS v. 7.2).
Properties are available using "unrelaxed" density. Due to efficiency
considerations, TDDFT is programmed for SAPS (spin-adapted antisymmetrized
product) basis only, so you cannot get both singlet and triplet states at once.

NCORE = n Omits the first n occupied alpha and beta orbitals from
the calculation. The default for n is the number
of chemical core orbitals.

NSTATE = Number of states to be found (excluding the
ground state).

ISTATE = State for which properties and/or gradient will
be calculated. Only one state can be chosen.

MULT = Multiplicity (1 or 3) of the singly excited
SAPS (the reference is necessarily single R-DFT).

DIAGZN = Hamiltonian diagonalization method.
= DAVID use Davidson diagonalization. (default)
= FULL construct the full matrix in memory and
diagonalize, thus determining all states
(not recommended except for small cases).

NGSVEC = Dimension of the Hamiltonian submatrix that is
diagonalized to form the initial CI vectors.
The default is the greater of NSTATE*2 and 10.

MXVEC = Maximum number of expansion basis vectors in the
iterative subspace during Davidson iterations,
before the expansion basis is truncated. The
default is the larger of 8*NSTATE and NGSVEC.

NDAVIT = Maximum number of Davidson iterations.
Default=50.

DAVCVG = Convergence criterion for Davidson eigenvectors.
Eigenvector accuracy is proportional to DAVCVG,
while the energy accuracy is proportional to its
square. The default is 3.0E-05.

RDTDVC = Flag to read TDDFT vectors from a $TDVEC group
in the input file. Default is .FALSE.

MNMEOP = Flag to force the use of the minimal amount of
memory during the Davidson iterations. This is
for debug purposes. The default is .FALSE.

MAXGC = maximum allowed number of trial vectors to be routed through GENCON engine, default is 1. If the number of trial vectors is greater than MAXGC, only FASTINTS will be used. The reason is that for moderately contracted GC basis sets like cc-pVXZ, gencon is faster than fastints only for relatively small number of trial vectors (this is by gencon design). On the other hand, for ANO-like basis sets, it is always better to set MAXGC to be equal the number of initial guess vectors, as fastints will be much slower.

PRTTOL = threshold for TDDFT csf printout and also for states symmetry determination. Default is 0.05.

ISTSYM = symmetry of states of interest. Default is zero, i.e., does not use any symmetry during calculations. Setting this to the desired index of irrep(according to PC Gamess/Firefly numbering) will solve only for the states of the desired symmetry and exploiting full (including non-abelian) symmetry of molecule, thus significantly reducing computation time.

ALTER = flag to modify internal logic of Davidson diagonalization code to use dynamic number of trial vectors. Default is .true. Setting it to .false. will slow-down calculations by forcing DAVIDSON diagonalization code to work exactly as CIS code in GAMESS (US).

TDA = flag to request Tamm-Dancoff approximation to TDDFT (TDDFT/TDA), which is programmed for both pure and hybrid functionals. Default is .false.

The state-tracking feature of the PC GAMESS' TDDFT code can be activated by selecting negative value of istate in the $TDDFT group. It is intended for geometry optimization of the excited states in the case of root flipping.

Note that oscillator strengths printed in the TDDFT summary table are calculated using transition dipoles length form only.

Below is the sample input file:

```
 $CONTRL SCFTYP=RHF DFTTYP=BLYP CITYP=TDDFT $END
 $SYSTEM TIMLIM=3000 MEMORY=3000000 $END
 $BASIS  GBASIS=n31 ngauss=6 NDFUNC=1 $END
 $TDDFT NSTATE=3 ISTSYM=0 ISTATE=1 $END
 $DATA
H2O
CNV 2

 O           8.0   0.0000000000   0.0000000000   0.7205815395
 H           1.0   0.0000000000   0.7565140024   0.1397092302
 $END
```

The example of TDDFT/TDA calculations:

```
 $CONTRL SCFTYP=RHF DFTTYP=B3LYP CITYP=TDDFT $END
 $SYSTEM TIMLIM=3000 MEMORY=3000000 $END
```

```
 $BASIS  GBASIS=n31 ngauss=6 NDFUNC=1 $END
 $TDDFT NSTATE=3 ISTSYM=0 ISTATE=1 TDA=.t. $END
 $DATA
H2O
CNV 2

 O           8.0   0.0000000000   0.0000000000   0.7205815395
 H           1.0   0.0000000000   0.7565140024   0.1397092302
 $END
```

==========

$TDVEC group required if RDTDVC in $TDDFT is chosen

This is formatted data generated by a previous TDDFT run, to
be read back in as starting vectors.

==========

==========

$EFRAG group (optional)

This group gives the name and position of one or more effective fragment potentials. It consists of a series of free format card images, which may not be combined onto a single line! The position of a fragment is defined by giving any three points within the fragment, relative to the ab initio system defined in $DATA, since the effective fragments have a frozen internal geometry. All other atoms within the fragment are defined by information in the $FRAGNAME group.

-1- a line containing one or more of these options:

COORD	=CART	selects use of Cartesians coords to define the fragment position at line -3-. (default)
	=INT	selects use of Z-matrix internal coordinates at line -3-.
POLMETHD=SCF		indicates the induced dipole for each fragment due to the ab initio electric field and other fragment fields is updated only once during each SCF iteration.
	=FRGSCF	requests microiterations during each SCF iteration to make induced dipoles due to ab initio and other fragment fields self consistent amoung the fragments. (default) Both methods converge to the same dipolar interaction.

POSITION=OPTIMIZE Allows full optimization within the ab initio part, and optimization of the rotational and translational motions of each fragment. (default)
=FIXED Allows full optimization of the ab initio system, but freezes the position of the fragments. This makes sense only with two or more fragments, as what is frozen is the fragments' relative orientation.
=EFOPT the same as OPTIMIZE, but if the fragment gradient is large, up to 5 geometry steps in which only the fragments move may occur, before the geometry of the ab initio piece is relaxed. This may save time by reusing the two electron integrals for the ab initio system.

Input a blank line if all the defaults are acceptable.

-2- FRAGNAME=XXX

XXX is the name of the fragment whose coordinates are to be given next. All other information defining the fragment is given in a supplemental $XXX group, which is referred to below as a $FRAGNAME group.

A RHF/DZP EFP for water is internally stored in PC Gamess/Firefly. Choose FRAGNAME=H2OEF2 to look up this numerical data, and then skip the input of $H2OEF2 and $FRGRPL groups.

-3- NAME, X, Y, Z (COORD=CART)
NAME, I, DISTANCE, J, BEND, K, TORSION (COORD=INT)

NAME = the name of a fragment point. The name used here must match one of the points in $FRAGNAME.

X, Y, Z = Cartesian coordinates defining the position of this fragment point RELATIVE TO THE COORDINATE ORIGIN used in $DATA. The choice of units is controlled by UNITS in $CONTRL.

I, DISTANCE, J, BEND, K, TORSION = the usual Z-matrix connectivity internal coordinate definition. The atoms I, J, K must be atoms in the ab initio system from in $DATA, or fragment points already defined in the current fragment or previously defined fragments.

Line -3- must be given a total of three times to define this fragment's position.

Repeat lines -2- and -3- to enter as many fragments as you

desire, and then end the group with a $END line.

Note that it is quite typical to repeat the same fragment name at line -2-, to use the same fragment system at many different positions.

```
=============================================================

        * * * * * * * * * * * * * * * * * * * * *
        For tips on effective fragment potentials
          see the 'Further information' section
        * * * * * * * * * * * * * * * * * * * * *


=============================================================
```

(required for each FRAGNAME given in $EFRAG)

$FRAGNAME group

This group gives all pertinent information for a given effective fragment potential (EFP). This information falls into three categories:

- electrostatic (distributed multipoles, screening)
- distributed polarizabilities
- exchange repulsion

It is input using several different subgroups, which should be given in the order shown below. Each subgroup is specified by a particular name, and is terminated by the word STOP. You may omit any of the subgroups to omit that term from the EFP. All values are given in atomic units.

```
To input monopoles,             follow input sequence -EM-
To input dipoles,               follow input sequence -ED-
To input quadrupoles,           follow input sequence -EQ-
To input octupoles,             follow input sequence -EO-
To input screening parameters,  follow input sequence -ES-
To input polarizable points,    follow input sequence -P-
To input repulsive points,      follow input sequence -R-
```

-1- a single descriptive title card

-2- COORDINATES

COORDINATES signals the start of the subgroup containing the multipolar expansion terms (charges, dipoles, ...). Optionally, one can also give the coordinates of the polarizable points, or centers of exchange repulsion.

-3- NAME, X, Y, Z, WEIGHT, ZNUC

NAME is a unique string identifying the point.
X, Y, Z are the Cartesian coordinates of the point.
WEIGHT and ZNUC are the atomic mass and nuclear charge, and are given only for the points which are nuclei.

Typically the true nuclei will appear twice, once for

defining the positive nuclear charge and its screening, and a second time for defining the electronic distributed multipoles.

Repeat line -3- for each expansion point, and terminate the list with a "STOP".

-EM1- MONOPOLES

MONOPOLES signals the start of the subgroup containing the electronic and nuclear monopoles.

-EM2- NAME, CHARGE

NAME must match one given in the COORDINATES subgroup.
CHARGE = nuclear or electronic monopole at this point.

Repeat -EM2- to define all desired charges.
Terminate this subgroup with a "STOP".

-ED1- DIPOLES

DIPOLES signals the start of the subgroup containing the dipolar part of the multipolar expansion.

-ED2- NAME, MUX, MUY, MUZ

NAME must match one given in the COORDINATES subgroup.
MUX, MUY, MUZ are the components of the electronic dipole.

Repeat -ED2- to define all desired dipoles.
Terminate this subgroup with a "STOP".

-EQ1- QUADRUPOLES

QUADRUPOLES signals the start of the subgroup containing the quadrupolar part of the multipolar expansion.

-EQ2- NAME, XX, YY, ZZ, XY, XZ, YZ

NAME must match one given in the COORDINATES subgroup.
XX, YY, ZZ, XY, XZ, and YZ are the components of the electronic quadrupole moment.

Repeat -EQ2- to define all desired quadrupoles.
Terminate this subgroup with a "STOP".

-EO1- OCTUPOLES

OCTUPOLES signals the start of the subgroup containing the octupolar part of the multipolar expansion.

-EO2- NAME, XXX, YYY, ZZZ, XXY, XXZ,
XYY, YYZ, XZZ, YZZ, XYZ

NAME must match one given in the COORDINATES subgroup.

XXX, ... are the components of the electronic octupole.

Repeat -EO2- to define all desired octupoles.
Terminate this subgroup with a "STOP".

-ES1- SCREEN

SCREEN signals the start of the subgroup containing the screening terms (A*exp[-B*r**2]) for the distributed multipoles, which account for charge penetration effects.

-ES2- NAME, A, B

NAME must match one given in the COORDINATES subgroup.
A, B are the parameters of the Gaussian screening term.

Repeat -ES2- to define all desired screening points.
Terminate this subgroup with a "STOP".

-P1- POLARIZABLE POINTS

POLARIZABLE POINTS signals the start of the subgroup containing the distributed polarizability tensors, and their coordinates.

-P2- NAME, X, Y, Z

NAME gives a unique identifier to the location of this polarizability tensor. It might match one of the points already defined in the COORDINATES subgroup, but often does not. Typically the distributed polarizability tensors are located at the centroids of localized MOs.

X, Y, Z are the coordinates of the polarizability point. They should be omitted if NAME did appear in COORDINATES. The units are controlled by UNITS= in $CONTRL.

-P3- XX, YY, ZZ, XY, XZ, YZ, YX, ZX, ZY

XX, ... are components of the distributed polarizability, which is not a symmetric tensor. XY means dMUx/dFy, where MUx is a dipole component, and Fy is a component of an applied field.

Repeat -P2- and -P3- to define all desired polarizability tensors, and terminate this subgroup with a "STOP".

-R1- REPULSIVE POTENTIAL

REPULSIVE POTENTIAL signals the start of the subgroup containing the fitted exchange repulsion potential, for the interaction between the fragment and the ab initio part of the system. This term also accounts for charge transfer effects. The term has the form

N

```
      sum   C * exp[-D  * r**2]
       i     i        i
```

-R2- NAME, X, Y, Z, N

NAME may match one given in the COORDINATES subgroup, but need not. If NAME does not match one of the known points, you must give its coordinates X, Y, and Z, otherwise omit these three values. N is the total number of terms in the fitted repulsive potential.

-R3- C, D

These two values define the i-th term in the repulsive potential. Repeat line -R3- for all N terms.

Repeat -R2- and -R3- to define all desired repulsive potentials, and terminate this subgroup with a "STOP".

==========

The entire $FRAGNAME group is terminated by a " $END".

==========

$FRGRPL group

This group defines the inter-fragment repulsive potential, which consists primarily of exchange repulsions but also includes charge transfer. Note that the functional form used for the fragment-fragment repulsion differs from that used for the ab initio-fragment repulsion, which is defined in the $FRAGNAME group. The form of the potential is

```
       N
      sum   A * exp[-B * r]
       i     i        i
```

-1- PAIR=FRAG1 FRAG2

specifies which two fragment repulsions are being defined. $FRAGNAME input for the two names FRAG1 and FRAG2 must have been given.

-2- NAME1 NAME2 A B
or
NAME1 NAME2 'EQ' NAME3 NAME4

NAME1 must be one of the "NAME" points defined in the $FRAG1 group's COORDINATE section. Similarly NAME2 must be a point from the $FRAG2 group. In addition, NAME1 or NAME2 could be the keyword CENTER, indicating the center of mass of the fragment.

A and B are the parameters of the fitted repulsive
potential.

The second form of the input allows equal potential fits
to be used. The syntax implies that the potential between
the points NAME1 and NAME2 should be taken the same as the
potential previously given in this group for the pair of
points NAME3 and NAME4.

If there are NPT1 points in FRAG1, and NPT2 points in
FRAG2, input line -2- should be repeated NPT1*NPT2 times.
Terminate the pairs of potentials with a "STOP" card.
Any pairs which you omit will be set to zero interaction.

Typically the number of points on which fitted potentials
might be taken to be all the nuclei in a fragment, plus
the center of mass.

Repeat lines -1- and -2- for all pairs of fragments, then
terminate the group with a $END line.

==========
$PCM group (optional)

   This group controls solvent effect computations using
the Polarizable Continuum Method. If this group is found
in the input file, a PCM computation is performed. The
default calculation, chosen by selecting only the SOLVNT
keyword, is to compute the electrostatic free energy.
Appropriate numerical constants are provided for a wide
range of solvents. Additional keywords allow for more
sophisticated computations, namely cavitation, repulsion,
and dispersion free energies. The methodology for these
is general, but only numerical constants for water are
provided. There is additional information on PCM in the
'Further information' chapter of this manual.

ANALYZ = Number of passes performed with the PCManalyze in order to
          remove problematic tesserae and improve numerical stability.
          In some instances PCManalyze tends to enter into a
          quasi-infinite loop. If this happens, set analyz=0 to skip
          this procedure.

--- the first set of parameters controls the computation:
        ICOMP, ICAV, IDISP, IREP, IDP, and IFIELD.

ICOMP = Renormalization procedure for induced charges.
         Gradient runs require ICOMP be 0 or 2 only.
       = 0 No.
       = 1 Yes, each charge is corrected in proportion
           to the area of the tessera to which it belongs.
       = 2 Yes, using the same factor for all tesserae.
           (default)
       = 3 Yes, with explicit consideration of the
           portion of solute electronic charge outside
           the cavity, by the method of Mennucci and
           Tomasi. See the $NEWCAV group.

ICAV = At the end of the run, calculate the cavitation
energy, by the method of Pierotti and Claverie:
= 0 skip the computation (default)
= 1 perform the computation.

If ICAV=1, the following parameter is relevant:

TABS = the absolute temperature, in units K.
(default=298.0)

There are two procedures for the calculation
of the repulsion and dispersion free energy.
IDISP is incompatible with IREP and IDP.

IDISP = Calculation of both dispersion and repulsion
free energy through the empirical method of
Floris and Tomasi.
= 0 skip the computation (default)
= 1 perform the computation. See $DISREP group.

The next two options add repulsive and dispersive terms
to the solute hamiltonian, in an ab initio manner, by
the method of Amovilli and Mennucci.

IREP = Calculation of repulsion free energy
= 0 skip the computation (default)
= 1 perform the computation. See $NEWCAV group.

IDP = Calculation of dispersion free energy
= 0 skip the computation (default)
= 1 perform the computation. See $DISBS group.

If IDP=1, then three additional parameters must be
defined. The two solvent values correspond to water,
and therefore these must be input for other solvents.

WA = solute average transition energy. This is
computed from the orbital energies for RHF,
but must be input for MCSCF runs.
(default=1.10)
WB = ionization potential of solvent, in Hartrees.
(default=0.451)
ETA2 = square of the zero frequency refractive index
of the solvent. (default=1.75)

IFIELD = At run end, calculate the electric potential
and electric field generated by the apparent
surface charges.
= 0 skip the computation (default)
= 1 on nuclei
= 2 on a planar grid

If IFIELD=2, the following data must be input:

AXYZ,BXYZ,CXYZ = each defines three components of the
vertices of the plane where the reaction
field is to be computed (in Angstroms)
A ===> higher left corner of the grid
B ===> lower left corner of the grid

```
      C ===> higher right corner of the grid
NAB = vertical subdivision (A--B edge) of the grid
NAC = horizontal subdivision (A--C edge) of the grid.

IPRINT = 0 normal printing (default)
       = 1 turns on debugging printout



--- the next group of keywords defines the solvent

SOLVNT = keyword naming the solvent of choice.  The eight
         numerical constants defining the solvent are
         internally stored for the following:
             WATER (or H2O)
             CH3OH
             C2H5OH
             CLFORM (or CHCl3)
             METHYCL (or CH2Cl2)
             12DCLET (or C2H4Cl2)
             CTCL (or CCl4)
             BENZENE (or C6H6)
             TOLUENE (or C6H5CH3)
             CLBENZ (or C6H5Cl)
             NITMET (or CH3NO2)
             NEPTANE (or C7H16)
             CYCHEX (or C6H12)
             ANILINE (or C6H5NH2)
             ACETONE (or CH3COCH3)
             THF
             DMSO (or DMETSOX)
         The default solvent name is
             INPUT
         which indicates you will specify your solvent by
         giving the following 8 numerical values:

RSOLV  = the solvent radius, in units Angstrom
EPS    = the dielectric constant
EPSINF = the dielectric constant at infinite frequency.
         This value must be given only for RUNTYP=TDHF,
         if the external field frequency is in the optical
         range and the solvent is polar; in this case the
         solvent response is described by the electronic
         part of its polarization.  Hence the value of the
         dielectric constant to be used is that evaluated
         at infinite frequency, not the static one (EPS).
         For nonpolar solvents, the difference between
         the two is almost negligible.
TCE    = the thermal expansion coefficient, in units 1/K
VMOL   = the molar volume, in units ml/mole
STEN   = the surface tension, in units dyne/cm
DSTEN  = the thermal coefficient of log(STEN)
CMF    = the cavity microscopic coefficient

Values for TCE, VMOL, STEN, DSTEN, CMF need to be given
only for the case ICAV=1.  Input of any or all of these
values will override the internally stored value.

--- the next set of keywords defines the molecular cavity

NESFP  = the number of initial spheres.
```

(default = number of atoms in solute molecule)

ICENT = option for definition of initial spheres.
= 0 centers spheres on each nucleus. (default)
= 1 sphere centers XE, YE, ZE and radii RIN will be specified explicitly in $PCMCAV.

The cavity generation algorithm may use additional spheres to smooth out sharp grooves, etc. The following parameters control how many extra spheres are generated:

OMEGA and FRO = GEPOL parameters for the creation of the `added spheres' defining the solvent accessible surface. When an excessive number of spheres is

created, which may cause problems of convergence, the value of OMEGA and/or FRO must be increased. For example, OMEGA from 40 to 50 ... up to 90,
FRO from 0.2 ... up to 0.7.
(defaults are OMEGA=40.0, FRO=0.7)

RET = minimum radius (in A) of the added spheres. Increasing RET decreases the number of added spheres. A value of 100.0 inhibits the addition of spheres. (default=0.2)

==========

$PCMCAV group (optional)

This group controls generation of the cavity holding the solute during Polarizable Continuum Method runs. The cavity is a union of spheres, according to ICENT and associated input values given in $PCM. The data given here must be given in Angstrom units.

XE,YE,ZE = arrays giving the coordinates of the spheres.
if ICENT=0, the atomic positions will be used.
if ICENT=1, you must supply NESFP values here.

RIN = an array giving the sphere radii.
if ICENT=0, the program will look up the internally stored van der Waals radius for: H,He,
B,C,N,O,F,Ne, Na,Al,Si,P,S,Cl,Ar,
K,As,Se,Br,Kr, Rb,Sb,Te,I, Cs,Bi
Data for other elements is not tabulated.
if ICENT=1, give NESFP values.

ALPHA = an array of scaling factors, for the definition of the solvent accessible surface. If only the first value is given, all radii are scaled by the same factor. (default is ALPHA(1)=1.2)

Example: Suppose the 4th atom in your molecule is Fe, but all other atoms have van der Waals radii. You decide a good guess for Fe is twice the covalent radius: $PCMCAV RIN(4)=2.33 $END

The source for the van der Waals radii is "The Elements", 2nd Ed., John Emsley, Clarendon Press, Oxford, 1991,

except that for C,N,O, the U.Pisa's experience with the best radii for PCM treatment of singly bonded C,N,O atoms is used instead. The radii for a few transition metals are given by A.Bondi, J.Phys.Chem. 68, 441-451(1968).

==========

$NEWCAV group (optional)

This group controls generation of the "escaped charge" cavity, used when ICOMP=3 or IREP=1 in $PCM. This cavity is used only to calculate the fraction of the solute electronic charge escapes from the original cavity.

IPTYPE = choice for tessalation of the cavity's spheres.
= 1 uses a tetrahedron
= 2 uses a pentakisdodecahedron (default)

ITSNUM = m, the number of tessera to use on each sphere.

if IPTYPE=1, input m=30*(n**2), with n=1,2,3 or 4
if IPTYPE=2, input m=60*(n**2), with n=1,2,3 or 4
(default is 60)

*** the next three parameters pertain to IREP=1 ***

RHOW = density, relative to liquid water (default = 1.0)

PM = molecular weight (default = 18.0)

NEVAL = number of valence electrons on solute (default=8)

The defaults for RHOW, PM, and NEVAL correspond to water, and therefore must be correctly input for other solvents.

==========

$DISBS group (optional)

This group defines auxiliary basis functions used to evaluate the dispersion free energy by the method of Amovilli and Mennucci. These functions are used only for the dispersion calculation, and thus have nothing to do with the normal basis given in $BASIS or $DATA. If the input group is omitted, only the normal basis is used for the IDP=1 dispersion energy.

NADD = the number of added shells

XYZE = an array giving the x,y,z coordinates (in bohr) of the center, and exponent of the added shell, for each of the NADD shells.

NKTYPE = an array giving the angular momenta of the shells

An example placing 2s,2p,2d,1f on one particular atom,

```
 $DISBS  NADD=7 NKTYP(1)= 0 0 1 1 2 2 3
         XYZE(1)=2.9281086   0.0  .0001726   0.2
                 2.9281086   0.0  .0001726   0.05
```

```
                2.9281086   0.0  .0001726   0.2
                2.9281086   0.0  .0001726   0.05
                2.9281086   0.0  .0001726   0.75
                2.9281086   0.0  .0001726   0.2
                2.9281086   0.0  .0001726   0.2  $END
```

==========

$DISREP group (optional)

This group controls evaluation of the dispersion and repulsion energies by the empirical method of Floris and Tomasi. The group must be given with IDISP=1 in $PCM. The two options are controlled by ICLAV and ILJ, only one of which should be selected.

ICLAV = selects Claverie's disp-rep formalism.
= 0 skip computation.
= 1 Compute the solute-solvent disp-rep interaction as a sum over atom-atom interactions through a Buckingham-type formula (R^-6 for dispersion, exp for repulsion). (default)
Ref: Pertsin-Kitaigorodsky "The atom-atom potential method", page 146.

ILJ = selects a Lennard-Jones formalism.
= 0 skip computation. (default)
= 1 solute atom's-solvent molecule interaction is modeled by Lennard-Jones type potentials, R^-6 for dispersion, R^-12 for repulsion).

---- the following data must given for ICLAV=1:

RHO = solvent numeral density
N = number of atom types in the solvent molecule
NT = an array of the number of atoms of each type in a solvent molecule
RDIFF = distances between the first atoms of each type and the cavity
DK,RW = parameters of atom-atom interactions

The defaults are chosen for water,
RHO=3.348D-02
N=2
NT(1)=2,1
RDIFF(1)=1.20,1.50
DKT(1)=1.0,1.36
RWT(1)=1.2,1.5

---- the following data must given for ILJ=1:

RHO = solvent numeral density
EPSI = an array of energy constants referred to each atom of the solute molecule.
SIGMA = an array of typical distances, relative to each solute atom

==========

$SCRF group (optional)

The presence of this group in the input turns on the use of the Kirkwood-Onsager spherical cavity model for the study of solvent effects. The method is implemented for RHF, UHF, ROHF, GVB and MCSCF wavefunctions and gradients, and so can be used with any RUNTYP involving the gradient. The method is not implemented for MP2, CI, any of the semiempirical models, or for analytic hessians.

DIELEC = the dielectric constant, 80 is often used for H2O

RADIUS = the spherical cavity radius, in Angstroms

G = the proportionality constant relating the solute molecule's dipole to the strength of the reaction field. Since G can be calculated from DIELEC and RADIUS, do not give G if they were given.

Additional information on the $SCRF model can be found in the 'Further information' chapter of this manual.

==========

$ECP group (required if ECP=READ in $CONTRL)

This group lets you read in effective core potentials, for some or all of the atoms in the molecule. You can use built in potentials for some of the atoms if you like. This is a free format (positional) input group.

*** Give a card set -1-, -2-, and -3- for each atom ***

-card 1- PNAME, PTYPE, IZCORE, LMAX+1

PNAME is a 8 character descriptive tag for this potential. If it is repeated for a subsequent atom, no other information need be given on this card, and cards -2- and -3- may also be skipped. The information will be copied from the first atom by this PNAME. Do not use the option to repeat the previously read ECP for an atom with PTYPE=NONE, instead type "none".

PTYPE = GEN a general potential should be read.
= SBKJC look up the Stevens/Basch/Krauss/Jasien/ Cundari potential for this type of atom.
= HW look up the Hay/Wadt built in potential for this type of atom.
= NONE treat all electrons on this atom.

IZCORE is the number of core electrons to be removed.

LMAX is the maximum angular momentum occupied in the core orbitals being removed (usually). Give IZCORE and LMAX only if PTYPE is GEN.

*** For the first occurence of PNAME, if PTYPE is GEN, ***
*** then give cards -2- and -3-. Otherwise go to -1-. ***

*** Card sets -2- and -3- are repeated LMAX+1 times ***

The potential U(LMAX+1) is given first, followed by U(L)-U(LMAX+1), for L=1,LMAX.

-card 2- NGPOT

NGPOT is the number of Gaussians in this part of the
      local effective potential.

-card 3-   CLP,NLP,ZLP   (repeat this card NGPOT times)

CLP is the coefficient of this Gaussian in the potential.
NLP is the power of r for this Gaussian.
ZLP is the exponent of this Gaussian.

* * *

By far the easiest way to use the SBKJC potential for all atoms in the formic acid molecule is to request ECP=SBKJC in $CONTRL. But the next page shows two alternatives.

The first way is to look up the program's internally stored SBKJC potentials one atom at a time:

```
 $ECP
C-ECP SBKJC
H-ECP NONE
O-ECP SBKJC
O-ECP
H-ECP NONE
 $END
```

The second oxygen duplicates the first, no core electrons are removed for hydrogen. The order of the atoms must follow that generated by $DATA. Note PTYPE allows you to type in one or more atoms explicitly, while using built in data for some other atoms.

The second example reads all SBKJC potentials explicitly:

```
 $ECP
C-ECP GEN 2 1
1      ----- CARBON U(P) -----
 -0.89371  1  8.56468
2      ----- CARBON U(S)-U(P) -----
  1.92926  0  2.81497
 14.88199  2  8.11296
H-ECP NONE
O-ECP GEN 2 1
1      ----- OXYGEN U(P) -----
 -0.92550  1 16.11718
2      ----- OXYGEN U(S)-U(P) -----
  1.96069  0  5.05348
 29.13442  2 15.95333
O-ECP
H-ECP NONE
 $END
```

Again, the 2nd oxygen copies from the first. It is handy to use the rest of card -2- as a descriptive comment.

As a final example, for antimony we have LMAX+1=3 (there are core d's). One must first enter U(f), followed by U(s)-U(f), U(p)-U(f), U(d)-U(f).

Caution:

At the present time, there are some numerical problems in the ECP code, which has been programed to use spdfg basis sets, and core potentials up to g. If one is using a g basis function, or a g potential (bottom row elements beyond the lanthanide series), there are small errors in the ECP integrals. A tight optimization (OPTTOL=1D-05) will usually result in the energy rising slightly during the last few geometry steps. The error seems to be about 0.000002 Hartree, so be cautious about using a g potential or basis function. When both are used in the same run, the error in the energy is about 0.0002, which means the use of both is too inaccurate to be trusted. The use of f functions or f potentials (or lower) seems to be free of any errors.

==========

$EFIELD group (not required)

This group permits the study of the influence of an external electric field on the molecule. The method is general, and so works for all ab initio SCFTYPs.

EVEC = an array of the three x,y,z components of the applied electric field.

SYM = a flag to specify when the field to be applied breaks the molecular symmetry. Since most fields break symmetry, the default is .FALSE.

==========

Restrictions: analytic hessians are not available, but numerical hessians are. Because an external field causes a molecule with a dipole to experience a torque, geometry optimizations must be done in Cartesian coordinates only. Internal coordinates eliminate the rotational degrees of freedom, which are no longer free.

Notes: a hessian calculation will have two rotational modes with non-zero "frequency", caused by the torque. A gas phase molecule will rotate so that the dipole moment is anti-parallel to the applied field. To carry out this rotation during geometry optimization will take many steps, and you can help save much time by inputting a field opposite the molecular dipole. There is also a stationary point at higher energy with the dipole parallel to the field, which will have two imaginary frequencies in the hessian. Careful, these will appear as the first two modes in a hessian run, but will not have the i for imaginary included on the printout since they are rotational modes.

==========

$INTGRL group (optional)

This group controls AO integral formats. It should probably never be given, as the program always picks sensible values.

SCHWRZ = a flag to activate use of the Schwarz inequality to predetermine small integrals. There is no loss of accuracy when choosing this option, and there are appreciable time savings for bigger molecules. Default=.TRUE. for over 5 atoms, or for direct SCF, and is .FALSE. otherwise.

NOPK = 0 PK integral option on, which is permissible for RHF, UHF, ROHF, GVB energy/gradient runs.
= 1 PK option off (default for all jobs). Must be off for anything with a transformation.

NORDER = 0 (default)
= 1 Sort integrals into canonical order. There is little point in selecting this option, as no part of PC Gamess/Firefly requires ordered integrals. See also NSQUAR.

NINTMX = Maximum no. of integrals in a record block. (default=15000 for J or P file, =10000 for PK)

The following parameters control the integral sort. (values given are defaults)

NSQUAR = 0 Sorted integrals will be in triangular canonical order (default)
= 1 instead sort to square canonical order.
NDAR = Number of direct access logical records to be used for the integral sort (default=2000)
LDAR = Length of direct access records (site dependent)
NBOXMX = 200 Maximum number of bins.
NWORD = 0 Memory to be used (default=all of it).
NOMEM = 0 If non-zero, force external sort.

IREST = 0 Normal operation (default)
= 1 Only 1-e integrals and integrals needed for Schwarz screening are calculated, 2-e integrals are read from the
= 2 Only the integrals needed for Schwarz screening are calculated, 2-e integrals are read from the existing AOINTS file.
= -1 Similar to 1, but 2-e integrals are never recalculated, even if the molecular geometry is changed.
= -2 Similar to 2, but 2-e integrals are never recalculated, even if the molecular geometry is changed.

PACKAO = .False. Normal operation
= .True. Turns on the additional packing of the AO integral file. Both indices and integrals are packed. This option reduces the size of AOINTS file by a factor of up to 2.5-3, thus saving the disk space and the CPU time. See also a more detailed discussion on the packing-related problems. Relevant only for NOPK=1 case (conventional integral storage mode, not in a

```
                        supermatrix form)
 PKTHR                  A threshold to activate packing of 2-e integrals.
                        Default is 10^-7

      The following parameters control integral restarts
      (values given are defaults)
 IST=    1      JST=    1    KST=    1    LST=    1
 NREC=   1      INTLOC= 1
```

==========

$FMM group (relevant if QFMM selected in $INTGRL)

QFMM is turned on by the logical variable QFMM in the $INTGRL group (its default value is .false.,i.e., no QFMM calculations). You must select DIRSCF=.TRUE. in $SCF and SCHWRZ=.TRUE. (default) in $INTGRL if you use this option. Most of the QFMM-related options are controlled by the corresponding $FMM group. Some keywords in the $CONTRL group affect QFMM as well, namely ICUT, ITOL, FSTINT and REORDR.

This group controls the quantum fast multipole method evaluation of Fock matrices. The defaults are reasonable, so there is little need to give this input.

ITGERR = Target error in final energy, to 10**-(ITGERR) Hartree. The accuracy is usually better than the setting of ITGERR, in fact QFMM runs should suffer no loss of accuracy or be more accurate than a conventional integral run (default=7).

QOPS = a flag to use the Quantum Optimum Parameter Searching technique, which finds an optimum FMM parameter set. (Default=.TRUE.)

If QOPS=.FALSE., the ITGERR value is not used. In this case the user should specify the following parameters:

NP = the highest multipole order for FMM (Default=15).

NS = the highest subdivision level (Default=2).

IWS = the minimum well-separateness (Default=2).

IDPGD = point charge approximation error (10**(-IDPGD)) of the Gaussian products (Default=9).

IEPS = very fast multipole method (vFMM) error, (10**(-IEPS)) (Default=9)

These are additional useful options which are either PC GAMESS specific or not documented in the GAMESS (US) manual due to bugs in their implementation:

METHOD = one of DISK, SEMIDRCT, or FULLDRCT. Controls disk vs CPU usage during first (FMM) part of the calculations. At present, FULLDRCT (fully direct) is equivalent to SEMIDRCT (semidirect). Semidirect uses less disk space and is usually faster than disk-based (DISK), especially for very large systems, and is the default.

NUMRD = (positive integer). Controls disk read caching during FMM,

as well as the granularity of the static/dynamic load balancing during parallel QFMM runs. The default value is 10 and is reasonable in most cases.

MODIFY = a flag to allow QOPS code to modify ICUT and ITOL variables of $CONTRL group. The default is .false., i.e., not to modify them. Although it can be set it to .true. for better compatibility with GAMESS (US) it is generally not recommended to activate this option.

MQOPS = 0 or 1. If set to zero (default) and when QOPS=.TRUE., SCLF AND NS are determined automatically. If MQOPS=1, user provided values of SCLF and NS override those found by QOPS.

SCLF = FMM cube scaling factor, must be greather or equal to 1.00

STATIC = a flag to use static load balancing (SLB) during FMM part of calculations even if DLB is activated. Default is .true. because in the case of homogeneous environment the static load balancing is implemented much more efficient.

NEARJ = 0, 1, or 2. Selects the routine to calculate near field Coulomb terms.

NEARJ=0 means to select an optimal default based on FSTINT & REORDR settings in $CONTRL.

NEARJ=1 means use of the bugfixed/improved GAMESS (US)-based routine using HONDO integral package.

NEARJ=2 means use of the PC GAMESS specific routine based on the fastints code which is generally much faster. It requires FSTINT and REORDR to be set in $CONTRL.

LEX = 0, 1, 2, or 3. Selects the routine to calculate HF exchange terms.

LEX=0 means to select an optimal default based on the FSTINT & REORDR settings in $CONTRL, as well as on the molecular symmetry.

LEX=1 means use of the bugfixed/improved GAMESS (US)-based routine using HONDO integral package. You should take into account that this implementation of linear exchange is by design to some degree approximate and is not fully equivalent to direct SCF, although in most cases one can safely neglect this fact.

LEX=2 means use of the fastints based routine which evaluates some extra integrals but is the only part of the QFMM which can take into account the molecular symmetry. It is strictly equivalent to the direct SCF. For highly-symmetrical systems it is faster than any other available method. It requires FSTINT to be set in $CONTRL.

LEX=3 means use of the fastints based routine which evaluates the minimal number of all the necessary 2-e integrals and is also strictly equivalent to direct SCF. It does not exploit molecular symmetry, though. For low-symmetry systems it is the fastest method available. It requires FSTINT and REORDR to be set in $CONTRL.

Default is LEX=0.

SKIP1 = a flag to modify the behavior of the inner loop of the 2-e integral selection code of the LEX=1 exchange routine. SKIP1=.true. means GAMESS (US)-style behavior. SKIP1=.false. makes it more precise by the cost of some CPU overhead. The default is .true. (see note at the end of the STRICT flag input description).

SKIP2 = a flag to modify the behavior of the outer loop of the 2-e integral selection code of the LEX=1 exchange routine. SKIP2=.true. means GAMESS (US)-style behavior. SKIP2=.false. makes it more precise by the cost of some CPU overhead. The default is .true. (see note at the end of the STRICT flag input description).

STRICT = a flag to modify the behavior of the density matrix sorting and nonzero elements selection for the LEX=1 exchange routine. STRICT=.false. means GAMESS (US)-style behavior. STRICT=.true. makes it more precise by the cost of some CPU overhead. The default is .false. because even if SKIP1=.false. SKIP2=.false. STRICT=.true. the LEX=1 routine is not exactly equivalent to direct SCF, while the CPU overhead is very significant.

==========

$TRANS group (optional for -CI- or -MCSCF-)
(relevant to analytic hessians)
(relevant to energy localization)

This group controls the integral tranformation. MP2 integral transformations are controlled instead by the $MP2 input group. There is little reason to give any but the first variable.

DIRTRF = a flag to recompute AO integrals rather than storing them on disk. The default is .FALSE. for MCSCF and CI runs. If your job reads $SCF, and you select DIRSCF=.TRUE. in that group, a direct transformation will be done, no matter how DIRTRF is set.

Note that the transformation may do many passes over the AO integrals for large basis sets, and thus the direct recomputation of AO integrals can be very time consuming.

MPTRAN = method to use for the integral transformation. the default is try 0, then 1, then 2.
0 means use the incore method
1 means use the segmented method. This is the only method that works in parallel.
2 means use the alternate method, which uses less memory than 2, but requires an extra large disk file.

NWORD = Number of words of fast memory to allow. Zero uses all available memory. (default=0)

CUTTRF = Threshold cutoff for keeping transformed two electron integrals. (default= 10**(-9))

AOINTS = defines AO integral storage during conventional
integral transformations, during parallel runs.
DUP stores duplicated AO lists on each node, and
is the default for parallel computers with slow
interprocessor communication, e.g. ethernet.
DIST distributes the AO integral file across
all nodes, and it is the default for parallel
computers with high speed communications.
DELETE means that AOINTS file is deleted when the integral
transformation is done

IREST = 0 Normal operation
= 1 Only 1-e integrals are transformed, while 2-e integrals are
read from the existing MOINTS file Default is 0
= 2 Both 1-e and 2-e integrals are read from the existing
MOINTS file
=-1,-2 Similar to 1 and 2, but the effect is permanent for the
whole calculation.

==========

$NBO

From the very beginning of its history, PC GAMESS included fully-functional NBO module. However, NBO part of the PC GAMESS requires activation as it is a commercial code. You should purchase PC GAMESS NBO license codes and NBO manual from TCI/NBO. The PC GAMESS' NBO license is very inexpensive ($30).
To activate NBO module, you should add the following strings to your PC GAMESS input:

```
$license nbolid=lid nbokey=key $end
$nbo < nbo options > $end
```

Normally, you should receive both NBO activation key (nbokey, 8-digit hexadecimal code), and NBO license ID (nbolid, some decimal number) from TCI. Nevertheless, in many cases they forgot to provide users by nbolid value. If it is the case, please contact us as we can recover your nbolid using NBO activation key value by just looking in our NBO database.
Note that we have NBO v. 5.0 code at hand, nevertheless, the PC GAMESS at moment has older version of NBO (4.M) incorporated in it. We plan to upgrade NBO code in the PC GAMESS to NBO v. 5.0 soon enough. The purchased PC GAMESS NBO license will be valid for both versions 4.M and 5.x of NBO code.

==========

The remaining groups apply only to -CI- and -MCSCF- runs.

* * * * * * * * * * * * * * * * * * * *
For hints on how to do -CI- and -MCSCF-
see the 'Further information' section
* * * * * * * * * * * * * * * * * * * *

==========

$CIINP group (optional, relevant for CITYP=GUGA or ALDET)

This group is the control box for Graphical Unitary Group Approach (GUGA) CI calculations, or Ames Laboratory determinant (ALDET) full CI. Each step which is executed potentially requires a further input group described later.

NRNFG = An array of 10 switches controlling which steps of a CI computation are performed.
1 means execute the module, 0 means don't.

- NRNFG(1) = Generate the configurations. See either $CIDRT or $CIDET input. (default=1)
- NRNFG(2) = Transform the integrals. See $TRANS. (default=1)
- NRNFG(3) = Sort integrals and calculate the Hamiltonian matrix. See $CISORT and $GUGEM. (default=1) This does not apply to ALDET.
- NRNFG(4) = Diagonalize the Hamiltonian matrix. See $GUGDIA or $CIDET. (default=1)
- NRNFG(5) = Construct the one electron density matrix, and generate NO's. See $GUGDM or $CIDET. (default=1)
- NRNFG(6) = Construct the two electron density matrix. See $GUGDM2 or $CIDET. (default=0 normally, but 1 for CI gradients)
- NRNFG(7) = Construct the Lagrangian of the CI function. Requires DM2 matrix exists. See $LAGRAN. (default=0 normally, but 1 for CI gradients) This does not apply to ALDET.
- NRNFG(8-10) are not used.

Users are not encouraged to change these values, as the defaults are quite reasonable ones.

NPFLG = An array of 10 switches to produce debug printout. There is a one to one correspondance to NRNFG, set to 1 for output. (default = 0,0,0,0,0,0,0,0,0,0) The most interesting is NPFLG(2)=1 to see the transformed 1e- integrals, NPFLG(2)=2 adds the very numerous transformed 2e- integrals to this.

IREST = n Restart the -CI- at stage NRNFG(n).

CASTRF= .t. Selects fast MCSCF-like integral transformation for standalone CI runs.

==========

$CIS group required when CITYP=CIS

The CIS method (singly excited CI) is the simplest way to treat excited states. By Brillouin's Theorem, a single determinant reference such as RHF will have zero matrix elements with singly substituted determinants. The ground state reference therefore has no mixing with the excited states treated with singles only. Reading the references given in Section 4 of this manual will show the CIS method can be thought of as a non-correlated method, rigorously

so for the ground state, and effectively so for the various excited states. Some issues making CIS rather less than a black box method are:

a) any states characterized by important doubles are simply missing from the calculation.
b) excited states commonly possess Rydberg (diffuse) character, so the AO basis used must allow this.
c) excited states often have different point group symmetry than the ground state, so the starting geometries for these states must reflect their actual symmetry.
d) excited state surfaces frequently cross, and thus root flipping may very well occur.

The CIS code in the PC GAMESS is based on heavily modified sources of the original GAMESS (US) AO-basis CIS code written by Simon P. Webb

The implementation allows the use of only RHF references, but can pick up both singlet and triplet excited states. Nuclear gradients are available, as are properties.

NCORE = n Omits the first n occupied alpha and beta orbitals from the calculation. The default for n is the number of chemical core orbitals.

NSTATE = Number of states to be found (excluding the ground state).

ISTATE = State for which properties and/or gradient will be calculated. Only one state can be chosen.

HAMTYP = Type of CI Hamiltonian to use.
= SAPS spin-adapted antisymmetrized product of the desired MULT will be used (default)
= DETS determinant based, so both singlets and triplets will be obtained. Note that this option will disable use of FASTINTS/GENCON code during direct CIS runs.

MULT = Multiplicity (1 or 3) of the singly excited SAPS (the reference is necessarily single RHF). Only relevant for SAPS based run.

DIAGZN = Hamiltonian diagonalization method.
= DAVID use Davidson diagonalization. (default)
= FULL construct the full matrix in memory and diagonalize, thus determining all states (not recommended except for small cases).

DGAPRX = Flag to control whether approximate diagonal elements of the CIS Hamiltonian (based only on the orbital energies) are used in the Davidson algorithm. Note, this only affects the rate of convergence, not the resulting final energies. If set .FALSE., the exact diagonal elements are determined and used. Default=.TRUE.

NGSVEC = Dimension of the Hamiltonian submatrix that is diagonalized to form the initial CI vectors. The default is the greater of NSTATE*2 and 10.

MXVEC = Maximum number of expansion basis vectors in the iterative subspace during Davidson iterations, before the expansion basis is truncated. The default is the larger of 8*NSTATE and NGSVEC.

NDAVIT = Maximum number of Davidson iterations. Default=50.

DAVCVG = Convergence criterion for Davidson eigenvectors. Eigenvector accuracy is proportional to DAVCVG, while the energy accuracy is proportional to its square. The default is 1.0E-05.

CISPRP = Flag to request the determination of CIS level properties, using the relaxed density. Relevant to RUNTYP=ENERGY jobs, although the default is .FALSE. because additional CPHF calculation will be required. Properties are computed as a normal byproduct of runs involving the CIS gradient.

CHFSLV = Chooses type of CPHF solver to use.
= CONJG selects an ordinary preconditioned conjugate gradient solver. This is the default.
= DIIS selects a diis-like iterative solver.

RDCISV = Flag to read CIS vectors from a $CISVEC group in the input file. Default is .FALSE.

MNMEDG = Flag to force the use of the minimal amount of memory in construction of the CIS Hamiltonian diagonal elements. This is only relevant when DGAPRX=.FALSE., and is meant for debug purposes. The default is .FALSE.

MNMEOP = Flag to force the use of the minimal amount of memory during the Davidson iterations. This is for debug purposes. The default is .FALSE.

The additional PC GAMESS (Firefly) specific keywords are:

CPTOL - convergence criterion for CPHF equations, default is 1.0d-5 for GAMESS (US) compatibility.

THRDII - threshold to turn on DIIS if it was selected to solve CPHF. Default is 0.05

MAXGC - maximum allowed number of trial vectors to be routed through GENCON engine, default is 1. If the number of trial vectors is greater than MAXGC, only FASTINTS will be used. The reason is that for moderately contracted GC basis sets like cc-pVXZ, gencon is faster than fastints only for relatively small number of trial vectors (this is by gencon design). On the other hand, for ANO-like basis sets, it is always better to set MAXGC to be equal the number of initial guess vectors, as fastints will be much slower.

ONEDIM = .true./.false. Default is true for abelian groups, false otherwise. Option to select fast algorithm for excited state

properties/gradients for systems with symmetry. Any state
transforming by any pure one-dimension representation on the
point group can be handled by this manner. It is safe to always
turn this option on as the check is performed and this option
disables itself if the symmetry is not appropriate.

PRTTOL - threshold for CIS determinants/csf printout and also for states
symmetry determination. Default is 0.05.

ISTSYM - symmetry of states of interest. Default is zero, i.e., does not
use any symmetry during calculations. Setting this to the
desired index of irrep(according to PC Gamess/Firefly numbering)
will solve
only for the states of the desired symmetry and exploiting full
(including non-abelian) symmetry of molecule, thus significantly
reducing computation time.

ALTER - flag to modify internal logic of Davidson diagonalization code
to use dynamic number of trial vectors. Default is .true.
Setting it to .false. will slow-down calculations by forcing
DAVIDSON diagonalization code to work exactly as in GAMESS (US).

The state-tracking feature of the PC GAMESS' CIS code can be
activated by selecting negative value of istate in the $cis
group. It is intended for geometry optimization of the excited
states in the case of root flipping.
Note that oscillator strengths printed in the CIS summary table
are calculated using transition dipoles length form only.

==========

$CISVEC group required if RDCISV in $CIS is chosen

Below is the sample input file.

```
 $CONTRL SCFTYP=RHF CITYP=CIS $END
 $SYSTEM TIMLIM=3000 MEMORY=3000000 $END
 $BASIS  GBASIS=n31 ngauss=6 NDFUNC=1 $END
 $CIS NSTATE=3 ISTSYM=0 ISTATE=1 $END
 $DATA
H2O
CNV 2

 O           8.0   0.0000000000   0.0000000000   0.7205815395
 H           1.0   0.0000000000   0.7565140024   0.1397092302
 $END
```

==========

$DET group (required for SCFTYP=MCSCF if CISTEP=ALDET)
$CIDET group (required if CITYP=ALDET)

This group describes the determinants to be used in a
full MCSCF active space, or full CI wavefunction.

Determinants contain several spin states, in contrast
to configuration state functions. The Sz quantum number
of each determinant is the same, but the Hamiltonian

eigenvectors will have various spins S=Sz, Sz+1, Sz+2,

In addition, the determinant CI code does not exploit the spatial symmetry of the orbitals, so the CI states will include not only various spin symmetries, but also all space symmetries. You may need to think about NSTATE!

There is no default for NCORE, NACT, and NELS:

NCORE = total number of orbitals doubly occupied in all determinants.

NACT = total number of active orbitals.

NELS = total number of active electrons.

SZ = azimuthal spin quantum number for each of the determinants, two times SZ is therefore the number of excess alpha spins in each determinant. The default is SZ=S, extracted from the MULT=2S+1 given in $CONTRL.

* * * the following control the diagonalization * * *

NSTATE = Number of CI states to be found, the default is 1. The maximum number of states is 100.

PRTTOL = Printout tolerance for CI coefficients, the default is to print any larger than 0.05.

ITERMX = Maximum number of Davidson iterations per root. The default is 100.

CVGTOL = Convergence criterion for Davidson eigenvector routine. This value is proportional to the accuracy of the coeficients of the eigenvectors found. The energy accuracy is proportional to its square. The default is 1.0E-5.

NHGSS = dimension of the Hamiltonian submatrix which is diagonalized to obtain the initial guess eigenvectors. The determinants forming the submatrix are chosen on the basis of a low diagonal energy, or if needed to complete a spin eigenfunction. The default is 300.

NSTGSS = Number of eigenvectors from the initial guess Hamiltonian to be included in the Davidson's iterative scheme. It is seldom necessary to include extra states to obtain convergence to the desired states. The default is the value for NSTATE.

MXXPAN = Maximum number of expansion basis vectors in the iterative subspace during the Davidson iterations before the expansion basis is truncated. The default is the larger of 10 or NSTGSS+1. If this value is decreased, convergence is hurt but the memory required is decreased.

* * * the following control the 1st order density * * *
These are ignored during MCSCF, but are used during a CI.

IROOT = the root whose density is saved on the disk file
         for subsequent property analysis. Only one root
         can be saved, and the default value of 1 means
         the ground state. Be sure to set NFLGDM to form
         the density of the state you are interested in!

NFLGDM = Controls each state's density formation.
         0 -> do not form density for this state.
         1 -> form density and natural orbitals for this
              state, print and punch occ.nums. and NOs.
         2 -> same as 1, plus print density over MOs.
         The default is NFLGDM(1)=1,0,0,...,0 meaning
         only ground state NOs are generated.

    * * * the following control the state averaged
    * * * 1st and 2nd order density matrix computation
Usually ignored by CI runs, these are relevant to MCSCF.

PURES = a flag controlling the spin purity of the state
         avaraging. If true, the WSTATE array pertains
         to the lowest states of the same S value as is
         given by the MULT keyword in $CONTRL. In this
         case the value of NSTATE will need to be bigger
         than the total number of weights given by WSTATE
         if there are other spin states present at low
         energies. If false, it is possible to state
         average over more than one S value, which might
         be of interest in spin-orbit coupling jobs.
         The default is .TRUE.

WSTATE = An array of up to 100 weights to be given to the
         densities of each state in forming the average.
         The default is to optimize a pure ground state,
         WSTATE(1)=1.0,0.0,...,0.0
         A small amount of the ground state can help the
         convergence of excited states greatly.
         Gradient runs are possible only with pure states.
         Be sure to set NSTATE above appropriately!

==========

$DRT group   (required for SCFTYP=MCSCF if CISTEP=GUGA)
$CIDRT group   (required if CITYP=GUGA)

    This group describes the -MCSCF- or -CI- wavefunction.
The distinct row table is the means by which the Graphical
Unitary Group Approach (GUGA) names the configurations.

   The group is spelled DRT for MCSCF runs, and CIDRT for
CI runs. The main difference in these is NMCC vs. NFZC.

    There is no default for GROUP, and you must choose one
of FORS, FOCI, SOCI, or IEXCIT.

GROUP = the name of the point group to be used. This is
        usually the same as that in $DATA, except for

RUNTYP=HESSIAN, when it must be C1. Choose from the following: C1, C2, CI, CS, C2V, C2H, D2, D2H, C4V, D4, D4H. If your $DATA group is not listed, choose only C1 here.

FORS = flag specifying the Full Optimized Reaction Space set of configuration should be generated. This is usually set true for MCSCF runs, but if it is not, see FORS in $MCSCF. (Default=.FALSE.)

FOCI = flag specifying first order CI. In addition to the FORS configurations, all singly excited CSFs from the FORS reference are included. Default=.FALSE.

SOCI = flag specifying second order CI. In addition to the FORS configurations, all singly and doubly excited configurations from the FORS reference are included. (Default=.FALSE.)

IEXCIT= electron excitation level, for example 2 will lead to a singles and doubles CI. This variable is computed by the program if FORS, FOCI, or SOCI is chosen, otherwise it must be entered.

* * the next variables define the single reference * *

The single configuration reference is defined by filling in the orbitals by each type, in the order shown. The default for each type is 0.

Core orbitals, which are always doubly occupied:
NMCC = number of MCSCF core MOs (in $DRT only).
NFZC = number of CI frozen core MOs (in $CIDRT only).

Internal orbitals, which are partially occupied:
NDOC = number of doubly occupied MOs in the reference.
NAOS = number of alpha occupied MOs in the reference, which are singlet coupled with a corresponding number of NBOS orbitals.
NBOS = number of beta spin singly occupied MOs.
NALP = number of alpha spin singly occupied MOs in the reference, which are coupled high spin.
NVAL = number of empty MOs in the reference.

External orbitals, occupied only in FOCI or SOCI:
NEXT = number of external MOs. If given as -1, this will be set to all remaining orbitals (apart from any frozen virtual orbitals).
NFZV = number of frozen virtual MOs, never occupied.

* * * the final choices are seldom used * * *

INTACT= flag to select the interacting space option. The CI will include only those spin couplings which have a nonvanishing matrix element with the reference configuration.

MXNINT = Buffer size for sorted integrals. (default=20000)

```
MXNEME = Buffer size for energy matrix.  (default=10000)

NPRT   = Configuration printout control switch.
         This can consume a HUMUNGUS amount of paper!
         0 = no print (default)
         1 = print electron occupancies, one per line.
         2 = print determinants in each CSF.

=============================================================

$MCSCF group                          (optional for -MCSCF-)

    This group controls the MCSCF orbital optimization
step. The difference between the four convergence methods
is outlined in Chapter Three of this manual, which you must
carefully study before attempting MCSCF computations.

OPTACT = .False.   Default
       = .True.    Activates minor changes in the MCSCF FULLNR
                   converger. This may somehow help in the case of a
                   slow convergence of FULLNR MCSCF for the non FORS-
                   type wavefunction.

 --- the next choose the configuration basis ---

CISTEP = ALDET chooses the Ames Lab. determinant CI, and
               requires $DET input. (default)
       = GUGA  chooses the graphical unitary group CSFs,
               and requires $DRT input.  This is the
               only value usable with the QUAD converger.

 --- the next four choose the orbital optimizer ---

FOCAS  = a flag to select a method with a first order
         convergence rate.  (default=.FALSE.)

SOSCF  = a flag selecting an approximately second order
         convergence method.  (default=.TRUE.)

FULLNR = a flag selecting a second order method, with an
         exact orbital hessian.  (default=.FALSE.)

QUAD   = a flag to pick a fully quadratic (orbital and
         CI coefficient) optimization method, which is
         applicable to FORS or non-FORS wavefunctions.
         QUAD may not be used with state-averaging.
         (default = .FALSE.)

Note that FOCAS must be used only with FORS=.TRUE. in $DRT.


The other convergers are usable for either FORS or non-FORS
wavefunctions, although convergence is always harder in the
latter case, when FORS below must be set .FALSE.

   --- the next apply to all convergence methods ---

FORS   = a flag to specify that the MCSCF function is of
         the Full Optimized Reaction Space type, which is
         sometimes known as CAS-SCF.  .TRUE. means omit
```

act-act rotations from the optimization. Since convergence is usually better for FULLNR with these rotations included, the default is sensible for the case FORS=.TRUE. in $DRT. (default is .TRUE. for FOCAS/SOSCF, .FALSE. for FULLNR/QUAD)

ACURCY = the major convergence criterion, the maximum permissible asymmetry in the Lagrangian matrix. (default=1.0E-05)

ENGTOL = a secondary convergence criterion, the run is considered converged when the energy change is smaller than this value. (default=1.0E-10)

MAXIT = Maximum number of iterations (default=100 for FOCAS, 60 for SOSCF, 30 for FULLNR or QUAD)

MICIT = Maximum number of microiterations within a single MCSCF iteration. (default=5 for FOCAS or SOSCF, or 1 for FULLNR or QUAD)

NWORD = The maximum memory to be used, the default is to use all available memory. (default=0)

CANONC = a flag to cause formation of the closed shell Fock operator, and generation of canonical core orbitals. This will order the MCC core by their orbital energies. (default=.TRUE.)

EKT = a flag to cause generation of extended Koopmans' theorem orbitals and energies. (Default=.FALSE.)
For this option, see R.C.Morrison and G.Liu, J.Comput.Chem., 13, 1004-1010 (1992). Note that the process generates non-orthogonal orbitals, as

well as physically unrealistic energies for the weakly occupied MCSCF orbitals. The method is meant to produce a good value for the first I.P.

NPUNCH = MCSCF punch option (analogous to $SCF NPUNCH)
0 do not punch out the final orbitals
1 punch out the occupied orbitals
2 punch out occupied and virtual orbitals
The default is NPUNCH = 2.

NPFLG = an array of debug print control. This is analagous to the same variable in $CIINP. Elements 1,2,3,4,6,8 make sense, the latter controls debugging the orbital optimization.

--- the next refers to SOSCF optimizations ---

NOFO = set to 1 to skip use of FOCAS for one iteration during SOSCF. This is a testing parameter, at present NOFO defaults to 0 to do one FOCAS iter.

--- the next three refer to FOCAS optimizations ---

CASDII = threshold to start DIIS (default=0.05)

CASHFT = level shift value (default=1.0)

NRMCAS = renormalization flag, 1 means do Fock matrix
renormalization, 0 skips (default=1)

--- the next applies to the QUAD method ---
(note that FULLNR input is also relevant)

QUDTHR = threshold on the orbital rotation parameter,
SQCDF, to switch from the initial FULLNR
iterations to the fully quadratic method.
(default = 0.05)

--- all remaining input applies only to FULLNR ---

DAMP = damping factor, this is adjusted by the program
as necessary. (default=0.0)

METHOD = DM2 selects a density driven construction of the
Newton-Raphson matrices. (default).
= TEI selects 2e- integral driven NR construction.
See the 'Further information' chapter for more
details concerning these methods. TEI is slow!

LINSER = a flag to activate a method similar to direct
minimization of SCF. The method is used if
the energy rises between iterations. It may in
some circumstances increase the chance of
converging excited states. (default=.FALSE.)

FCORE = a flag to freeze optimization of the MCC core
orbitals, which is useful in preparation for
RUNTYP=TRANSITN jobs. Setting this flag will
automatically force CANONC false. This option
is incompatible with gradients, so can only be
used with RUNTYP=ENERGY. (default=.FALSE.)

--- the next four are seldom used ---

DROPC = a flag to include MCC core orbitals during the
CI computation. The default is to drop them
during the CI, instead forming Fock operators
which are used to build the correct terms in
the orbital hessian. (default = .TRUE.)

NORB = the number of orbitals to be included in the
optimization, the default is to optimize with
respect to the entire basis. This option is
incompatible with gradients, so can only be used
with RUNTYP=ENERGY. (default=number of AOs
given in $DATA).

MOFRZ = an array of orbitals to be frozen out of the
orbital optimization step (default=none frozen).

```
NOROT  = an array of up to 250 pairs of orbital rotations
         to be omitted from the NR optimization process.
         The program automatically deletes all core-core
         rotations, all act-act rotations if FORS=.T.,
         and all core-act and core-virt rotations if
         FCORE=.T.  Additional rotations are input as
         I1,J1,I2,J2... to exclude rotations between
         orbital I running from 1 to NORB, and J running
         up to the smaller of I or NVAL in $TRANS.
```

==========

```
$MCQDPT group       (relevant to SCFTYP=MCSCF if MPLEVL=2)
$XMCQDPT group      (relevant to SCFTYP=MCSCF if MPLEVL=2)
```

Controls 2nd order MCQDPT (multiconfiguration quasi-degenerate perturbation theory) runs, if requested by MPLEVL=2 in $CONTRL. MCQDPT2 is implemented only for FORS (aka CASSCF) wavefunctions. The MCQDPT method is a multistate, as well as multireference perturbation theory. A more complete discussion may be found in the 'Further information' chapter. Analytic gradients are not available.

*** MCSCF reference wavefunction ***

```
NEL    =   total number of electrons, including core.
           (default from $DATA and ICHARG in $CONTRL)
MULT   =   spin multiplicity (default from $CONTRL)
NMOACT =   Number of orbitals in FORS active space
           (default is the active space in $DET or $DRT)
NMOFZC =   number of frozen core orbitals, NOT correlated
           in the perturbation calculation.  (default is
           number of chemical cores)
NMODOC =   number of orbitals which are doubly occupied in
           every MCSCF configuration, that is, not active
           orbitals, which are to be included in the
           perturbation calculation.  (The default is all
           valence orbitals between the chemical core and
           the active space)
NMOFZV =   number of frozen virtuals, NOT occupied during
           the perturbation calculation.  The default is
           to use all virtuals in the MP2.  (default=0)
ISTSYM =   the state symmetry of the target state(s).
           This is given as an integer, note that only
           Abelian groups are supported in $DATA:
             ISTSYM= 1   2   3   4   5   6   7   8
                C1   A
                Ci   Ag  Au
                Cs   A'  A''
                C2   A   B
                C2v  A1  A2  B1  B2
                C2h  Ag  Bg  Au  Bu
                D2   A   B1  B2  B3
                D2h  Ag  B1g B2g B3g Au  B1u B2u B3u
           (The default is 1, the totally symmetric state)
```

*** perturbation specification ***

KSTATE= state is used (1) or not (0) in the MCQDPT2.
         Maximum of 20 elements, including zeros.
         For example, if you want the perturbation
         correction to the second and the fourth roots,
             KSTATE(1)=0,1,0,1
         (default=1,0,0,0,0,0,0,...)

       *** MO input and flow control ***

INORB = 0 optimize the MCSCF wavefunction in this run.
       = 1 read the converged orbitals from a $VEC group,
           and skip immediately to the MCQDPT computation.
           A complete $VEC including virtuals must be given.
           (default=0)

       *** Canonical Fock orbitals ***

IFORB = 1 determine the canonical Fock orbitals
       = 0 omit this step. (default=1)
AVECOE =   weight of each CAS-CI state in computing the
           closed shell Fock matrix.
           (default is AVECOE(1)=1.0,1.0,1.0,...)

       *** Intruder State Removal ***

EDSHFT =   energy denominator shift. (default=0.0)
  Intruder State Avoidance (ISA) calculations
  can be made by changing the energy denominators
  around poles (where the denominator is zero).
  Each denominator x is replaced by x + EDSHFT/x,
  so that far from the poles (when x is large) the
  effect of such change is small. A suggested
  value is 0.02, but experimentation with your system
  is recommended. Setting this value to zero is ordinary
  MCQDPT, and infinite collapses to the MCSCF reference.
  Note that the energy denominators (which are ket-dependent
  in MCQDPT) are changed in a different way for each
  ket-vector, that is, for each row in MCQDPT
  Hamiltonian matrix. In other words, the zeroth
  order energies are not "universal", but state
  specific. This is strictly speaking some weak
  inconsistency in defining zeroth order energies
  that are usually chosen "universally".

  In order to maintain continuity when studying
  a PES, one usually uses the same EDSHFT value
  for all points on PES. In order to study the
  potential surface for any extended range of
  geometries, it is recommended to use ISA, as it
  is quite likely that one or more regions of the
  PES will be unphysical due to intruder states.

  For an example of how intruder states can appear
  at some points on the PES, see Figures 1,2,7 of
     K.R.Glaesemann, M.S.Gordon, H.Nakano
     Phys.Chem.Chem.Phys. 1, 967-975(1999)
  and also

H.A.Witek, D.G.Fedorov, K.Hirao, A.Viel,
P.-O.Widmark J.Chem.Phys. 116, 8396-406(2002)
For a discussion of intruder state removal from
MCQDPT, see
H.A.Witek, Y.-K.Choe, J.P.Finley, K.Hirao
J.Comput.Chem. 23, 957-965(2002)

*** Miscellaneous options ***

THRGEN = threshold for one-, two-, and three-body density matrix elements in the perturbation calculation. If you want to obtain energies, for instance, to 6 figures after point, choose THRGEN=1.0D-08 or 1.0D-09. (default=1.0D-08)
THRENE = threshold for the energy convergence in the Davidson's method CAS-CI. (default=-1.0D+00)
THRCON = threshold for the vector convergence in the Davidson's method CAS-CI. (default=1.0D-06)
LPOUT = print option, 0 gives normal printout, while <0 gives debug print (e.g. -1,-5,-10,-100) (default=0)

Finally, there are additional very specialized input options, described in the source code routine MQREAD:

IROT, ISELCT, LENGTH, MAXCSF, MAXERI, MAXROW, MDI, MXBASE, MXTRFR, NSOLUT, NSTOP, GENZRO, THRERI, THRWGT, MAINCS, NSTATE, NSTCI

==========

$MCQFIT group

The presence of this group in the input file triggers the new MCQDPT2 code. The main purpose of the new MCQDPT2 code is to speed up calculations in the cases of medium to large basis sets and medium to large active spaces. It is not efficient for small active spaces (e.g., less than 500 CSF) and/or for small number of basis functions (e.g., less than 50). In the latter cases it is recommended to use the default PC GAMESS' MCQDPT2 code. It is also not recommended to use new code for calculations which do not use ISA energy denominators shift, as in the cases of intruder states new code can become numerically unstable. On the other hand, we recommend to use ISA technique/nonzero edshft parameter for any MCQDPT2 run.
In most cases, it is sufficient to just specify $mcqfit $end without altering any parameters of the $mcqfit group - the default parameters were selected to introduce errors in the computed energies which is typically less than 10-8 Hartree.

For better efficiency of the new MCQDPT2 code running in SMP mode, we recommend to manually increase the default value of the length parameter of $mcqdpt group to be equal to the number of CSF generated by your run.

DELTAE - double precision, default is zero. If nonzero, this value is used to define parameters of the interpolation grid.

NPOINT - integer, default is 400. Together with deltae this value is used to define parameters of the interpolation grid. Actual grid will be the smaller of npoint value and the number of points computed based on the deltae value. Increasing npoint will result in increase of required CPU time but will reduce errors in computed energy.

IORDER - integer, default value is 7. Available values are 3, 5, and 7.
Defines the order of polynomes used for interpolation.

==========

$CISORT group (optional, relevant for -CI- and -MCSCF-)

This group provides further control over the sorting of the transformed integrals.

NDAR = Number of direct access records.
(default = 2000)

LDAR = Length of direct access record (site dependent)

NBOXMX = Maximum number of boxes in the sort.
(default = 200)

NWORD = Number of words of fast memory to use in this step. A value of 0 results in automatic use of all available memory. (default = 0)

NOMEM = 0 (set to one to force out of memory algorithm)
MOINTS = DELETE Delete MOINTS file after sorting

==========

$GUGDRT group

NWORD = Controls the memory usage during generation of DRT. It may be necessary to increase the default value of 180018 DP words when the active space used in CI/MCSCF calculation is too large.

==========

$GUGEM group (optional, relevant for -CI- or -MCSCF-)

This group provides further control over the calculation of the energy (Hamiltonian) matrix.

CUTOFF = Cutoff criterion for the energy matrix.
(default=1.0E-8)

NWORD = not used.

FASTCI =
= .False. Normal operation
= .True. Turns on the packing of GUGA CI Hamiltonian file. Both indices and matrix elements are packed. This option reduces the size of WORK16 file by a factor of up to 2.7-3, thus saving the disk space and the CPU time. See also a more detailed discussion on the packing-related problems. (default)

PKTHR = A threshold to activate packing of the matrix elements of GUGA CI Hamiltonians. Default is 10^7*CUTOFF

PACK2 =.False. Normal operation (default)

=.True. Turns on the additional (secondary) packing of GUGA CI Hamiltonian file, which works in conjunction with the FASTCI method of compression. This is a lossless method, which generally reduces the size of WORK16 file by a factor of up to 3-6 additionally to the FASTCI squeezing. Thus, the total degree of compression can reach the factor of 10-20 and even more, especially for the non-FORS type CI. Hence, this option is very useful in the case of relatively large CI calculations (i.e. 50000 CSFs or more, up to several millions), saving the disk space and reducing the real execution time. Currently, the fastest way to perform any large CI/MCSCF calculation is to use PACK2 method of packing. The only exception is the case of GUGA CI Hamiltonians of medium size, that can entirely reside in the system file cache (in the case of Win95, WinNT, and Linux, the size of the latter is limited only by the amount of the physical memory currently unused). For such jobs, the default packing settings are still faster.
It is often faster to perform a CI energy calculation as the first iteration of the approximate second-order MCSCF program. This is due to the fact that the SOSCF MCSCF integral transformation can be significantly faster as compared to the generic one.
Relevant only when FASTCI=.True.

LCHAIN = A parameter that controls the degree of compression achieved by the PACK2 packing method. The greater the LCHAIN value, the better the compression and the higher the CPU overhead. Note also that the most efficient packing is always achieved when LCHAIN=0, because this special value is simply an alias for the infinitely large LCHAIN. Relevant only when PACK2=.True. Default is 7, which seems to be quite reasonable.

DIRCI = .True. Performs the direct GUGA CI calculation, in which the Hamiltonian matrix is not stored on disk, and the matrix elements are recalculated during each Davidson diagonalization step. This may consume considerably more time than normal GUGA CI calculation, but allows one to perform it in the case when the Hamiltonian matrix is too large to fit on disk.

= .False. Performs conventional GUGA CI calculation (default)

==========

$GUGDIA group (optional, relevant for -CI- or -MCSCF-)

This group provides control over the Davidson method diagonalization step.

NSTATE = Number of CI states to be found. (default=1)

```
         You can solve for any number of states, but only
         100 can be saved for subsequent sections, such
         as state averaging.

PRTTOL = Printout tolerance for CI coefficients
         (default = 0.05)

MXXPAN = Maximum no. of expansion basis vectors used
         before the expansion basis is truncated.
         (default=30)

ITERMX = Maximum number of iterations (default=50)

CVGTOL = Convergence criterion for Davidson eigenvector
         routine.  This value is proportional to the
         accuracy of the coeficients of the eigenvector(s)
         found.  The energy accuracy is proportional to
         its square.  (default = 1.0E-5)

NWORD  = Number of words of fast memory to use in this
         step.  A value of zero results in the use of all
         available memory.  (default = 0)

MEMMAX = 1000*limgiv + limh where limgiv is the largest
         matrix diagonalized via Givens-Householder
         (default=50) and limh is the dimension of the
         largest Hamiltonian that may be memory resident
         (default=100)

NIMPRV = Maximum no. of eigenvectors to be improved every
         iteration. (default = nstate)

NSELCT = Determines initial guess to eigenvectors.
         = 0 ->  Unit vectors corresponding to the NSTATE
                 lowest diagonal elements and any diagonal
                 elements within SELTHR of them. (default)
         < 0 ->  First abs(NSELCT) unit vectors.
         > 0 ->  use NSELCT unit vectors corresponding to
                  the NSELCT lowest diagonal elements.

SELTHR = Guess selection threshold when NSELCT=0.
         (default=0.01)

NEXTRA = Number of extra expansion basis vectors to be
         included on the first iteration.  NEXTRA is
         decremented by one each iteration.  This may be
         useful in "capturing" vectors for higher states.
         (default=5)

KPRINT = Print flag bit vector used when
         NPFLG(4)=1 in the $CIINP group       (default=8)
         value  1 bit 0 print final eigenvalues
         value  2 bit 1 print final tolerances
         value  4 bit 2 print eigenvalues and tolerances
                        at each truncation
         value  8 bit 3 print eigenvalues every iteration
         value 16 bit 4 print tolerances every iteration
```

==========

$GUGDM group (optional, relevant for -CI-)

This group provides further control over formation of the 1-particle density matrix.

NFLGDM = Controls each state's density formation.
0 -> do not form density for this state.
1 -> form density and natural orbitals for this state, print and punch occ.nums. and NOs.
2 -> same as 1, plus print density over MOs.
(default=1,99*0, meaning G.S. NOs only)
See also NSTATE in $GUGDIA. Note that forming the 1-particle density for a state is negligible against the diagonalization time for that state.

IROOT = The -CI- root whose density matrix is saved on the direct access dictionary file for later computation of properties. (default=1)

IBLOCK = Density blocking switch. If nonzero, the off diagonal block of the density below row IBLOCK will be set to zero before the (approximate) natural orbitals are found. One use for this is to keep the internal and external orbitals in a FOCI or SOCI calculation from mixing.
(default=0)

NWORD = Number of words of fast memory to use in this step. A value of zero uses all available memory (default=0).

==========

$GUGDM2 group (optional, relevant for -CI- or -MCSCF-)

This group provides control over formation of the 2-particle density matrix.

WSTATE = An array of up to 100 weights to be given to the 2 body density of each state in forming the DM2. The default is to optimize a pure ground state.
(Default=1.0,99*0.0)
A small amount of the ground state can help the convergence of excited states greatly.
Gradient runs are possible only with pure states.

Be sure to set NSTATE in $GUGDIA appropriately!

CUTOFF = Cutoff criterion for the 2nd-order density.
(default = 1.0E-9)

NWORD = Number of words of fast memory to use in sorting the DM2. The default uses all available memory.
(default=0).

NOMEM = 0 uses in memory sort, if possible.
= 1 forces out of memory sort.

NDAR = Number of direct access records. (default=4000)

LDAR   = Length of direct access record (site dependent)

NBOXMX = Maximum no. of boxes in the sort. (default=200)

==========

$LAGRAN group   (optional, relevant for -CI- gradient)

     This group provides further control over formation of the CI Lagrangian, a quantity which is necessary for the computation of CI gradients.

```
 NOMEM =   0 form in core, if possible
       =   1 forces out of core formation
 NWORD =   0 (0=use all available memory)
 NDAR  = 4000
 LDAR  = Length of each direct access record
         (default is NINTMX from $INTGRL)
```

==========

$TRFDM2 group   (optional, relevant for -CI- gradient)

     This group provides further control over the back transformation of the 2 body density to the AO basis.

```
 NOMEM =   0 transform and sort in core, if possible
       =   1 transform in core, sort out of core, if poss.
       =   2 transform out of core, sort out of core
 NWORD =   0 (0=use all available memory)
 CUTOFF= 1.0D-9, threshold for saving DM2 values
 NDAR  = 2000
 LDAR  = Length of each direct access record
         (default is system dependent)
 NBOXMX= 200
```

==========

Usually neither of these two groups is given. Since these groups are normally used only for CI gradient runs, we list here some of the restrictions on the CI gradients:

  a) SCFTYP=RHF, only
  b) no FZV orbitals in $CIDRT, all MOs must be used.
  c) the derivative integrals are computed in the 2nd derivative code, which is limited to spd basis sets.
  d) the code does not run in parallel.
  e) Use WSTATE in $GUGDM2 to specify the state whose gradient is to be found. Use IROOT in $GUGDM to specify the state whose other properties will be found. These must be the same state!
  f) excited states often have different symmetry than the ground state, so think about GROUP in $CIDRT.
  g) the gradient can probably be found for any CI for which you have sufficient disk to do the CI itself. Time is probably about 2/3 additional.

==========

$TRANST group (relevant for RUNTYP=TRANSITN)
(only for CITYP=GUGA) (relevant for RUNTYP=SPINORBT)

This group controls the evaluation of the radiative transition moment, or spin orbit coupling. The defaults assume that there is one common set of orbitals, all of which are occupied. The program can use two separately optimized MO sets, provided certain conditions are met.

NUMVEC = the number of different MO sets. This can be either 1 or 2. (default=1)

NUMCI = the number of different CI calculations to do. This should equal NUMVEC for TRANSITN, and is usually 2 or greater for SPINORBT (1 if the multiplicities of the states are the same). (default=1)

NOCC = the number of occupied orbitals. The default is the number of AOs given in $DATA, which is not usually correct.

NFZC = When NUMVEC=2, this is the number of identical core orbitals in the two vector sets, and most likely this is NFZC in the $CDIRT groups. When NUMVEC=1, and for RUNTYP=TRANSITN, NFZC should equal NOCC, whereas for RUNTYP=SPINORBT, it should equal the NFZC value in the $CIDRTs. The default is the number of AOs given in $DATA, again this is not very reasonable.

IROOTS = array containing the number of CI states for which the transition moments are to be found. The default is 1 for each CI, which is probably not a correct choice for TRANSITN runs, but is quite reasonable for SPINORBT.

NSTATE = array of CI states to be found when diagonalising the CI Hamiltonian. Of those, the first IROOTS(i) will be used to find transition moments. NSTATE in $GUGDIA overrides NSTATE(i), if it is larger.

Reasonable values for TRANSITN runs where you want to know the moments for the first 4 states are IROOTS(1)=4 NSTATE(1)=8 for the case of a common orbital set, or IROOTS(1)=1,4 NSTATE(1)=8,8 for two sets of orbitals.

* * * the next seven pertain only to spin-orbit runs * * *

METHOD = ZEFF regards the nuclear charge appearing in the one electron operator as a scaling parameter to compensate for omitting the two electron term altogether. (default)
= BREIT includes the full 1e- and 2e- operator. Only singlet and triplet states can be coupled. If you ask for more than one triplet state (iroots(2)>1), then no Hso diagonalisation will be performed.

```
                This code is presently limited to a
                maximum of 10 orbitals (32 bit machines)
                or about 16 orbitals (64 bit machines)
                in the active space.

ZEFTYP   specifies what Zeff charges to use
       = 3-21G  selects a set obtained by 3-21G, but
                probably approriate for any all electron
                basis set
       = SBK    selects a set obtained for the SBKJC ECP
                basis set, specifically.

ZEFF   = an array of effective nuclear charges overriding
         the charges chosen in ZEFTYP.  This pertains only
         to METHOD=ZEFF SPINORBT runs.
         (default: true charges with any effective core
          potential protons removed if ECPs are used).

TMOMNT = flag to control spin-orbit transition moment
         calculation (default is .FALSE.) (only with ZEFF)

SAVDSK = flag to repeat the form factor calculation twice.
         This saves disk space if limited storage is
         available.  (default=.FALSE.) (only with BREIT)

ACTION   control of disk file DAFL30 reuse.
       = NORMAL calculate the form factors in this run.
       = SAVE   calculate, and store the form factors on
                disk for future runs with the same active
                space characteristics.
       = READ   read the form factors from disk from an
                earlier run which used SAVE.
         (default=NORMAL) (only with BREIT)

MS     = limit spin projection of the triplet state
         0 do matrix elements for ms=0 only
         1 do matrix elements for ms=1 only
        -2 do matrix elements for all ms (0,1 and -1)
        -3 do not calculate any matrix element
           (ie useful for saving form factors)
        (default=-2) (only BREIT)

* * * the remaining parameters are not so important * * *

PRTCMO = flag to control printout of the corresponding
         orbitals.  (default is .FALSE.)

TOLZ   = MO coefficient zero tolerance (as for $GUESS).
         (default=1.0E-8)

TOLE   = MO coefficient equating tolerance (as for
         $GUESS).  (default=1.0E-5)

PRTPRM = flag to provide detailed information about the
         composition of the spin-mixed states in terms of
         adiabatic states. (default is .FALSE.) (only ZEFF)

PRMTOL = threshold for Slater determinant output.
         (default is 0.0) (only ZEFF)

=============================================================
```

* *

For information on TRANSITN and SPINORBT,
see the 'Further information' section.

* *

****
*
* CHAPTER III - Further Information*
*
****

This chapter of the manual contains both references, and hints on how to do things. The following is a list of the topics covered:

- Computational References.
- History of GAMESS (US).
- Basis Set References, and descriptions.
- How to do RHF, ROHF, UHF, and GVB calculations.
  General considerations — Other open shell SCF cases
  Direct SCF — True GVB perfect pairing
  SCF convergence methods — The special case of TCSCF
  High spin open shell SCF — A caution about symmetry
- How to do MCSCF and CI calculations.
  MCSCF implementation — CSF CI
  Orbital updates — starting orbitals
  CI coefficient optimization — references
  determinant CI
- Second order perturbation theory.
- Geometry Searches and Internal Coordinates.
  Quasi-Newton searches — Practical matters
  The nuclear hessian — Saddle points
  Coordinate choices — Mode following
  The role of symmetry
- Intrinsic Reaction Coordinate (IRC) methods
- Gradient Extremals
- Continuum solvation methods: SCRF and PCM
- Effective Fragment Potential method
  Terms in an EFP — Current Limitations
  Constructing an EFP — Practical Hints
- MOPAC calculations within PC GAMESS
- Molecular Properties, and conversion factors.

- Localization tips.

- Transition moments and spin-orbit coupling.

Computational References

PC GAMESS/Firefly -
http://classic.chem.msu.su/gran/gamess/index.html

GAMESS (US) -
M.W.Schmidt, K.K.Baldridge, J.A.Boatz, S.T.Elbert,
M.S.Gordon, J.H.Jensen, S.Koseki, N.Matsunaga,
K.A.Nguyen, S.Su, T.L.Windus, M.Dupuis, J.A.Montgomery
J.Comput.Chem. 14, 1347-1363 (1993)

HONDO -
These papers describes many of the algorithms in detail,
and much of these applies also to the legacy GAMESS (US) code
in Firefly:
"The General Atomic and Molecular Electronic Structure
System: HONDO 7.0" M.Dupuis, J.D.Watts, H.O.Villar,
G.J.B.Hurst Comput.Phys.Comm. 52, 415-425(1989)
"HONDO: A General Atomic and Molecular Electronic
Structure System" M.Dupuis, P.Mougenot, J.D.Watts,
G.J.B.Hurst, H.O.Villar in "MOTECC: Modern Techniques
in Computational Chemistry" E.Clementi, Ed.
ESCOM, Leiden, the Netherlands, 1989, pp 307-361.
"HONDO: A General Atomic and Molecular Electronic
Structure System" M.Dupuis, A.Farazdel, S.P.Karna,
S.A.Maluendes in "MOTECC: Modern Techniques in
Computational Chemistry" E.Clementi, Ed.
ESCOM, Leiden, the Netherlands, 1990, pp 277-342.
M.Dupuis, S.Chin, A.Marquez in Relativistic and Electron
Correlation Effects in Molecules, G.Malli, Ed. Plenum
Press, NY 1994.

sp integrals and gradient integrals -
J.A.Pople, W.J.Hehre J.Comput.Phys. 27, 161-168(1978)
H.B.Schlegel, J.Chem.Phys. 90, 5630-5634(1989)

spdfg integrals -
"Numerical Integration Using Rys Polynomials"
H.F.King and M.Dupuis J.Comput.Phys. 21,144(1976)
"Evaluation of Molecular Integrals over Gaussian
Basis Functions"
M.Dupuis,J.Rys,H.F.King J.Chem.Phys. 65,111-116(1976)
"Molecular Symmetry and Closed Shell HF Calculations"
M.Dupuis and H.F.King Int.J.Quantum Chem. 11,613(1977)
"Computation of Electron Repulsion Integrals using
the Rys Quadrature Method"
J.Rys,M.Dupuis,H.F.King J.Comput.Chem. 4,154-157(1983)

spdfg gradient integrals -
"Molecular Symmetry. II. Gradient of Electronic Energy
with respect to Nuclear Coordinates"
M.Dupuis and H.F.King J.Chem.Phys. 68,3998(1978)
although the implementation is much newer than this paper.

spd hessian integrals -
"Molecular Symmetry. III. Second derivatives of Electronic
Energy with respect to Nuclear Coordinates"
T.Takada, M.Dupuis, H.F.King
J.Chem.Phys. 75, 332-336 (1981)

the Q matrix, and integral transformation symmetry -
E.Hollauer, M.Dupuis J.Chem.Phys. 96, 5220 (1992)

Effective core potentials (ECP) -
C.F.Melius, W.A.Goddard Phys.Rev.A, 10,1528-1540(1974)
L.R.Kahn, P.Baybutt, D.G.Truhlar
J.Chem.Phys. 65, 3826-3853 (1976)
M.Krauss, W.J.Stevens Ann.Rev.Phys.Chem. 35, 357-385(1985)
J.Breidung, W.Thiel, A.Komornicki
Chem.Phys.Lett. 153, 76-81(1988)
B.M.Bode, M.S.Gordon J.Chem.Phys. 111, 8778-8784(1999)
See also the papers listed for SBKJC and HW basis sets.

Quantum fast multipole method (QFMM) -
E.O.Steinborn, K.Ruedenberg
Adv.Quantum Chem. 7, 1-81(1973)
L.Greengard "The Rapid Evaluation of Potential Fields in
Particle Systems" (MIT, Cambridge, 1987)
C.H.Choi, J.Ivanic, M.S.Gordon, K.Ruedenberg
J.Chem.Phys. 111, 8825-8831(1999)
C.H.Choi, K.Ruedenberg, M.S.Gordon
J.Comput.Chem. 22, 1484-1501(2001)
C.H.Choi J.Chem.Phys. 120, 3535-3543(2004)

RHF -
C.C.J. Roothaan Rev.Mod.Phys. 23, 69(1951)

UHF -
J.A. Pople, R.K. Nesbet J.Chem.Phys 22, 571 (1954)

GVB and OCBSE-ROHF -
F.W.Bobrowicz and W.A.Goddard, in Modern Theoretical
Chemistry, Vol 3, H.F.Schaefer III, Ed., Chapter 4.

ROHF -
R.McWeeny, G.Diercksen J.Chem.Phys. 49,4852-4856(1968)
M.F.Guest, V.R.Saunders, Mol.Phys. 28, 819-828(1974)
J.S.Binkley, J.A.Pople, P.A.Dobosh
Mol.Phys. 28, 1423-1429 (1974)
E.R.Davidson Chem.Phys.Lett. 21,565(1973)
K.Faegri, R.Manne Mol.Phys. 31,1037-1049(1976)
H.Hsu, E.R.Davidson, and R.M.Pitzer
J.Chem.Phys. 65,609(1976)

GVB and low spin coupling ROHF -
F.W.Bobrowicz and W.A.Goddard, in Modern Theoretical
Chemistry, Vol 3, H.F.Schaefer III, Ed., Chapter 4.

MCSCF - see reference list in the subsection below

closed, unrestricted open shell 2nd order Moller-Plesset -
J.A.Pople, J.S.Binkley, R.Seeger

Int. J. Quantum Chem. S10, 1-19(1976)
M.J.Frisch, M.Head-Gordon, J.A.Pople,
Chem.Phys.Lett. 166, 275-280(1990)

open shell MP2, so called ZAPT method -
T.J.Lee, D.Jayatilaka Chem.Phys.Lett. 201, 1-10(1993)
T.J.Lee, A.P.Rendell, K.G.Dyall, D.Jayatilaka
J.Chem.Phys. 100, 7400-7409(1994)

open shell MP2, so called RMP method -
P.J.Knowles, J.S.Andrews, R.D.Amos, N.C.Handy, J.A.Pople
Chem.Phys.Lett. 186, 130-136 (1991)
W.J.Lauderdale, J.F.Stanton, J.Gauss, J.D.Watts,
R.J.Bartlett Chem.Phys.Lett. 187, 21-28(1991)

multiconfigurational quasidegenerate perturbation theory -
H.Nakano, J.Chem.Phys. 99, 7983-7992(1993)

GUGA CI -
B.Brooks and H.F.Schaefer J.Chem. Phys. 70,5092(1979)
B.Brooks, W.Laidig, P.Saxe, N.Handy, and H.F.Schaefer,
Physica Scripta 21,312(1980).

Determinant full CI (ALDET) -
J.Ivanic, K.Ruedenberg Theoret.Chem.Acc. 106, 339-351(2001)

CIS energy and gradient -
J.B.Foresman, M.Head-Gordon, J.A.Pople, M.J.Frisch
J.Phys.Chem. 96, 135-149(1992)
R.M.Shroll, W.D.Edwards
Int.J.Quantum Chem. 63, 1037-1049(1997)
S.P.Webb
Theoret.Chem.Acc. 116, 355-372(2006)

Direct SCF -
J.Almlof, K.Faegri, K.Korsell
J.Comput.Chem. 3, 385-399 (1982)
M.Haser, R.Ahlrichs
J.Comput.Chem. 10, 104-111 (1989)

DIIS (Direct Inversion in the Iterative Subspace) -
P.Pulay J.Comput.Chem. 3, 556-560(1982)

SOSCF -
T.H.Fischer, J.Almlof, J.Phys.Chem. 96,9768-74(1992)
G. Chaban, M. W. Schmidt, M. S. Gordon
Theor.Chem.Acc. 97, 88-95(1997)

Modified Virtual Orbitals (MVOs) -
C.W.Bauschlicher, Jr. J.Chem.Phys. 72,880-885(1980)

MOPAC 6 -
J.J.P.Stewart J.Computer-Aided Molecular Design
4, 1-105 (1990)
References for parameters for individual atoms may be
found on the printout from your runs.

RHF/ROHF/TCSCF coupled perturbed Hartree Fock -
"Single Configuration SCF Second Derivatives on a Cray"

H.F.King, A.Komornicki in "Geometrical Derivatives of Energy Surfaces and Molecular Properties" P.Jorgensen J.Simons, Ed. D.Reidel, Dordrecht, 1986, pp 207-214.
Y.Osamura, Y.Yamaguchi, D.J.Fox, M.A.Vincent, H.F.Schaefer J.Mol.Struct. 103, 183-186 (1983)
M.Duran, Y.Yamaguchi, H.F.Schaefer J.Phys.Chem. 92, 3070-3075 (1988)
"A New Dimension to Quantum Chemistry" Y.Yamaguchi, Y.Osamura, J.D.Goddard, H.F.Schaefer Oxford Press, NY 1994

Davidson eigenvector method -
E.R.Davidson J.Comput.Phys. 17,87(1975) and "Matrix Eigenvector Methods" p. 95 in "Methods in Computational Molecular Physics" ed. by G.H.F.Diercksen and S.Wilson

Energy orbital localization -
C.Edmiston, K.Ruedenberg Rev.Mod.Phys. 35, 457-465(1963).
R.C.Raffenetti, K.Ruedenberg, C.L.Janssen, H.F.Schaefer, Theoret.Chim.Acta 86, 149-165(1993)

Boys orbital localization -
S.F.Boys, "Quantum Science of Atoms, Molecules, and Solids" P.O.Lowdin, Ed, Academic Press, NY, 1966, pp 253-262.

Population orbital localization -
J.Pipek, P.Z.Mezey J.Chem.Phys. 90, 4916(1989).

Mulliken Population Analysis -
R.S.Mulliken J.Chem.Phys. 23, 1833-1840, 1841-1846, 2338-2342, 2343-2346(1955)

so called "Lowdin Population Analysis" -
This should be described as "a Mulliken population analysis (ref M1-M4 above) based on symmetrically orthogonalized orbitals (ref L)", where reference L is
P.-O.Lowdin Adv.Chem.Phys. 5, 185-199(1970)
Lowdin populations are not invariant to rotation if the basis set used is Cartesian d,f,...:
I.Mayer, Chem.Phys.Lett. 393, 209-212(2004).

Bond orders and valences -
M.Giambiagi, M.Giambiagi, D.R.Grempel, C.D.Heymann J.Chim.Phys. 72, 15-22(1975)
I.Mayer, Chem.Phys.Lett. 97,270-274(1983), 117,396(1985).
M.S.Giambiagi, M.Giambiagi, F.E.Jorge Z.Naturforsch. 39a, 1259-73(1984)
I.Mayer, Theoret.Chim.Acta 67, 315-322(1985).
I.Mayer, Int.J.Quantum Chem. 29, 73-84(1986).
I.Mayer, Int.J.Quantum Chem. 29, 477-483(1986).
The same formula (apart from a factor of two) may also be seen in equation 31 of the second of these papers (the bond order formula in the 1st of these is not the same formula):
T.Okada, T.Fueno Bull.Chem.Soc.Japan 48, 2025-2032(1975)
T.Okada, T.Fueno Bull.Chem.Soc.Japan 49, 1524-1530(1976)
a review about bond orders:
I. Mayer, J.Comput.Chem. 28, 204-221(2007).

Geometry optimization and saddle point location -
J.Baker J.Comput.Chem. 7, 385-395(1986).
T.Helgaker Chem.Phys.Lett. 182, 503-510(1991).
P.Culot, G.Dive, V.H.Nguyen, J.M.Ghuysen

Theoret.Chim.Acta 82, 189-205(1992).

Vibrational Analysis in Cartesian coordinates -
W.D.Gwinn J.Chem.Phys. 55,477-481(1971)

Normal coordinate decomposition analysis -
J.A.Boatz and M.S.Gordon,
J.Phys.Chem. 93, 1819-1826(1989).

Raman intensity -
A.Komornicki, J.W.McIver J.Chem.Phys. 70, 2014-2016(1979)
G.B.Bacskay, S.Saebo, P.R.Taylor
Chem.Phys. 90, 215-224(1984)

static polarizabilities:
H.A.Kurtz, J.J.P.Stewart, K.M.Dieter
J.Comput.Chem. 11, 82-87 (1990)

dynamic polarizabilites:
P.Korambath, H.A.Kurtz, in "Nonlinear Optical Materials",
ACS Symposium Series 628, S.P.Karna and A.T.Yeates, Eds.
pp 133-144, Washington DC, 1996.

Dynamic Reaction Coordinate (DRC) -
J.J.P.Stewart, L.P.Davis, L.W.Burggraf,
J.Comput.Chem. 8, 1117-1123 (1987)
S.A.Maluendes, M.Dupuis, J.Chem.Phys. 93, 5902-5911(1990)
T.Taketsugu, M.S.Gordon, J.Phys.Chem. 99, 8462-8471(1995)
T.Taketsugu, M.S.Gordon, J.Phys.Chem. 99, 14597-604(1995)
T.Taketsugu, M.S.Gordon, J.Chem.Phys. 103, 10042-9(1995)
M.S.Gordon, G.Chaban, T.Taketsugu
J.Phys.Chem. 100, 11512-11525(1996)
T.Takata, T.Taketsugu, K.Hirao, M.S.Gordon
J.Chem.Phys. 109, 4281-4289(1998)
T.Taketsugu, T.Yanai, K.Hirao, M.S.Gordon
THEOCHEM 451, 163-177(1998)

parallelization in GAMESS (US) -
for SCF, the main GAMESS (US) paper quoted above.
T.L.Windus, M.W.Schmidt, M.S.Gordon,
Chem.Phys.Lett., 216, 375-379(1993)
T.L.Windus, M.W.Schmidt, M.S.Gordon,
Theoret.Chim.Acta 89, 77-88 (1994)
T.L.Windus, M.W.Schmidt, M.S.Gordon, in "Parallel Computing
in Computational Chemistry", ACS Symposium Series 592,
Ed. by T.G.Mattson, ACS Washington, 1995, pp 16-28.
K.K.Baldridge, M.S.Gordon, J.H.Jensen, N.Matsunaga,
M.W.Schmidt, T.L.Windus, J.A.Boatz, T.R.Cundari
ibid, pp 29-46.
G.D.Fletcher, M.W.Schmidt, M.S.Gordon
Adv.Chem.Phys. 110, 267-294 (1999)

MacMolPlt -
B.M.Bode, M.S.Gordon J.Mol.Graphics Mod. 16, 133-138(1998)

History of GAMESS (US)

GAMESS (US) is a synthesis, with many major modifications, of several programs. A large part of the program is from

HONDO 5. For sp basis functions, GAUSSIAN76 integrals have been adapted to the HONDO symmetry procedure, while Rys polynomials are used for any higher angular momentum.

Extension of the 1e- and 2e- integral routines to handle spdfg basis sets was done by Theresa Windus at North Dakota State University.

The current spdfg gradient package consists of HONDO8 code for higher angular momentum, and the Gaussian80 code for sp bases. The code was adapted into GAMESS (US) by Brett Bode at Iowa State University.

The ECP code goes back to Louis Kahn, with gradient modifications originally made by K.Kitaura, S.Obara, and K.Morokuma at IMS in Japan. The code was adapted to HONDO by Stevens, Basch, and Krauss, from whence Kiet Nguyen adapted it to GAMESS (US) at NDSU. Modifications for f functions were made by Drora Cohen and Brett Bode. This code was completely rewritten to use spdfg basis sets, to exploit shell structure during integral evaluation, and to add the capability of analytic second derivatives by Brett Bode at ISU in 1997-1998.

Changes in the manner of entering the basis set, and the atomic coordinates (including Z-matrix forms) are due to Jan Jensen at North Dakota State University.

The direct SCF implementation was done at NDSU, guided by a pilot code for the RHF case by Frank Jensen.

The Direct Inversion in the Iterative Subspace (DIIS) convergence procedure was implemented by Brenda Lam (then at the University of Houston), for RHF and UHF functions.

The UHF code was taught to do high spin ROHF by John Montgomery at United Technologies, who extended DIIS use to ROHF and the one pair GVB case. Additional GVB-DIIS cases were programmed by Galina Chaban at ISU.

The GVB part is a heavily modified version of GVBONE.

The CI module is based on Brooks and Schaefer's unitary group program which was modified to run within GAMESS (US), using a Davidson eigenvector method written by Steve Elbert.

Programming of the analytic CI gradient was done by Simon Webb at Iowa State University.

The FULLNR and FOCAS MCSCF programs were contributed by Michel Dupuis of IBM from the HONDO program.

The approximate 2nd order SCF was implemented by Galina Chaban at Iowa State University. SOSCF is provided for RHF, ROHF, GVB, and MCSCF cases.

The sequential MP2 code was adapted from HONDO by Nikita Matsunaga at Iowa State, who also added the RMP2

open shell option in 1992. The MP2 gradient code is also from HONDO, and was adapted to GAMESS (US) in 1995 by Simon Webb and Nikita Matsunaga. In 1996, Simon Webb added the frozen core gradient option at ISU. Haruyuki Nakano from the University of Tokyo interfaced his multireference MCQDPT code to GAMESS (US) during a 1996 visit to ISU.

Incorporation of enough MOPAC version 6 routines to run PM3, AM1, and MNDO calculations from within GAMESS (US) was done by Jan Jensen at North Dakota State University.

The numerical force constant computation and normal mode analysis was adapted from Komornicki's GRADSCF program, with decomposition of normal modes in internal coordinates written at NDSU by Jerry Boatz.

The code for the analytic computation of RHF Hessians was contributed by Michel Dupuis of IBM from HONDO 7, with open shell CPHF code written at NDSU. The TCSCF CPHF code is the result of a collaboration between NDSU and John Montgomery at United Technologies. IR intensities and analytic polarizabilities during hessian runs were programmed by Simon Webb at ISU.

Most geometry search procedures in GAMESS (US) (NR, RFO, QA, and CONOPT) were developed by Frank Jensen of Odense University. These methods are adapted to use GAMESS (US) symmetry, and Cartesian or internal coordinates.

The non-gradient optimization so aptly described as "trudge" was adapted from HONDO 7 by Mariusz Klobukowski at U.Alberta, who added the option for CI optimizations.

The intrinsic reaction coordinate pathfinder was written at North Dakota State University, and modified later for new integration methods by Kim Baldridge. The Gonzales-Schelegel IRC stepper was incorporated by Shujun Su at Iowa State, based on a pilot code from Frank Jensen.

The code for the Dynamic Reaction Coordinate was developed by Tetsuya Taketsugu at Ochanomizu U. and U. of Tokyo, and added to GAMESS (US) by him at ISU in 1994.

The two algorithms for tracing gradient extremals were programmed by Frank Jensen at Odense University.

The surface scanning option was implemented by Richard Muller at the University of Southern California.

The radiative transition moment and Zeff spin-orbit coupling modules were written by Shiro Koseki at both North Dakota State University and at Mie University.

The full Breit-Pauli spin-orbit coupling integral package was written by Tom Furlani. This code was incorporated into GAMESS (US) by Dmitri Fedorov at Iowa State University in 1997, who generalized the active space from two electrons in two orbitals, with assistance

from a vist to ISU by Tom Furlani and Shiro Koseki.

Most polarizability calculations in GAMESS (US) were implemented by Henry Kurtz of the University of Memphis. This includes a general numerical differentiation based on application of finite electric fields, and a fully analytic calculation of static and frequency dependent NLO properties for closed shell systems. The latter code was based on a MOPAC implementation by Prakashan Korambath at U. Memphis.

Edmiston-Ruedenberg energy localization is done with a version of the ALIS program "LOCL", modified to run inside GAMESS (US) at NDSU. Foster-Boys localization is based on a highly modified version of QCPE program 354 by D.Boerth, J.A.Hasmall, and A.Streitweiser. John Montgomery implemented the population localization. The LCD SCF decomposition and the MP2 decomposition were written by Jan Jensen at Iowa State in 1994.

Point Determined Charges were implemented by Mark Spackman at the University of New England, Australia.

The Morokuma decomposition was implemented by Wei Chen at Iowa State University.

Development of the EFP method began in the group of Walt Stevens at NIST's Center for Advanced Research in Biotechnology (CARB) in 1988. Walt is the originator of this method, and has provided both guidance and some financial support to ISU for its continued development. Mark Gordon's group's participation began in 1989-90 as discussions during a year Mark spent in the DC area, and became more serious in 1991 with a visit by Jan Jensen to CARB. At this time the method worked for the energy, and gradient with respect to the ab initio nuclei, for one fragment only. Jan has assisted with most aspects of the multi-fragment development since. Paul Day at NDSU and ISU derived and implemented the gradient with respect to fragments, and programmed EFP geometry optimization. Wei Chen at ISU debugged many parts of the EFP energy and gradient, developed the code for following IRCs, improved geometry searches, and fitted much more accurate repulsive potentials. Simon Webb at ISU programmed the current self-consistency process for the induced dipoles. The EFP method was sufficiently developed, tested, and described to be released in Sept 1996.

The SCRF solvent model was implemented by Dave Garmer at CARB, and was adapted to GAMESS (US) by Jan Jensen and Simon Webb at Iowa State University.

The PCM code originates in the group of Jacopo Tomasi at the University of Pisa. Benedetta Mennucci was instrumental in interfacing the PCM code to GAMESS (US), in 1997, and answering many technical questions about the code, the methodology, and the documentation.

The Ames Laboratory determinant full CI code was

written by Joe Ivanic and Klaus Ruedenberg. As befits code written by an Australian living in Iowa, it was interfaced to GAMESS (US) during an extremely cordial visit by a reluctant Iowan to Australia National University in January 1998.

Delocalized internal coordinates were implemented by Jim Shoemaker at the Air Force Institute of Technology in 1997, and put online in GAMESS (US) by Cheol Choi at ISU after further improvements in 1998.

Basis Set References

An excellent review of the relationship between the atomic basis used, and the accuracy with which various molecular properties will be computed is: E.R.Davidson, D.Feller Chem.Rev. 86, 681-696(1986).

STO-NG	H-Ne	Ref. 1 and 2
	Na-Ar,	Ref. 2 and 3 **
	K,Ca,Ga-Kr	Ref. 4
	Rb,Sr,In-Xe	Ref. 5
	Sc-Zn,Y-Cd	Ref. 6

1) W.J.Hehre, R.F.Stewart, J.A.Pople
   J.Chem.Phys. 51, 2657-2664(1969).
2) W.J.Hehre, R.Ditchfield, R.F.Stewart, J.A.Pople
   J.Chem.Phys. 52, 2769-2773(1970).
3) M.S.Gordon, M.D.Bjorke, F.J.Marsh, M.S.Korth
   J.Am.Chem.Soc. 100, 2670-2678(1978).
   ** the valence scale factors for Na-Cl are taken
      from this paper, rather than the "official"
      Pople values in Ref. 2.
4) W.J.Pietro, B.A.Levi, W.J.Hehre, R.F.Stewart,
   Inorg.Chem. 19, 2225-2229(1980).
5) W.J.Pietro, E.S.Blurock, R.F.Hout,Jr., W.J.Hehre, D.J.
   DeFrees, R.F.Stewart Inorg.Chem. 20, 3650-3654(1980).
6) W.J.Pietro, W.J.Hehre J.Comput.Chem. 4, 241-251(1983).

MINI/MIDI	H-Xe	Ref. 9

9) "Gaussian Basis Sets for Molecular Calculations"
   S.Huzinaga, J.Andzelm, M.Klobukowski, E.Radzio-Andzelm,
   Y.Sakai, H.Tatewaki Elsevier, Amsterdam, 1984.

The MINI bases are three gaussian expansions of each atomic orbital. The exponents and contraction coefficients are optimized for each element, and s and p exponents are not constrained to be equal. As a result these bases give much lower energies than does STO-3G. The valence MINI orbitals of main group elements are scaled by factors optimized by John Deisz at North Dakota State University. Transition metal MINI bases are not scaled. The MIDI bases are derived from the MINI sets by floating the outermost primitive in each valence orbitals, and renormalizing the remaining 2 gaussians. MIDI bases are not scaled by PC GAMESS. The transition metal bases are

taken from the lowest SCF terms in the s**1,d**n
configurations.

3-21G	H-Ne	Ref. 10	(also 6-21G)
	Na-Ar	Ref. 11	(also 6-21G)
K,Ca,Ga-Kr,Rb,Sr,In-Xe		Ref. 12	
	Sc-Zn	Ref. 13	
	Y-Cd	Ref. 14	

10) J.S.Binkley, J.A.Pople, W.J.Hehre
J.Am.Chem.Soc. 102, 939-947(1980).
11) M.S.Gordon, J.S.Binkley, J.A.Pople, W.J.Pietro,
W.J.Hehre J.Am.Chem.Soc. 104, 2797-2803(1982).
12) K.D.Dobbs, W.J.Hehre J.Comput.Chem. 7,359-378(1986)
13) K.D.Dobbs, W.J.Hehre J.Comput.Chem. 8,861-879(1987)
14) K.D.Dobbs, W.J.Hehre J.Comput.Chem. 8,880-893(1987)

N-31G	references for	4-31G	5-31G	6-31G
	H	15	15	15
	He	23	23	23
	Li	19,24		19
	Be	20,24		20
	B	17		19
	C-F	15	16	16
	Ne	23		23
	Na-Ga			22
	Si			21 **
	P-Cl	18		22
	Ar			22
	K-Zn			25

15) R.Ditchfield, W.J.Hehre, J.A.Pople
J.Chem.Phys. 54, 724-728(1971).
16) W.J.Hehre, R.Ditchfield, J.A.Pople
J.Chem.Phys. 56, 2257-2261(1972).
17) W.J.Hehre, J.A.Pople J.Chem.Phys. 56, 4233-4234(1972).
18) W.J.Hehre, W.A.Lathan J.Chem.Phys. 56,5255-5257(1972).
19) J.D.Dill, J.A.Pople J.Chem.Phys. 62, 2921-2923(1975).
20) J.S.Binkley, J.A.Pople J.Chem.Phys. 66, 879-880(1977).
21) M.S.Gordon Chem.Phys.Lett. 76, 163-168(1980)
** - Note that the built in 6-31G basis for Si is
not that given by Pople in reference 22.
The Gordon basis gives a better wavefunction,
for a ROHF calculation in full atomic (Kh)
symmetry,

6-31G	Energy	virial
Gordon	-288.828573	1.999978
Pople	-288.828405	2.000280

See the input examples for how to run in Kh.
22) M.M.Francl, W.J.Pietro, W.J.Hehre, J.S.Binkley,
M.S.Gordon, D.J.DeFrees, J.A.Pople
J.Chem.Phys. 77, 3654-3665(1982).
23) Unpublished, copied out of GAUSSIAN82.
24) For Li and Be, 4-31G is actually a 5-21G expansion.
25) V.A.Rassolov, J.A.Pople, M.A.Ratner, T.L.Windus
J.Chem.Phys. 109, 1223-1229(1998)

Extended basis sets

--> 6-311G

28) R.Krishnan, J.S.Binkley, R.Seeger, J.A.Pople
J.Chem.Phys. 72, 650-654(1980).

--> valence double zeta "DZV" sets:

"DH" basis - DZV for H, Li-Ne, Al-Ar
30) T.H.Dunning, Jr., P.J.Hay Chapter 1 in "Methods of
Electronic Structure Theory", H.F.Shaefer III, Ed.
Plenum Press, N.Y. 1977, pp 1-27.
Note that PC GAMESS uses inner/outer scale factors of
1.2 and 1.15 for DH's hydrogen (since at least 1983).
To get Thom's usual basis, scaled 1.2 throughout:

```
    HYDROGEN   1.0   x, y, z
       DH  0  1.2   1.2
```

DZV for K,Ca
31) J.-P.Blaudeau, M.P.McGrath, L.A.Curtiss, L.Radom
J.Chem.Phys. 107, 5016-5021(1997)
"BC" basis - DZV for Ga-Kr
32) R.C.Binning, Jr., L.A.Curtiss
J.Comput.Chem. 11, 1206-1216(1990)

--> valence triple zeta "TZV" sets:

TZV for H,Li-Ne
40) T.H. Dunning, J.Chem.Phys. 55 (1971) 716-723.
TZV for Na-Ar - also known as the "MC" basis
41) A.D.McLean, G.S.Chandler
J.Chem.Phys. 72,5639-5648(1980).
TZV for K,Ca
42) A.J.H. Wachters, J.Chem.Phys. 52 (1970) 1033-1036.
(see Table VI, Contraction 3).
TZV for Sc-Zn (taken from HONDO 7)
This is Wachters' (14s9p5d) basis (ref 42) contracted
to (10s8p3d) with the following modifications

1. the most diffuse s removed;
2. additional s spanning 3s-4s region;
3. two additional p functions to describe the 4p;
4. (6d) contracted to (411) from ref 43,
   except for Zn where Wachter's (5d)/[41]
   and Hay's diffuse d are used.

43) A.K. Rappe, T.A. Smedley, and W.A. Goddard III,
J.Phys.Chem. 85 (1981) 2607-2611

Valence only basis sets (for use with corresponding ECPs)

SBKJC -31G splits, bigger for trans. metals (available Li-Rn)
50) W.J.Stevens, H.Basch, M.Krauss
J.Chem.Phys. 81, 6026-6033 (1984)
51) W.J.Stevens, H.Basch, M.Krauss, P.Jasien
Can.J.Chem, 70, 612-630 (1992)
52) T.R.Cundari, W.J.Stevens

J.Chem.Phys. 98, 5555-5565(1993)

HW -21 splits (sp exponents not shared)
transition metals (not built in at present, although they will work if you type them in).

53) P.J.Hay, W.R.Wadt J.Chem.Phys. 82, 270-283 (1985)
main group (available Na-Xe)
54) W.R.Wadt, P.J.Hay J.Chem.Phys. 82, 284-298 (1985)
see also
55) P.J.Hay, W.R.Wadt J.Chem.Phys. 82, 299-310 (1985)

Polarization exponents

STO-NG*

60) J.B.Collins, P. von R. Schleyer, J.S.Binkley, J.A.Pople J.Chem.Phys. 64, 5142-5151(1976).

3-21G*. See also reference 12.

61) W.J.Pietro, M.M.Francl, W.J.Hehre, D.J.DeFrees, J.A. Pople, J.S.Binkley J.Am.Chem.Soc. 104,5039-5048(1982)

6-31G* and 6-31G**. See also reference 22 above.

62) P.C.Hariharan, J.A.Pople
Theoret.Chim.Acta 28, 213-222(1973)

multiple polarization, and f functions

63) M.J.Frisch, J.A.Pople, J.S.Binkley J.Chem.Phys. 80, 3265-3269 (1984).

STO-NG* means d orbitals are used on third row atoms only. The original paper (ref 60) suggested z=0.09 for Na and Mg, and z=0.39 for Al-Cl. At NDSU we prefer to use the same exponents as in 3-21G* and 6-31G*, so we know we're looking at changes in the sp basis, not the d exponent.

3-21G* means d orbitals on main group elements in the third and higher periods. Not defined for the transition metals, where there are p's already in the basis. Except for alkalis and alkali earths, the 4th and 5th row zetas are from Huzinaga, et al (ref 9). The exponents are normally the same as for 6-31G*.

6-31G* means d orbitals on second and third row atoms. We use Mark Gordon's z=0.395 for silicon, as well as his fully optimized sp basis (ref 21). This is often written 6-31G(d) today. For the first row transition metals, the * means an f function is added.

6-31G** means the same as 6-31G*, except that p functions are added on hydrogens. This is often written 6-31G(d,p) today.

6-311G** means p orbitals on H, and d orbitals elsewhere. The exponents were derived from correlated atomic

states, and so are considerably tighter than the polarizing functions used in 6-31G**, etc. This is often written 6-311G(d,p) today.

The definitions for 6-31G* for C-F are disturbing in that they treat these atoms the same. Dunning and Hay (ref 30) have recommended a better set of exponents for second row atoms and a slightly different value for H.

2p, 3p, 2d, 3p polarization sets are usually thought of as arising from applying splitting factors to the 1p and 1d values. For example, SPLIT2=2.0, 0.5 means to double and halve the single value. The default values for SPLIT2 and SPLIT3 are taken from reference 72, and were derived with correlation in mind. The SPLIT2 values often produce a higher (!) HF energy than the singly polarized run, because the exponents are split too widely. SPLIT2=0.4,1.4 will always lower the SCF energy (the values are the unpublished personal preference of MWS), and for SPLIT3 we might suggest 3.0,1.0,1/3. See ref 63 for more on this.

With all this as background, we are ready to present the table of polarization exponents built into PC GAMESS.

Built in polarization exponents, chosen by POLAR= in the $BASIS group. The values are for d functions unless otherwise indicated.

Please note that the names associated with each column are only generally descriptive. For example, the column marked "Pople" contains a value for Si with which John Pople would not agree, and the Ga-Kr values in this column are actually from the Huzinaga "green book". The exponents for K-Kr under "Dunning" are from Curtiss, et al., not Thom Dunning. And so on.

	POPLE	POPN311	DUNNING	HUZINAGA	HONDO7
H	1.1(p)	0.75(p)	1.0(p)	1.0(p)	1.0(p)
He	1.1(p)	0.75(p)	1.0(p)	1.0(p)	1.0(p)
Li	0.2	0.200		0.076(p)	
Be	0.4	0.255		0.164(p)	0.32
B	0.6	0.401	0.70	0.388	0.50
C	0.8	0.626	0.75	0.600	0.72
N	0.8	0.913	0.80	0.864	0.98
O	0.8	1.292	0.85	1.154	1.28
F	0.8	1.750	0.90	1.496	1.62
Ne	0.8	2.304	1.00	1.888	2.00
Na	0.175			0.061(p)	0.157
Mg	0.175			0.101(p)	0.234
Al	0.325			0.198	0.311
Si	0.395			0.262	0.388
P	0.55			0.340	0.465
S	0.65			0.421	0.542
Cl	0.75			0.514	0.619
Ar	0.85			0.617	0.696

K	0.2		0.260	0.039(p)	
Ca	0.2		0.229	0.059(p)	
Sc-Zn	0.8(f)	N/A	N/A	N/A	N/A
Ga	0.207		0.141		
Ge	0.246		0.202		
As	0.293		0.273		
Se	0.338		0.315		
Br	0.389		0.338		
Kr	0.443		0.318		
Rb	0.11			0.034(p)	
Sr	0.11			0.048(p)	

A blank means the value equals the "Pople" column.

Common d polarization for all sets ("green book"):

In	Sn	Sb	Te	I	Xe
0.160	0.183	0.211	0.237	0.266	0.297
Tl	Pb	Bi	Po	At	Rn
0.146	0.164	0.185	0.204	0.225	0.247

f polarization functions, from reference 63:

Li	Be	B	C	N	O	F	Ne
0.15	0.26	0.50	0.80	1.00	1.40	1.85	2.50
Na	Mg	Al	Si	P	S	Cl	Ar
0.15	0.20	0.25	0.32	0.45	0.55	0.70	--

Anion diffuse functions

3-21+G, 3-21++G, etc.
70) T.Clark, J.Chandrasekhar, G.W.Spitznagel, P. von R. Schleyer J.Comput.Chem. 4, 294-301(1983)
71) G.W.Spitznagel, Diplomarbeit, Erlangen, 1982.

Anions usually require diffuse basis functions to properly represent their spatial diffuseness. The use of diffuse sp shells on atoms in the second and third rows is denoted by a + sign, also adding diffuse s functions on hydrogen is symbolized by ++. These designations can be applied to any of the Pople bases, e.g. 3-21+G, 3-21+G*, 6-31++G**. The following exponents are for L shells, except for H. For H-F, they are taken from ref 70. For Na-Cl, they are taken directly from reference 71. These values may be found in footnote 13 of reference 63. For Ga-Br, In-I, and Tl-At these were optimized for the atomic ground state anion, using ROHF with a flexible ECP basis set, by Ted Packwood at NDSU.

H						
0.0360						
Li	Be	B	C	N	O	F
0.0074	0.0207	0.0315	0.0438	0.0639	0.0845	0.1076
Na	Mg	Al	Si	P	S	Cl
0.0076	0.0146	0.0318	0.0331	0.0348	0.0405	0.0483
		Ga	Ge	As	Se	Br
		0.0205	0.0222	0.0287	0.0318	0.0376
		In	Sn	Sb	Te	I

0.0223	0.0231	0.0259	0.0306	0.0368
Tl	Pb	Bi	Po	At
0.0170	0.0171	0.0215	0.0230	0.0294

Additional information about diffuse functions and also Rydberg type exponents can be found in reference 30.

The following atomic energies are from UHF calculations (RHF on 1-S states), with p orbitals not symmetry equivalenced, and using the default molecular scale factors. They should be useful in picking a basis of the desired energy accuracy, and estimating the correct molecular total energies.

Atom	state	STO-2G	STO-3G	3-21G	6-31G
H	2-S	-.454397	-.466582	-.496199	-.498233
He	1-S	-2.702157	-2.807784	-2.835680	-2.855160
Li	2-S	-7.070809	-7.315526	-7.381513	-7.431236
Be	1-S	-13.890237	-14.351880	-14.486820	-14.566764
B	2-P	-23.395284	-24.148989	-24.389762	-24.519492
C	3-P	-36.060274	-37.198393	-37.481070	-37.677837
N	4-S	-53.093007	-53.719010	-54.105390	-54.385008
O	3-P	-71.572305	-73.804150	-74.393657	-74.780310
F	2-P	-95.015084	-97.986505	-98.845009	-99.360860
Ne	1-S	-122.360485	-126.132546	-127.803825	-128.473877
Na	2-S	-155.170019	-159.797148	-160.854065	-161.841425
Mg	1-S	-191.507082	-197.185978	-198.468103	-199.595219
Al	2-P	-233.199965	-239.026471	-240.551046	-241.854186
Si	3-P	-277.506857	-285.563052	-287.344431	-288.828598
P	4-S	-327.564244	-336.944863	-339.000079	-340.689008
S	3-P	-382.375012	-393.178951	-395.551336	-397.471414
Cl	2-P	-442.206260	-454.546015	-457.276552	-459.442939
Ar	1-S	-507.249273	-521.222881	-524.342962	-526.772151

Atom	state	DH	6-311G	MC	SCF limit *
H	2-S	-.498189	-.499810	--	-0.5
He	1-S	--	-2.859895	--	-2.861680
Li	2-S	-7.431736	-7.432026	--	-7.432727
Be	1-S	-14.570907	-14.571874	--	-14.573023
B	2-P	-24.526601	-24.527020	--	-24.529061
C	3-P	-37.685571	-37.686024	--	-37.688619
N	4-S	-54.397260	-54.397980	--	-54.400935
O	3-P	-74.802707	-74.802496	--	-74.809400
F	2-P	-99.395013	-99.394158	--	-99.409353
Ne	1-S	-128.522354	-128.522553	--	-128.547104
Na	2-S	--	--	-161.845587	-161.858917
Mg	1-S	--	--	-199.606558	-199.614636
Al	2-P	-241.855079	--	-241.870014	-241.876699
Si	3-P	-288.829617	--	-288.847782	-288.854380
P	4-S	-340.689043	--	-340.711346	-340.718798
S	3-P	-397.468667	--	-397.498023	-397.504910
Cl	2-P	-459.435938	--	-459.473412	-459.482088
Ar	1-S	--	--	-526.806626	-526.817528

* M.W.Schmidt and K.Ruedenberg, J.Chem.Phys. 71,

3951-3962(1979). These are ROHF energies in Kh symmetry.

How to do RHF, ROHF, UHF, and GVB calculations

* * * General considerations * * *

These four SCF wavefunctions are all based on Fock operator techniques, even though some GVB runs use more than one determinant. Thus all of these have an intrinsic N**4 time dependence, because they are all driven by integrals in the AO basis. This similarity makes it convenient to discuss them all together. In this section we will use the term HF to refer generically to any of these four wavefunctions, including the multi-determinate GVB-PP functions. $SCF is the main input group for all these HF wavefunctions.

As will be discussed below, in PC GAMESS the term ROHF refers to high spin open shell SCF only, but other open shell coupling cases are possible using the GVB code.

Analytic gradients are implemented for every possible HF type calculation possible in PC GAMESS, and therefore numerical hessians are available for each.

Analytic hessian calculation is implemented for RHF, ROHF, and any GVB case with NPAIR=0 or NPAIR=1. Analytic hessians are more accurate, and much more quickly computed than numerical hessians, but require additional disk storage to perform an integral transformation, and also more physical memory.

The second order Moller-Plesset energy correction (MP2) is implemented for RHF, UHF, ROHF, and MCSCF wave functions. Analytic gradients may be obtained for MP2 with a RHF reference function. MP2 properties are formed only by RHF gradient runs, or if you select MP2PRP in the $MP2 group. All other cases give properties for the SCF function.

Direct SCF is implemented for every possible HF type calculation. The direct SCF method may not be used with DEM convergence. Direct SCF may be used during energy, gradient, numerical or analytic hessian, CI or MP2 energy correction, or localized orbitals computations.

* * * direct SCF * * *

Normally, HF calculations proceed by evaluating a large number of two electron repulsion integrals, and storing these on a disk. This integral file is read back in during each HF iteration to form the appropriate Fock operators. In a direct HF, the integrals are not stored on disk, but are instead reevaluated during each HF iteration. The default for DIRSCF in $SCF is .FALSE.

You can estimate the disk storage requirements for conventional HF using a P or PK file by the following formulae:

```
nint = 1/sigma * 1/8 * N**4
Mbytes = nint * x / 1024**2
```

Here N is the total number of basis functions in your run, which you can learn from an EXETYP=CHECK run. The 1/8 accounts for permutational symmetry within the integrals. Sigma accounts for the point group symmetry, and is difficult to estimate accurately. Sigma cannot be smaller than 1, in no symmetry (C1) calculations. For benzene, sigma would be almost six, since you generate 6 C's and 6 H's by entering only 1 of each in $DATA. For water sigma is not much larger than one, since most of the basis set is on the unique oxygen, and the C2v symmetry applies only to the H atoms. The factor x is 12 bytes per integral for RHF, and 20 bytes per integral for ROHF, UHF, and GVB. Finally, since integrals very close to zero need not be stored on disk, the actual power dependence is not as bad as N**4, and in fact in the limit of very large molecules can be as low as N**2. Thus plugging in sigma=1 should give you an upper bound to the actual disk space needed. If the estimate exceeds your available disk storage, your only recourse is direct HF.

What are the economics of direct HF? Naively, if we assume the run takes 10 iterations to converge, we must spend 10 times more CPU time doing the integrals on each iteration. However, we do not have to waste any CPU time reading blocks of integrals from disk, or in unpacking their indices. We also do not have to waste any wall clock time waiting for a relatively slow mechanical device such as a disk to give us our data.

There are some less obvious savings too, as first noted by Almlof. First, since the density matrix is known while we are computing integrals, we can use the Schwarz inequality to avoid doing some of the integrals. In a conventional SCF this inequality is used to avoid doing small integrals. In a direct SCF it can be used to avoid doing integrals whose contribution to the Fock matrix is small (density times integral=small). Secondly, we can form the Fock matrix by calculating only its change since the previous iteration. The contributions to the change in the Fock matrix are equal to the change in the density times the integrals. Since the change in the density goes to zero as the run converges, we can use the Schwarz screening to avoid more and more integrals as the calculation progresses. The input option FDIFF in $SCF selects formation of the Fock operator by computing only its change from iteration to iteration. The FDIFF option is not implemented for GVB since there are too many density matrices from the previous iteration to store, but is the default for direct RHF, ROHF, and UHF.

So, in our hypothetical 10 iteration case, we do not spend as much as 10 times more time in integral

evaluation. Additionally, the run as a whole will not slow down by whatever factor the integral time is increased. A direct run spends no additional time summing integrals into the Fock operators, and no additional time in the Fock diagonalizations. So, generally speaking, a RHF run with 10-15 iterations will slow down by a factor of 2-4 times when run in direct mode. The energy gradient time is unchanged by direct HF, and this is a large time compared to HF energy, so geometry optimizations will be slowed down even less. This is really the converse of Amdahl's law: if you slow down only one portion of a program by a large amount, the entire program slows down by a much smaller factor.

* * * SCF convergence accelerators * * *

Generally speaking, the simpler the HF function, the better its convergence. In our experience, the majority of RHF, ROHF, and UHF runs will converge readily from GUESS=HUCKEL. GVB runs typically require GUESS=MOREAD, although the Huckel guess usually works for NPAIR=0. RHF convergence is the best, closely followed by ROHF. In the current implementation in PC GAMESS, ROHF is always better convergent than the closely related unrestricted high spin UHF. For example, the radical cation of diphosphine converges in 12 iterations for ROHF but requires 15 iters for UHF, both runs starting from the neutral's closed shell orbitals. GVB calculations require much more care, and cases with NPAIR greater than one are particularly difficult.

Unfortunately, not all HF runs converge readily. The best way to improve your convergence is to provide better starting orbitals! In many cases, this means to MOREAD orbitals from some simpler HF case. For example, if you want to do a doublet ROHF, and the HUCKEL guess does not seem to converge, do this: Do an RHF on the +1 cation. RHF is typically more stable than ROHF, UHF, or GVB, and cations are usually readily convergent. Then MOREAD the cation's orbitals into the neutral calculation which you wanted to do at first.

GUESS=HUCKEL does not always guess the correct electronic configuration. It may be useful to use PRTMO in $GUESS during a CHECK run to examine the starting orbitals, and then reorder them with NORDER if that seems appropriate.

Of course, by default PC GAMESS uses the convergence procedures which are usually most effective. Still, there are cases which are difficult, so the $SCF group permits you to select several alternative methods for improving convergence. Briefly, these are

EXTRAP. This extrapolates the three previous Fock matrices, in an attempt to jump ahead a bit faster. This is the most powerful of the old-fashioned accelerators, and normally should be used at the beginning of any SCF run. When an extrapolation occurs, the counter at the left of the SCF printout is set to zero.

DAMP. This damps the oscillations between several successive Fock matrices. It may help when the energy is seen to oscillate wildly. Thinking about which orbitals should be occupied initially may be an even better way to avoid oscillatory behaviour.

SHIFT. This shifts the diagonal elements of the virtual part of the Fock matrix up, in an attempt to uncouple the unoccupied orbitals from the occupied ones. At convergence, this has no effect on the orbitals, just their orbital energies, but will produce different (and hopefully better) orbitals during the iterations.

RSTRCT. This limits mixing of the occupied orbitals with the empty ones, especially the flipping of the HOMO and LUMO to produce undesired electronic configurations or states. This should be used with caution, as it makes it very easy to converge on incorrect electronic configurations, especially if DIIS is also used. If you use this, be sure to check your final orbital energies to see if they are sensible. A lower energy for an unoccupied orbital than for one of the occupied ones is a sure sign of problems.

DIIS. Direct Inversion in the Iterative Subspace is a modern method, due to Pulay, using stored error and Fock matrices from a large number of previous iterations to interpolate an improved Fock matrix. This method was developed to improve the convergence at the final stages of the SCF process, but turns out to be quite powerful at forcing convergence in the initial stages of SCF as well. By giving ETHRSH as 10.0 in $SCF, you can practically guarantee that DIIS will be in effect from the first iteration. The default is set up to do a few iterations with conventional methods (extrapolation) before engaging DIIS. This is because DIIS can sometimes converge to solutions of the SCF equations that do not have the lowest possible energy. For example, the 3-A-2 small angle state of SiLi2 (see M.S.Gordon and M.W.Schmidt, Chem.Phys.Lett., 132, 294-8(1986)) will readily converge with DIIS to a solution with a reasonable S**2, and an energy about 25 milliHartree above the correct answer. A SURE SIGN OF TROUBLE WITH DIIS IS WHEN THE ENERGY RISES TO ITS FINAL VALUE. However, if you obtain orbitals at one point on a PES without DIIS, the subsequent use of DIIS with MOREAD will probably not introduce any problems. Because DIIS is quite powerful, EXTRAP, DAMP, and SHIFT are all turned off once DIIS begins to work. DEM and RSTRCT will still be in use, however.

SOSCF. Approximate second-order (quasi-Newton) SCF orbital optimization. SOSCF will converge about as well as DIIS at the initial geometry, and slightly better at subsequent geometries. There's a bit less work solving the SCF equations, too. The method kicks in after the orbital gradient falls below SOGTOL. Some systems, particularly transition metals with ECP basis sets, may have Huckel orbitals for which the gradient is much larger than SOGTOL. In this case it is probably better to use DIIS instead,

with a large ETHRSH, rather than increasing SOGTOL, since you may well be outside the quadratic convergence region. SOSCF does not exhibit true second order convergence since it uses an approximation to the inverse hessian. SOSCF

will work for MOPAC runs, but is slower in this case. SOSCF will work for UHF, but the convergence is slower than DIIS. SOSCF will work for non-Abelian ROHF cases, but may encounter problems if the open shell is degenerate.

DEM. Direct energy minimization should be your last recourse. It explores the "line" between the current orbitals and those generated by a conventional change in the orbitals, looking for the minimum energy on that line. DEM should always lower the energy on every iteration, but is very time consuming, since each of the points considered on the line search requires evaluation of a Fock operator. DEM will be skipped once the density change falls below DEMCUT, as the other methods should then be able to affect final convergence. While DEM is working, RSTRCT is held to be true, regardless of the input choice for RSTRCT. Because of this, it behooves you to be sure that the initial guess is occupying the desired orbitals. DEM is available only for RHF. The implementation in PC GAMESS resembles that of R.Seeger and J.A.Pople, J.Chem.Phys. 65, 265-271(1976). Simultaneous use of DEM and DIIS resembles the ADEM-DIOS method of H.Sellers, Chem.Phys.Lett. 180, 461-465(1991). DEM does not work with direct SCF.

* * * High spin open shell SCF (ROHF) * * *

Open shell SCF calculations are performed in PC GAMESS by both the ROHF code and the GVB code. Note that when the GVB code is executed with no pairs, the run is NOT a true GVB run, and should be referred to in publications and discussion as a ROHF calculation.

The ROHF module in PC GAMESS can handle any number of open shell electrons, provided these have a high spin coupling. Some commonly occurring cases are:

one open shell, doublet:

```
 $CONTRL SCFTYP=ROHF MULT=2 $END
```

two open shells, triplet:

```
 $CONTRL SCFTYP=ROHF MULT=3 $END
```

m open shells, high spin:

```
 $CONTRL SCFTYP=ROHF MULT=m+1 $END
```

John Montgomery (then at United Technologies) is responsible for the current ROHF implementation in PC GAMESS. The following discussion is due to him:

The Fock matrix in the MO basis has the form

	closed	open	virtual
closed	F2	Fb	(Fa+Fb)/2
open	Fb	F1	Fa
virtual	(Fa+Fb)/2	Fa	F0

where Fa and Fb are the usual alpha and beta Fock matrices any UHF program produces. The Fock operators for the doubly, singly, and zero occupied blocks can be written as

```
F2 = Acc*Fa + Bcc*Fb
F1 = Aoo*Fa + Boo*Fb
F0 = Avv*Fa + Bvv*Fb
```

Some choices found in the literature for these canonicalization coefficients are

	Acc	Bcc	Aoo	Boo	Avv	Bvv
Guest and Saunders	1/2	1/2	1/2	1/2	1/2	1/2
Roothaan single matrix	-1/2	3/2	1/2	1/2	3/2	-1/2
Davidson	1/2	1/2	1	0	1	0
Binkley, Pople, Dobosh	1/2	1/2	1	0	0	1
McWeeny and Diercksen	1/3	2/3	1/3	1/3	2/3	1/3
Faegri and Manne	1/2	1/2	1	0	1/2	1/2

The choice of the diagonal blocks is arbitrary, as ROHF is converged when the off diagonal blocks go to zero. The exact choice for these blocks can however have an effect on the convergence rate. This choice also affects the MO coefficients, and orbital energies, as the different choices produce different canonical orbitals within the three subspaces. All methods, however, will give identical total wavefunctions, and hence identical properties such as gradients and hessians.

The default coupling case in PC GAMESS is the Roothaan single matrix set. Note that pre-1988 versions of GAMESS (US) produced "Davidson" orbitals. If you would like to fool around with any of these other canonicalizations, the Acc, Aoo, Avv and Bcc, Boo, Bvv parameters can be input as the first three elements of ALPHA and BETA in $SCF.

* * * Other open shell SCF cases (GVB) * * *

Genuine GVB-PP runs will be discussed later in this section. First, we will consider how to do open shell SCF with the GVB part of the program.

It is possible to do other open shell cases with the GVB code, which can handle the following cases:

```
one open shell, doublet:
     $CONTRL SCFTYP=GVB MULT=2 $END
     $SCF    NCO=xx NSETO=1 NO(1)=1 $END
two open shells, triplet:
     $CONTRL SCFTYP=GVB MULT=3 $END
```

```
     $SCF    NCO=xx NSETO=2 NO(1)=1,1 $END
```
two open shells, singlet:
```
     $CONTRL SCFTYP=GVB MULT=1 $END
     $SCF    NCO=xx NSETO=2 NO(1)=1,1 $END
```

Note that the first two cases duplicate runs which the ROHF module can do better. Note that all of these cases are really ROHF, since the default for NPAIR in $SCF is 0.

Many open shell states with degenerate open shells (for example, in diatomic molecules) can be treated as well.

If you would like to do any cases other than those shown above, you must derive the coupling coefficients ALPHA and BETA, and input them with the occupancies F in the $SCF group.

Mariusz Klobukowski of the University of Alberta has shown how to obtain coupling coefficients for the GVB open shell program for many such open shell states. These can be derived from the values in Appendix A of the book "A General SCF Theory" by Ramon Carbo and Josep M. Riera, Springer-Verlag (1978). The basic rule is

```
       (1)      F(i) = 1/2 * omega(i)
       (2)  ALPHA(i) =       alpha(i)
       (3)   BETA(i) =     - beta(i),
```

where omega, alpha, and beta are the names used by Ramon in his Tables.

The variable NSETO should give the number of open shells, and NO should give the degeneracy of each open shell. Thus the 5-S state of carbon would have NSETO=2, and NO(1)=1,3.

Some specific examples, for the lowest term in each of the atomic P**N configurations are

```
!   p**1   2-P state
 $CONTRL SCFTYP=GVB  MULT=2   $END
 $SCF    NCO=xx   NSETO=1  NO=3   COUPLE=.TRUE.
      F(1)=  1.0  0.16666666666667
  ALPHA(1)=  2.0  0.33333333333333  0.00000000000000
   BETA(1)= -1.0 -0.16666666666667 -0.00000000000000  $END

!   p**2   3-P state
 $CONTRL SCFTYP=GVB  MULT=3   $END
 $SCF    NCO=xx   NSETO=1  NO=3   COUPLE=.TRUE.
      F(1)=  1.0  0.333333333333333
  ALPHA(1)=  2.0  0.66666666666667  0.16666666666667
   BETA(1)= -1.0 -0.33333333333333 -0.16666666666667  $END

!   p**3   4-S state
 $CONTRL SCFTYP=ROHF  MULT=4   $END

!   p**4   3-P state
 $CONTRL SCFTYP=GVB  MULT=3   $END
```

```
 $SCF    NCO=xx   NSETO=1  NO=3   COUPLE=.TRUE.
      F(1)=  1.0  0.66666666666667
   ALPHA(1)=  2.0  1.33333333333333  0.83333333333333
    BETA(1)= -1.0 -0.66666666666667 -0.50000000000000  $END

!   p**5   2-P state
 $CONTRL SCFTYP=GVB  MULT=2   $END
 $SCF    NCO=xx   NSETO=1  NO=3   COUPLE=.TRUE.
      F(1)=  1.0  0.83333333333333
   ALPHA(1)=  2.0  1.66666666666667  1.33333333333333
    BETA(1)= -1.0 -0.83333333333333 -0.66666666666667  $END
```

Be sure to give all the digits, as these are part of a double precision energy formula.

Coupling constants for d**N configurations are

```
!     d**1   2-D state
 $CONTRL SCFTYP=GVB MULT=2 $END
 $SCF    NCO=xx NSETO=1 NO=5 COUPLE=.TRUE.  F(1)=1.0,0.1
         ALPHA(1)= 2.0, 0.20, 0.00
          BETA(1)=-1.0,-0.10, 0.00  $END

!     d**2   average of 3-F and 3-P states
 $CONTRL SCFTYP=GVB MULT=3 $END
 $SCF    NCO=xx NSETO=1 NO=5 COUPLE=.TRUE.  F(1)=1.0,0.2
         ALPHA(1)= 2.0, 0.40, 0.05
          BETA(1)=-1.0,-0.20,-0.05  $END



!     d**3   average of 4-F and 4-P states
 $CONTRL SCFTYP=GVB MULT=4 $END
 $SCF    NCO=xx NSETO=1 NO=5 COUPLE=.TRUE.  F(1)=1.0,0.3
         ALPHA(1)= 2.0, 0.60, 0.15
          BETA(1)=-1.0,-0.30,-0.15  $END

!     d**4   5-D state
 $CONTRL SCFTYP=GVB MULT=5 $END
 $SCF    NCO=xx NSETO=1 NO=5 COUPLE=.TRUE.  F(1)=1.0,0.4
         ALPHA(1)= 2.0, 0.80, 0.30
          BETA(1)=-1.0,-0.40,-0.30 $END

!     d**5   6-S state
 $CONTRL SCFTYP=ROHF MULT=6 $END

!     d**6   5-D state
 $CONTRL SCFTYP=GVB MULT=5 $END
 $SCF    NCO=xx NSETO=1 NO=5 COUPLE=.TRUE.  F(1)=1.0,0.6
         ALPHA(1)= 2.0, 1.20, 0.70
          BETA(1)=-1.0,-0.60,-0.50 $END

!     d**7   average of 4-F and 4-P states
 $CONTRL SCFTYP=GVB MULT=4 $END
 $SCF    NCO=xx NSETO=1 NO=5 COUPLE=.TRUE.  F(1)=1.0,0.7
         ALPHA(1)= 2.0, 1.40, 0.95
          BETA(1)=-1.0,-0.70,-0.55  $END

!     d**8   average of 3-F and 3-P states
```

```
 $CONTRL SCFTYP=GVB MULT=3 $END
 $SCF    NCO=xx NSETO=1 NO=5 COUPLE=.TRUE.  F(1)=1.0,0.8
         ALPHA(1)= 2.0, 1.60, 1.25
          beta(1)=-1.0,-0.80,-0.65  $end

!     d**9   2-D state
 $CONTRL SCFTYP=GVB MULT=2 $END
 $SCF    NCO=xx NSETO=1 NO=5 COUPLE=.TRUE.  F(1)=1.0,0.9
         ALPHA(1)= 2.0, 1.80, 1.60
          BETA(1)=-1.0,-0.90,-0.80 $END
```

The source for these values is R.Poirier, R.Kari, and I.G.Csizmadia's book "Handbook of Gaussian Basis Sets", Elsevier, Amsterdam, 1985.

Note that PC GAMESS can do a proper calculation on the ground terms for the d**2, d**3, d**7, and d**8 configurations only by means of state averaged MCSCF. For d**8, use

```
 $CONTRL SCFTYP=MCSCF MULT=3 $END
 $DRT    GROUP=C1 FORS=.TRUE. NMCC=xx NDOC=3 NALP=2 $END
 $GUGDIA NSTATE=10 $END
 $GUGDM2 WSTATE(1)=1,1,1,1,1,1,1,0,0,0 $END
```

Open shell cases such as s**1,d**n are probably most easily tackled with the state-averaged MCSCF program.

* * * True GVB perfect pairing runs * * *

True GVB runs are obtained by choosing NPAIR nonzero. If you wish to have some open shell electrons in addition to the geminal pairs, you may add the pairs to the end of any of the GVB coupling cases shown above. The GVB module assumes that you have reordered your MOs into the order: NCO double occupied orbitals, NSETO sets of open shell orbitals, and NPAIR sets of geminals (with NORDER=1 in the $GUESS group).

Each geminal consists of two orbitals and contains two singlet coupled electrons (perfect pairing). The first MO of a geminal is probably heavily occupied (such as a bonding MO u), and the second is probably weakly occupied (such as an antibonding, correlating orbital v). If you have more than one pair, you must be careful that the initial MOs are ordered u1, v1, u2, v2..., which is -NOT- the same order that RHF starting orbitals will be found in. Use NORDER=1 to get the correct order.

These pair wavefunctions are actually a limited form of MCSCF. GVB runs are much faster than MCSCF runs, because the natural orbital u,v form of the wavefunction permits a Fock operator based optimization. However, convergence of the GVB run is by no means assured. The same care in selecting the correlating orbitals that you would apply to an MCSCF run must also be used for GVB runs. In particular, look at the orbital expansions when choosing the starting orbitals, and check them again after the run converges.

GVB runs will be carried out entirely in orthonormal natural u,v form, with strong orthogonality enforced on the geminals. Orthogonal orbitals will pervade your thinking in both initial orbital selection, and the entire orbital optimization phase (the CICOEF values give the weights of the u,v orbitals in each geminal). However, once the calculation is converged, the program will generate and print the nonorthogonal, generalized valence bond orbitals. These GVB orbitals are an entirely equivalent way of presenting the wavefunction, but are generated only after the fact.

Convergence of true GVB runs is by no means as certain as convergence of RHF, UHF, ROHF, or GVB with NPAIR=0. You can assist convergence by doing a preliminary RHF or ROHF calculation, and use these orbitals for GUESS=MOREAD. Few, if any, GVB runs with NPAIR non-zero will converge without using GUESS=MOREAD. Generation of MVOs during the prelimnary SCF can also be advantageous. In fact, all the advice outlined for MCSCF computations below is germane, for GVB-PP is a type of MCSCF computation.

The total number of electrons in the GVB wavefunction is given by the following formula:

$$NE = 2*NCO + \sum_{i} 2*F(i)*NO(i) + 2*NPAIR$$

The charge is obtained by subtracting the total number of protons given in $DATA. The multiplicity is implicit in the choice of alpha and beta constants. Note that ICHARG and MULT must be given correctly in $CONTRL anyway, as the number of electrons from this formula is double checked against the ICHARG value.

* * * the special case of TCSCF * * *

The wavefunction with NSETO=0 and NPAIR=1 is called GVB-PP(1) by Goddard, two configuration SCF (TCSCF) by Schaefer or Davidson, and CAS-SCF with two electrons in two orbitals by others. Note that this is just semantics, as these are all identical. This is a very important type of wavefunction, as TCSCF is the minimum acceptable treatment for singlet biradicals. The TCSCF wavefunction can be obtained with SCFTYP=MCSCF, but it is usually much faster to use the Fock based SCFTYP=GVB. Because of its importance, the TCSCF function (if desired, with possible open shells) permits analytic hessian computation.

* * * A caution about symmetry * * *

Caution! Some exotic calculations with the GVB program do not permit the use of symmetry. The symmetry algorithm in PC GAMESS was "derived assuming that the electronic charge density transforms according to the completely symmetric representation of the point group", Dupuis/King, JCP, 68, 3998(1978). This may not be true for certain open shell cases, and in fact during GVB runs, it may not be true for closed shell singlet cases!

First, consider the following correct input for the singlet-delta state of NH:

```
 $CONTRL SCFTYP=GVB NOSYM=1 $END
 $SCF    NCO=3 NSETO=2 NO(1)=1,1 $END
```

for the x**1y**1 state, or for the x**2-y**2 state,

```
 $CONTRL SCFTYP=GVB NOSYM=1 $END
 $SCF    NCO=3 NPAIR=1 CICOEF(1)=0.707,-0.707 $END
```

Neither gives correct results, unless you enter NOSYM=1. The electronic term symbol is degenerate, a good tip off that symmetry cannot be used. However, some degenerate states can still use symmetry, because they use coupling constants averaged over all degenerate states within a single term, as is done in EXAM15 and EXAM16. Here the "state averaged SCF" leads to a charge density which is symmetric, and these runs can exploit symmetry.

Secondly, since GVB runs exploit symmetry for each of the "shells", or type of orbitals, some calculations on totally symmetric states may not be able to use symmetry. An example is CO or N2, using a three pair GVB to treat the sigma and pi bonds. Individual configurations such as (sigma)**2,(pi-x)**2,(pi-y*)**2 do not have symmetric charge densities since neither the pi nor pi* level is completely filled. Correct answers for the sigma-plus ground states result only if you input NOSYM=1.

Problems of the type mentioned should not arise if the point group is Abelian, but will be fairly common in linear molecules. Since PC GAMESS cannot detect that the GVB electronic state is not totally symmetric (or averaged to at least have a totally symmetric density), it is left up to you to decide when to input NOSYM=1. If you have any question about the use of symmetry, try it both ways. If you get the same energy, both ways, it remains valid to use symmetry to speed up your run.

And beware! Brain dead computations, such as RHF on singlet O2, which actually is a half filled degenerate shell, violate the symmetry assumptions, and also violate nature. Use of partially filled degenerate shells always leads to very wild oscillations in the RHF energy, which is how the program tries to tell you to think first, and compute second. Configurations such as pi**2, e**1, or f2u**4 can be treated, but require GVB wavefunctions and F, ALPHA, BETA values from the sources mentioned.

How to do MCSCF and CI calculations

Multi-configuration self consistent field (MCSCF) wavefunctions are the most general SCF type, offering a description of chemical processes involving the separation of electrons (bond breaking, electronically excited states, etc), which are often not well represented using the single configuration SCF methods.

MCSCF wavefunctions, as the name implies, contain more than one configuration, each of which is multiplied by a "configuration interaction (CI) coefficient", determining it weight. In addition, the orbitals which form each of the configurations are optimized, just as in a simpler SCF, to self consistency.

Typically each chemical problem requires that an MCSCF wavefunction be designed to treat it, on a case by case basis. For example, one may be interested in describing the reactivity of a particular functional group, instead of elsewhere in the molecule. This means some attention must be paid in order to obtain correct results.

Procedures for the selection of configurations (which amounts to choosing the number of active electrons and active orbitals), for the two mathematical optimizations just mentioned, ways to interpret the resulting MCSCF wavefunction, and the treatment for dynamical correlation not included in the MCSCF wavefunction are the focus of a recent review article:

"The Construction and Interpretation
of MCSCF wavefunctions"
M.W.Schmidt and M.S.Gordon,
Ann.Rev.Phys.Chem. 49,233-266(1998)

One section of this is devoted to the problem of designing the correct active space to treat your problem. Additional reading is listed at the end of this section.

The most efficient technique implemented in PC GAMESS for finding the dynamic correlation energy is second order perturbation theory, in the variant known as MCQDPT. MCQDPT is discussed in a different section of this chapter. The use of CI, probably in the form of second order CI, will be described below, en passant, during discussion of the input defining the configurations for MCSCF. Selection of a CI following some type of SCF (except UHF) is made with CITYP in the $CONTRL group, and masterminded by the $CIINP group.

--- MCSCF implementation ---

With the exception of the QUAD converger, the MCSCF program is of the type termed "unfolded two step" by Roos. This means the orbital and CI coefficient optimizations are separated. The latter are obtained in a conventional CI diagonalization, while the former are optimized by a separate orbital improvement step.

Each MCSCF iteration consists of the following steps:
1) transformation of AO integrals to the current MO basis,
2) generation of the Hamiltonian matrix and optimization
of the CI coefficients by a Davidson diagonalization,
3) generation of the first and second order density matrix,
4) improvement of the molecular orbitals.

The CI problem in steps two and three has two options, namely a determinant or configuration state function (CSF) many electron basis set. The choice of these is determined

by CISTEP in $MCSCF. More will be said just below about the differences between determinants and CSFs. The word "configuration" is used in this section to refer to either when a generic term is needed for the many-electron basis, so please note there is a distinction between this and the very similar term CSF.

The orbital problem in step four has three options, namely FOCAS, SOSCF, and FULLNR, listed here in order of their increasing mathematical sophistication, convergence characteristics, and of course, their computer resource requirements. Again, these are chosen by keywords in the $MCSCF group. More will be said just below about the relative merits of these three.

Finally, we mention again the QUAD converger, which works only for a CSF basis, in which the two optimization problems are treated simultaneously, for modest numbers of configuratations (50-100 is probably the limit). In principle, this is the most robust method available, but in practice, it has not received very much use compared to the unfolded methods.

Depending on the converger chosen, the program will select the appropriate kind of integral transformation. There's seldom need to try to fine tune this, but note that the $TRANS group does let you pick AO integral direct transformation with the DIRTRF flag.

On the first iteration at the first geometry, you will receive the normal amount of output from each of these stages, while each subsequent iterations will produce only a single summarizing line.

--- orbital updates ---

There are presently four orbital improvement options, namely FOCAS, SOSCF, FULLNR, and QUAD. All MCSCF orbital optimizations run in parallel. The four convergers are discussed briefly in the following paragraphs, in order of increasing robustness.

FOCAS is a first order, complete active space MCSCF optimization procedure. The FOCAS code was written by Michel Dupuis and Antonio Marquez at IBM. It is based on a novel approach due to Meier and Staemmler, using very fast but numerous microiterations to improve the convergence of what is intrinsically a first order method. Since FOCAS requires only one virtual orbital index in the integral transformation to compute the orbital gradient (aka the Lagrangian), the total MCSCF job may take less time than a second order method, even though it may require twice as many iterations to converge. The use of microiterations is crucial to FOCAS' ability to converge. It is important to take a great deal of care choosing the starting orbitals.

SOSCF is a method built upon the FOCAS code, which seeks to combine the speed of FOCAS with second order convergence properties. Thus SOSCF is an approximate

Newton-Raphson, based on a diagonal guess at the orbital hessian, and in fact has much in common with the SOSCF option in $SCF. Its time requirements per iteration are like FOCAS, with a convergence rate better than FOCAS but not as good as true second order. Storage of only the diagonal of the orbital hessian allows the SOSCF method to be used with much larger basis sets than exact second order methods. Because it usually requires the least CPU time, disk space, and memory needs, SOSCF is the default. Good convergence by the SOSCF method requires that you prepare starting orbitals carefully, and read in all MOs in $VEC, as the provision of canonicalized virtual orbitals increases the diagonal dominance of the orbital hessian.

FULLNR means a full Newton-Raphson orbital improvement step is taken, using the exact orbital hessian. FULLNR is a quite powerful convergence method, and normally takes the fewest iterations to converge. Computing the exact orbital hessian requires two virtual orbital indices be included in the transformation, making this step quite time consuming, and of course memory for storage of the orbital hessian must be available. Because both the transformation and orbital improvement steps of FULLNR are time consuming, FULLNR is not the default. You may want to try FULLNR when convergence is difficult, assuming you have already tried preparing good starting orbitals by the hints below.

The FULLNR MCSCF code in PC GAMESS is adapted from the HONDO7 program, and was written by Michel Dupuis at IBM. It uses the the augmented hessian matrix approach to solve the Newton-Raphson equations. There are two suboptions for computation of the orbital hessian. DM2 is the fastest but takes more memory than TEI.

QUAD uses a fully quadratic, or second order approach and is thus the most powerful MCSCF converger. The QUAD code is also adapted from Michel Dupuis's HONDO. QUAD runs begin with unfolded FULLNR iterations, until the orbitals approach convergence sufficiently. QUAD then begins the simultaneous optimization of CI coefficients and orbitals, and convergence should be obtained in 3-4 additional MCSCF iterations. The QUAD method requires building the full hessian, including orbital/orbital, orbital/CI, and CI/CI blocks, which is a rather big matrix. QUAD may be helpful in converging excited electronic states, but note that you may not use state averaging with QUAD. QUAD is a memory hog, and so may be used only for fairly small numbers of configurations.

The input to control the orbital update step is the $MCSCF group, where you should pick one of the convergence procedures. Most of the input in this group is rather specialized, but note in particular MAXIT and ACURCY which control the convergence behaviour. In some circumstances the diagonalizations of the core and virtual orbitals to canonicalize these (after overall MCSCF convergence) may produce spatial orbital symmetry loss, particularly in point groups with degeneracy present, and then CANONC might be used to turn this canonicalization off.

--- CI coefficient optimization ---

Determinants or configuration state functions (CSFs) may be used to form the many electron basis set. It is necessary to explain these in a bit of detail so that you can understand the advantages of each.

A determinant is a simple object: a product of spin orbitals with a given Sz quantum number, that is, the number of alpha spins and number of beta spins are a constant. Matrix elements involving determinants are correspondingly simple, but unfortunately determinants are not necessarily eigenfunctions of the S**2 operator.

To expand on this point, consider the four familiar 2e- functions which satisfy the Pauli principle. Here u, v are space orbitals, and a, b are the alpha and beta spin functions. As you know, the singlet and triplets are:

```
S1 = (uv + vu)/sqrt(2) * (ab - ba)/sqrt(2)
T1 = (uv) * aa
T2 = (uv - vu)/sqrt(2) * (ab + ba)/sqrt(2)
T3 = (uv) * bb
```

It is a simple matter to multiply out S1 and T2, and to expand the two determinants which have Sz=0,

```
D1 = |ua vb|          D2 = |va ub|
```

This reveals that

```
S1 = (D1+D2)/sqrt(2)   or   D1 = (S1 + T2)/sqrt(2)
T2 = (D1-D2)/sqrt(2)        D2 = (S1 - T2)/sqrt(2)
```

Thus, one must take a linear combination of determinants in order to have a wavefunction with the desired total spin. There are two important points to note:

a) A two by two Hamiltonian matrix over D1 and D2 has eigenfunctions with -different- spins, S=0 and S=1.
b) use of all determinants with Sz=0 does allow for the construction of spin adapted states. D1+D2, or D1-D2, are -not- spin contaminated.

By itself, a determinant such as D1 is said to be "spin contaminated", being a fifty-fifty admixture of singlet and triplet (it is curious that calculations with just one such determinant are often called "singlet UHF"). Of course, some determinants are spin adapted all by themselves, for example the spin adapted functions T1 and T3 above are single determinants, as are the closed shells

```
S2 = (uu) * (ab - ba)/sqrt(2).
S3 = (vv) * (ab - ba)/sqrt(2).
```

It is possible to perform a triplet calculation, with no singlet states present, by choosing determinants with Sz=1 such as T1, since then no state with Sz=0 as is required when S=0 exists in the determinant basis set. To summarize, the eigenfunctions of a Hamiltonian formed by determinants with any particular Sz will be spin states with S=Sz, S=Sz+1, S=Sz+2, ... but will not contain any S values smaller than Sz.

CSFs are an antisymmetrized combination of a space orbital product, and a spin adapted linear combination of simple alpha-beta products. Namely, the following CSF

```
C1 = A (uv) * (ab-ba)/sqrt(2)
```

which has a singlet spin function is identical to S1 above if you write out what the antisymmetrizer A does, and the CSFs

```
       C2 = A (uv) * aa
       C3 = A (uv - vu)/sqrt(2) * (ab + ba)/sqrt(2)
       C4 = A (uv) * bb
```

equal T1-T3. Since the three triplet CSFs have the same energy, PC GAMESS works with the simpler form C2. Singlet and triplet computations using CSFs are done in separate runs, because when spin-orbit coupling is not considered, the Hamiltonian is block diagonal in a CSF basis.

Technical information about the CSFs are that they use Yamanouchi-Kotani spin couplings, and matrix elements are obtained using a GUGA, or graphical unitary group approach.

The determinant implementation in PC GAMESS is written at Ames Laboratory, and can perform only full CI computations, meaning its primary use is for MCSCF wavefunctions of the full active space type. The CSF code is capable of more general CI computations, and so can be used for first or second order CI computations. Other comparisons between the determinant and CSF implementations, as they exist in PC GAMESS today, are

	determinants	CSFs
parallel execution	yes	yes
direct CI	yes	yes
exploits space symmetry	yes	yes
state average mixed spins	yes	no
first order density	yes	yes
state averaged densities	yes	yes
can form CI Lagrangian	no	yes

In cases where there is high spatial symmetry, this and the restriction to spin-adapted functions, makes the CSF code the fastest. However, the determinant code generates matrix elements very quickly, cases with less spatial symmetry run faster with determinants, and because it is a direct implementation, determinants do not use disk space to store Hamiltonian elements. Finally we note the initial guess of the CI eigenvector in the determinant code is much better than in the CSF code, where sometimes convergence to excited roots instead of lower ones occurs. On balance, the default CISTEP in $MCSCF selects the ALDET, the Ames Laboratory determinant CI code.

The next two sections describe in detail the input for specification of the configurations, either determinants or CSFs.

--- determinant CI ---

The determinant CI code is capable only of full CI wavefunctions. The $DET group is relatively simple. Many runs can be done by specifying only the orbital and electron counts: NCORE, NACT, and NELS. The number of electrons is 2*NCORE+NELS, and will be checked against the charge implied by ICHARG. The MULT given in $CONTRL is used to determine the desired Sz value, by extracting S from MULT=2S+1, then by default Sz=S. If

you wish to include lower spin multiplicities, which will increase the CPU time of the run, but will let you know what the energies of such states are, just input a smaller value for SZ. The states whose orbitals will be MCSCF optimized will be those having the requested MULT value, unless you choose otherwise with the PURES flag.

The remaining parameters in the $DET group give extra control over the diagonalization process. Most are not given in normal circumstances, except NSTATE, which you may need to adjust to produce enough roots of the desired MULT value. The only important keyword which has not been discussed is the WSTATE array, giving the weights for each state in forming the first and second order density matrix elements, which drive the orbital update methods. Note that analytic gradients are available only when the WSTATE array is a unit vector, corresponding to a pure state, such as WSTATE(1)=0,1,0 which permits gradients of the first excited state to be computed. When used for state averaged MCSCF, WSTATE is normalized to a unit sum, thus input of WSTATE(1)=1,1,1 really means a weight of 0.33333... for the each of the states being averaged.

If you are performing a CITYP=ALDET calculation, you give the input defining this full CI in a $CIDET group, containing the same keywords just mentioned.

--- CSF CI ---

The GUGA-based CSF package was originally a set of different programs, so the input to control it is spread over several input groups. The CSFs are specified by a $CIDRT group in the case of CITYP=GUGA, and by a $DRT group for MCSCF wavefunctions. Thus it is possible to perform an MCSCF defined by a $DRT input (or perhaps using $DET during the MCSCF), and follow this with a second order CI defined by a $CIDRT group, in the same run.

The remaining input groups used by the GUGA CSFs are $CISORT, $GUGEM, $GUGDIA, and $GUGDM2 for MCSCF runs, with the latter two being the most important, and in the case of CI computations, $GUGDM and possibly $LAGRAN groups are relevant. Perhaps the most interesting variables outside the $DRT/$CIDRT group are NSTATE in $GUGDIA to include excited states in the CI computation, IROOT in $GUGDM to select the state for properties, and WSTATE in $GUGDM2 to control which (average) state's orbitals are optimized.

The $DRT and $CIDRT groups are almost the same, with the only difference being orbitals restricted to double occupancy are called MCC in $DRT, and FZC in $CIDET. Therefore the rest of this section refers only to "$DRT".

The CSFs are specified by giving a reference CSF, together with a maximum degree of electron excitation from that single CSF. The MOs in the reference CSF are filled in the order MCC or FZC first, followed by DOC, AOS, BOS, ALP, VAL, and EXT (the Aufbau principle). AOS, BOS, and ALP are singly occupied MOs. ALP means a high spin alpha

coupling, while AOS/BOS are an alpha/beta coupling to an

open shell singlet. This requires the value NAOS=NBOS, and their MOs alternate. An example is

```
    NFZC=1 NDOC=2 NAOS=2 NBOS=2 NALP=1 NVAL=3
```

which gives the reference CSF

```
    FZC,DOC,DOC,AOS,BOS,AOS,BOS,ALP,VAL,VAL,VAL
```

This is a doublet state with five unpaired electrons. VAL orbitals are unoccupied only in the reference CSF, they will become occupied as the other CSFs are generated. This is done by giving an excitation level, either explicitly by the IEXCIT variable, or implicitly by the FORS, FOCI, or SOCI flags. One of these four keywords must be chosen, and during MCSCF runs, this is usually FORS.

Consider another simpler example, for an MCSCF run,

```
      NMCC=3 NDOC=3 NVAL=2
```

which gives the reference CSF

```
      MCC,MCC,MCC,DOC,DOC,DOC,VAL,VAL
```

having six electrons in five active orbitals. Usually, MCSCF calculations are usually of the Full Optimized Reaction Space (FORS) type. Some workers refer to FORS as CASSCF, complete active space SCF. These are the same, but the keyword is spelled FORS. In the present instance, choosing FORS=.TRUE. gives an excitation level of 4, as the 6 valence electrons have only 4 holes available for excitation. MCSCF runs typically have only a small number of VAL orbitals. It is common to summarize this example as "six electrons in five orbitals".

The next example is a first or second order multi-reference CI wavefunction, where

```
      NFZC=3 NDOC=3 NVAL=2 NEXT=-1
```

leads to the reference CSF

```
      FZC,FZC,FZC,DOC,DOC,DOC,VAL,VAL,EXT,EXT,...
```

FOCI or SOCI is chosen by selecting the appropriate flag, the correct excitation level is automatically generated. Note that the negative one for NEXT causes all remaining MOs to be included in the external orbital space. One way of viewing FOCI and SOCI wavefunctions is as all singles, or all singles and doubles, from the entire MCSCF wave-function as a reference. An equivalent way of saying this is that all CSFs with N electrons (in this case N=6) distributed in the valence orbitals in all ways (that is the FORS MCSCF wavefunction) to make the base wavefunction. To this, FOCI adds all CSFs with N-1 electrons in active and 1 electron in external orbitals. SOCI adds all CSFs with N-2 electrons in active orbitals and 2 in external orbitals. SOCI is often prohibitively large, but is also a very accurate wavefunction.

Sometimes people use the CI package for ordinary single reference CI calculations, such as

```
        NFZC=3 NDOC=5 NVAL=34
```

which means the reference RHF wavefunction is

```
        FZC FZC FZC DOC DOC DOC VAL VAL ... VAL
```

and in this case NVAL is a large number conveying the total number of -virtual- orbitals into which electrons

are excited. The excitation level would be given as IEXCIT=2, perhaps, to perform a SD-CI. All excitations smaller than the value of IEXCIT are automatically included in the CI. Note that NVAL's spelling was chosen to make the most sense for MCSCF calculations, and so it is a bit of a misnomer here.

Before going on, there is a quirk related to single reference CI that should be mentioned. Whenever the single reference contains unpaired electrons, such as

```
NFZC=3 NDOC=4 NALP=2 NVAL=33
```

some "extra" CSFs will be generated. The reference here can be abbreviated

```
2222 11 000 000 000 000 000 000 000 000 000 000 000
```

Supposing IEXCIT=2, the following CSF

```
2200 22 000 011 000 000 000 000 000 000 000 000 000
```

will be generated and used in the CI. Most people would prefer to think of this as a quadrupole excitation from the reference, but acting solely on the reasoning that no more than two electrons went into previously vacant NVAL orbitals, the GUGA CSF package decides it is a double. So, an open shell SD-CI calculation with PC GAMESS will not give the same result as other programs, although the result for any such calculation with these "extras" is correctly computed.

As was discussed above, the CSFs are automatically spin-symmetry adapted, with S implicit in the reference CSF. The spin quantum number you appear to be requesting in $DRT (basically, S = NALP/2) will be checked against the value of MULT in $CONTRL, and the total number of electrons, 2*NMCC(or NFZC) + 2*NDOC + NAOS + NBOS + NALP will be checked against the input given for ICHARG.

The CSF package is also able to exploit spatial symmetry, which like the spin and charge, is implicitly determined by the choice of the reference CSF. The keyword GROUP in $DRT governs the use of spatial symmetry.

The CSF program works with Abelian point groups, which are D2h and any of its subgroups. However, $DRT allows the input of some (but not all) higher point groups. For non-Abelian groups, the program automatically assigns the orbitals to an irrep in the highest possible Abelian subgroup. For the other non-Abelian groups, you must at present select GROUP=C1. Note that when you are computing a Hessian matrix, many of the displaced geometries are asymmetric, hence you must choose C1 in $DRT (however, be sure to use the highest symmetry possible in $DATA!).

The symmetry of the reference CSF given in your $DRT determines the symmetry of the CSFs which are generated. As an example, consider a molecule with Cs symmetry, and these two reference CSFs

```
...MCC...DOC DOC VAL VAL
...MCC...DOC AOS BOS VAL
```

Suppose that the 2nd and 3rd active MOs have symmetries a'

and a". Both of these generate singlet wavefunctions, with 4 electrons in 4 active orbitals, but the former constructs 1-A' CSFs, while the latter generates 1-A" CSFs. However, if the 2nd and 3rd orbitals have the same symmetry type, an identical list of CSFs is generated.

In cases with high point group symmetry, it may be possible to generate correct state degeneracies only by using no symmetry (GROUP=C1) when generating CSFs. As an example, consider the 2-pi ground state of NO. If you use GROUP=C4V, which will be mapped into its highest Abelian subgroup C2v, the two components of the pi state will be seen as belonging to different irreps, B1 and B2. The only way to ensure that both sets of CSFs are generated is to enforce no symmetry at all, so that CSFs for both components of the pi level are generated. This permits state averaging (WSTATE(1)=0.5,0.5) to preserve cylindrical symmetry. It is however perfectly feasible to use C4v or D4h symmetry in $DRT when treating sigma states.

The use of spatial symmetry decreases the number of CSFs, and thus the size of the Hamiltonian that must be computed. In molecules with high symmetry, this may lead to faster run times with the GUGA CSF code, compared to the determinant code.

--- starting orbitals ---

The first step is to partition the orbital space into core, active, and external sets, in a manner which is sensible for your chemical problem. This is a bit of an art, and the user is referred to the references quoted at the end of this section. Having decided what MCSCF to perform, you now must consider the more pedantic problem of what orbitals to begin the MCSCF calculation with.

You should always start an MCSCF run with orbitals from some other run, by means of GUESS=MOREAD. Do not expect to be able to use HCORE or HUCKEL! Example 6 is a poor example, converging only because methylene has so much symmetry, and the basis is so small. If you are beginning your MCSCF problem, use orbitals from some appropriate converged SCF run. Thus, a realistic example of an MCSCF calculation is examples 8 and 9. Once you get an MCSCF to converge, you can and should use these MCSCF MOs (which will be Schmidt orthogonalized) at other nearby geometries.

Starting from SCF orbitals can take a little bit of care. Most of the time (but not always) the orbitals you want to correlate will be the highest occupied orbitals in the SCF. Fairly often, however, the correlating orbitals you wish to use will not be the lowest unoccupied virtuals of the SCF. You will soon become familiar with NORDER=1 in $GUESS, as reordering is needed in 50% or more cases.

The occupied and especially the virtual canonical SCF MOs are often spread out over regions of the molecule other than "where the action is". Orbitals which remedy this can generated by two additional options at almost no CPU cost.

One way to improve upon the SCF orbitals as starting MOs is to generate modified virtual orbitals (MVOs). MVOs are obtained by diagonalizing the Fock operator of a very positive ion, within the virtual orbital space only. As implemented in PC GAMESS, MVOs can be obtained at the end of any RHF, ROHF, or GVB run by setting MVOQ in $SCF nonzero, at the cost of a single SCF cycle. Typically, we use MVOQ=+6. Generating MVOs does not change any of the occupied SCF orbitals of the original neutral, but gives more valence-like LUMOs.

Another way to improve SCF starting orbitals is by a partial localization of the occupied orbitals. Typically MCSCF active orbitals are concentrated in the part of the molecule where bonds are breaking, etc. Canonical SCF MOs are normally more spread out. By choosing LOCAL=BOYS along with SYMLOC=.TRUE. in $LOCAL, you can get orbitals which are localized, but still retain orbital symmetry to help speed the MCSCF along. In groups with an inversion center, a SYMLOC Boys localization does not change the orbitals, but you can instead use LOCAL=POP. Localization tends to order the orbitals fairly randomly, so be prepared to reorder them appropriately.

Pasting the virtuals from a MVOQ run onto the occupied orbitals of a SYMLOC run (both can be done in the same SCF computation) gives the best possible set of starting orbitals. If you also take the time to design your active space carefully, select the appropriate starting orbitals from this combined $VEC, and inspect your converged results, you will be able to carry out MCSCF computations correctly.

Convergence of MCSCF is by no means guaranteed. Poor convergence can invariably be traced back to either a poor initial selection of orbitals, or a poorly chosen number of active orbitals. My advice is, before you even start:

"Look at the orbitals.
Then look at the orbitals again".

Later, if you have any trouble:

"Look at the orbitals some more".

Few people are able to see the orbital shapes in the LCAO matrix in a log file, and so need a visualization program.

Even if you don't have any trouble, look at the orbitals to see if they converged to what you expected, and have reasonable occupation numbers. MCSCF is by no means the sort of "black box" that RHF is these days, so LOOK VERY CAREFULLY AT YOUR RESULTS.

--- miscellaneous hints ---

It is very helpful to execute a EXETYP=CHECK run before doing any MCSCF or CI run. The CHECK run will tell you the total number of configurations and the charge and multiplicity based on your $DET, or if you are using a $DRT, the electronic state as well. The CHECK run also lets the program feel out the memory that will be required to actually do the run. Thus the CHECK run can potentially

prevent costly mistakes, or tell you when a calculation is prohibitively large.

A very common MCSCF wavefunction has 2 electrons in 2 active MOs. This is the simplest possible wavefunction describing a singlet diradical. While this function can be obtained in an MCSCF run (using NACT=2 NELS=2 or NDOC=1 NVAL=1), it can be obtained much faster by use of the GVB code, with one GVB pair. This GVB-PP(1) wavefunction is also known in the literature as two configuration SCF, or TCSCF. The two configurations of this GVB are equivalent to the three configurations used in this MCSCF, as orbital optimization in natural form (configurations 20 and 02) causes the coefficient of the 11 configuration to vanish.

If you are using many configurations (say 75,000 or more), the main bottleneck in the PC GAMESS CI/MCSCF code is the formation and diagonalization of the Hamiltonian, not the integral transformation and orbital improvement steps. In this case, you would be wise to switch to FULLNR, which will minimize the total number of iterations. In addition, each orbital improvement may contain some microiterations, which consists of an integral transformation over the new MOs, followed immediately by a orbital improvement, reusing the current 2nd order density matrix. This avoids the CI diagonalization step, which may be of some use in MCSCF calculations with a large number of configurations. Since the determinant CI is a direct CI, it is probably better to use it in this circumstance to avoid the very large disk file used to store the CSF Hamiltonian, and its associated I/O bottleneck.

If you choose an excitation level IEXCIT smaller than that needed to generate the FORS space, you must use the FULLNR method (FOCAS and SOSCF convergers assume complete active spaces). Be sure to set FORS=.FALSE. in $MCSCF or else very poor convergence will result. Actually, the convergence for incomplete active spaces is likely to be awful, anyway.

--- references ---

There are several review articles about MCSCF listed below. Of these, the first two are a nice overview of the subject, the final 3 are more technical.

1. "The Construction and Interpretation of MCSCF wavefunctions"
   M.W.Schmidt and M.S.Gordon,
   Ann.Rev.Phys.Chem. 49,233-266(1998)

2a. "The Multiconfiguration SCF Method"
   B.O.Roos, in "Methods in Computational Molecular Physics", edited by G.H.F.Diercksen and S.Wilson
   D.Reidel Publishing, Dordrecht, Netherlands, 1983, pp 161-187.

2b. "The Multiconfiguration SCF Method"
   B.O.Roos, in "Lecture Notes in Quantum Chemistry", edited by B.O.Roos, Lecture Notes in Chemistry v58,
   Springer-Verlag, Berlin, 1994, pp 177-254.

3. "Optimization and Characterization of a MCSCF State"
J.Olsen, D.L.Yeager, P.Jorgensen
Adv.Chem.Phys. 54, 1-176(1983).
4. "Matrix Formulated Direct MCSCF and Multiconfiguration Reference CI Methods"
H.-J.Werner, Adv.Chem.Phys. 69, 1-62(1987).
5. "The MCSCF Method"
R.Shepard, Adv.Chem.Phys. 69, 63-200(1987).

There is an entire section on the choice of active spaces in Reference 1. As this is a matter of great importance, here are two alternate presentations of the design of active spaces:

6. "The CASSCF Method and its Application in Electronic Structure Calculations"
B.O.Roos, in "Advances in Chemical Physics", vol.69, edited by K.P.Lawley, Wiley Interscience, New York, 1987, pp 339-445.
7. "Are Atoms Intrinsic to Molecular Electronic Wavefunctions?"
K.Ruedenberg, M.W.Schmidt, M.M.Gilbert, S.T.Elbert
Chem.Phys. 71, 41-49, 51-64, 65-78 (1982).

Two papers germane to the FOCAS implementation are

8. "An Efficient first-order CASSCF method based on the renormalized Fock-operator technique."
U.Meier, V.Staemmler Theor.Chim.Acta 76, 95-111(1989)
9. "Modern tools for including electron correlation in electronic structure studies"
M.Dupuis, S.Chen, A.Marquez, in "Relativistic and Electron Correlation Effects in Molecules and Solids", edited by G.L.Malli, Plenum, NY 1994

The paper germane to the the SOSCF method is

10. "Approximate second order method for orbital optimization of SCF and MCSCF wavefunctions"
G.Chaban, M.W.Schmidt, M.S.Gordon
Theor.Chem.Acc. 97: 88-95(1997)

Two papers germane to the FULLNR implementation are

11. "General second order MCSCF theory: A Density Matrix Directed Algorithm"
B.H.Lengsfield, III, J.Chem.Phys. 73,382-390(1980).
12. "The use of the Augmented Matrix in MCSCF Theory"
D.R.Yarkony, Chem.Phys.Lett. 77,634-635(1981).

For determinants and CSFs, respectively, see

13. "A new determinant-based full CI method"
P.J.Knowles, N.C.Handy
Chem.Phys.Lett. 111,315-321(1984)
14. "The GUGA approach to the electron correlation problem"
B.R.Brooks, H.F.Schaefer
J.Chem.Phys. 70, 5092-5106(1979)

The final references are simply some examples of FORS MCSCF applications, the latter done with GAMESS (US).

15. D.F.Feller, M.W.Schmidt, K.Ruedenberg,
J.Am.Chem.Soc. 104, 960-967(1982).
16. M.W.Schmidt, P.N.Truong, M.S.Gordon,
J.Am.Chem.Soc. 109, 5217-5227(1987).

Second order perturbation theory

The perturbation theory techniques available in PC GAMESS expand to the second order energy correction only, but permit use of a broad range of zeroth order wavefunctions. Since MP2 theory for systems well described as closed shells recovers only about 80% of the correlation energy (assuming the use of large basis sets), it is common to extend the perturbative treatment to higher order, or to use coupled cluster theory. While this is reasonable for systems well described by RHF or UHF with small spin contamination, this is probably a poor approach when the system is not well described at zeroth order by these wavefunctions.

The input for second order pertubation calculations based on SCFTYP=RHF, UHF, or ROHF is found in $MP2, while for SCFTYP=MCSCF, see $MCQDPT.

--- RHF and UHF MP2

These methods are well defined, due to the uniqueness of the Fock matrix definitions. These methods are also well understood, and there is little need to say more.

One point which may not be commonly appreciated is that the density matrix for the first order wavefunction for the RHF case, which is generated during gradient runs or if properties are requested in the $MP2 group, is of the type known as "response density", which differs from the more usual "expectation value density". The eigenvalues of the response density matrix (which are the occupation numbers of the MP2 natural orbitals) can therefore be greater than 2 for frozen core orbitals, or even negative values for the highest 'virtual' orbitals. The sum is of course exactly the total number of electrons. We have seen values outside the range 0-2 in several cases where the single configuration RHF wavefunction was not an appropriate description of the system, and thus these occupancies may serve as a guide to the wisdom of using a RHF reference.

RHF energy corrections are the only method currently programmed for parallel computation. The analytic energy gradient is available only for RHF references, and this does permit frozen cores.

--- high spin ROHF MP2

There are a number of open shell perturbation theories described in the literature. It is important to note that these methods give different results for the second order energy correction, reflecting ambiguities in the selection of the zeroth order Hamiltonian and in defining the ROHF Fock matrices. Two of these are available in PC GAMESS.

One theory is known as RMP, which it should be pointed out, is entirely equivalent to the ROHF-MBPT2 method. The theory is as UHF-like as possible, and can be chosen in PC GAMESS by selection of OSPT=RMP in $MP2. The second order energy is defined by

1. P.J.Knowles, J.S.Andrews, R.D.Amos, N.C.Handy, J.A.Pople Chem.Phys.Lett. 186, 130-136(1991)
2. W.J.Lauderdale, J.F.Stanton, J.Gauss, J.D.Watts, R.J.Bartlett Chem.Phys.Lett. 187, 21-28(1991).

The submission dates are in inverse order of publication dates, and -both- papers should be cited when using this method. Here we will refer to the method as RMP in keeping with much of the literature. The RMP method diagonalizes the alpha and beta Fock matrices separately, so that their occupied-occupied and virtual-virtual blocks are canonicalized. This generates two distinct orbital sets, whose double excitation contributions are processed by the usual UHF MP2 program, but an additional energy term from single excitations is required.

RMP's use of different orbitals for different spins adds to the CPU time required for integral transformations, of course. RMP is invariant under all of the orbital transformations for which the ROHF itself is invariant. Unlike UMP2, the second order RMP energy does not suffer from spin contamination, since the reference ROHF wave-function has no spin contamination. The RMP wavefunction, however, is spin contaminated at 1st and higher order, and therefore the 3rd and higher order RMP energies are spin contaminated. Other workers have extended the RMP theory to gradients and hessians at second order, and to fourth order in the energy,

3. W.J.Lauderdale, J.F.Stanton, J.Gauss, J.D.Watts, R.J.Bartlett J.Chem.Phys. 97, 6606-6620(1992)
4. J.Gauss, J.F.Stanton, R.J.Bartlett J.Chem.Phys. 97, 7825-7828(1992)
5. D.J.Tozer, J.S.Andrews, R.D.Amos, N.C.Handy Chem.Phys.Lett. 199, 229-236(1992)
6. D.J.Tozer, N.C.Handy, R.D.Amos, J.A.Pople, R.H.Nobes, Y.Xie, H.F.Schaefer Mol.Phys. 79, 777-793(1993)

We deliberately omit references to the ROMP precurser to the RMP formalism.

The ZAPT formalism is also implemented in PC GAMESS, as OSPT=ZAPT in $MP2. Because this theory is not spin-contaminated at any order, and has only a single set of orbitals in the MO transformation, it is the default. References for ZAPT (Z-averaged perturbation theory) are

7. T.J.Lee, D.Jayatilaka Chem.Phys.Lett. 201, 1-10(1993)
8. T.J.Lee, A.P.Rendell, K.G.Dyall, D.Jayatilaka J.Chem.Phys. 100, 7400-7409(1994)

The formulae for the seven terms in the energy are most clearly summarized in the paper

9. I.M.B.Nielsen, E.T.Seidl
   J.Comput.Chem. 16, 1301-1313(1995)

We would like to thank Tim Lee for very gracious assistance in the implementation of ZAPT.

There are a number of other open shell theories, with names such as HC, OPT1, OPT2, and IOPT. The literature for these is

10. I.Hubac, P.Carsky Phys.Rev.A 22, 2392-2399(1980)
11. C.Murray, E.R.Davidson
    Chem.Phys.Lett. 187,451-454(1991)
12. C.Murray, E.R.Davidson
    Int.J.Quantum Chem. 43, 755-768(1992)
13. P.M.Kozlowski, E.R.Davidson
    Chem.Phys.Lett. 226, 440-446(1994)
14. C.W.Murray, N.C.Handy
    J.Chem.Phys. 97, 6509-6516(1992)
15. T.D.Crawford, H.F.Schaefer, T.J.Lee
    J.Chem.Phys. 105, 1060-1069(1996)

The latter two of these give comparisons of the various high spin methods, and the numerical results in ref. 15 are the basis for the conventional wisdom that restricted open shell theory is better convergent with order of the perturbation level than unrestricted theory. Paper 8 has some numerical comparisons of spin-restricted theories as well. We are aware of one paper on low-spin coupled open shell SCF perturbation theory

16. J.S.Andrews, C.W.Murray, N.C.Handy
    Chem.Phys.Lett. 201, 458-464(1993)

but this is not implemented in PC GAMESS. See the MCQDPT code for cases such as this.

--- GVB based MP2

This is not implemented in PC GAMESS. Note that the MCSCF MP2 program discussed below should be able to develop the perturbation correction for open shell singlets, by using a $DRT input such as

```
NMCC=N/2-1 NDOC=0 NAOS=1 NBOS=1 NVAL=0
```

which generates a single CSF if the two open shells have different symmetry, or for a one pair GVB function

```
NMCC=N/2-1 NDOC=1 NVAL=1
```

which generates a 3 CSF function entirely equivalent to the two configuration TCSCF, a.k.a GVB-PP(1). For the record, we note that if we attempt a triplet state with the MCSCF program,

```
NMCC=N/2-1 NDOC=0 NALP=2 NVAL=0
```

we get a result equivalent to the OPT1 open shell method described above, not the RMP result. It is possible to generate the orbitals with a simpler SCF computation than the MCSCF $DRT examples just given, and read them into the MCSCF based MP2 program described below, by INORB=1.

--- MCSCF based MP2

Just as for the open shell case, there are several ways to define a multireference perturbation theory. The most noteworthy are the CASPT2 method of Roos' group, the MRMP2 method of Hirao, the MROPTn methods of Davidson, and the MCQDPT2 method of Nakano. Although the results of each method are different, energy differences should be rather similar. In particular, the MCQDPT2 method implemented in PC GAMESS gives results for the singlet-triplet splitting of methylene in close agreement to CASPT2, MRMP2(Fav), and MROPT1, and differs by 2 Kcal/mole from MRMP2(Fhs), and the MROPT2 to MROPT4 methods.

The MCQDPT method implemented in PC GAMESS is a multistate perturbation theory. If applied to 1 state, it is similar to the MRMP model of Hirao. When applied to more than one state, it is of the philosophy "perturb first, diagonalize second". This means that perturbations are made to both the diagonal and offdiagonal elements of an effective Hamiltonian, whose dimension equals the number of states being treated. The perturbed Hamiltonian is diagonalized to give the corrected state energies. Diagonalization after inclusion of the offdiagonal perturbation ensures that avoided crossings of states of the same symmetry are treated correctly. Such an avoided crossing is found in the LiF molecule, as shown in the first of the two papers on the MCQDPT method:

H.Nakano, J.Chem.Phys. 99, 7983-7992(1993)
H.Nakano, Chem.Phys.Lett. 207, 372-378(1993)

The closely related "diagonalize, then perturb" MRMP model is discussed by

K.Hirao, Chem.Phys.Lett. 190, 374-380(1992)
K.Hirao, Chem.Phys.Lett. 196, 397-403(1992)
K.Hirao, Int.J.Quant.Chem. S26, 517-526(1992)
K.Hirao, Chem.Phys.Lett. 201, 59-66(1993)

Single state MCQDPT computations are very similiar to MRMP computations. A beginning set of references to the other multireference methods used includes:

P.M.Kozlowski, E.R.Davidson
J.Chem.Phys. 100, 3672-3682(1994)
K.G.Dyall J.Chem.Phys. 102, 4909-4918(1995)
B.O.Roos, K.Andersson, M.K.Fulscher, P.-A.Malmqvist,
L.Serrano-Andres, K.Pierloot, M.Merchan
Adv.Chem.Phys. 93, 219-331(1996).

We close the discussion with an input example which illustrates RMP and MCQDPT computations on the ground state of NH2 radical:

```
!  2nd order perturbation test on NH2, following
!  T.J.Lee, A.P.Rendell, K.G.Dyall, D.Jayatilaka
!  J.Chem.Phys. 100, 7400-7409(1994), Table III.
!  State is 2-B-1, 69 AOs, 49 CSFs.
!
!  For 1 CSF reference,
!    E(ROHF) = -55.5836109825
!     E(RMP) = -55.7772299929   (lit. RMP = -75.777230)
!  E(MCQDPT) = -55.7830423024   (lit. OPT1= -75.783044)
! [E(MCQDPT) = -55.7830437413 at the lit's OPT1 geometry]
```

```
!
!  For 49 CSF reference,
!   E(MCSCF) = -55.6323324949
!  E(MCQDPT) = -55.7857458575
!
 $contrl scftyp=mcscf mplevl=2 runtyp=energy mult=2 $end
 $system timlim=60 memory=1000000 $end
 $guess  guess=moread norb=69 $end
 $mcscf  fullnr=.true. $end
!
!  Next two lines carry out a MCQDPT computation, after
!  carrying out a full valence MCSCF orbital optimization.
 $drt    group=c2v fors=.t. nmcc=1 ndoc=3 nalp=1 nval=2 $end
 $mcqdpt inorb=0 mult=2 nmofzc=1 nmodoc=0 nmoact=6
         istsym=3 nstate=1 $end
!
!  Next two lines carry out a single reference computation,
!  using converged ROHF orbitals from the $VEC.
--- $drt    group=c2v fors=.t. nmcc=4 ndoc=0 nalp=1 nval=0 $end
--- $mcqdpt inorb=1 nmofzc=1 nmodoc=3 nmoact=1
---         istsym=3 nstate=1 $end



 $data
NH2...2-B-1...TZ2Pf basis, RMP geom. used by Lee, et al.
Cnv  2

Nitrogen   7.0
  S 6
   1 13520.0    0.000760
   2  1999.0    0.006076
   3   440.0    0.032847
   4   120.9    0.132396
   5    38.47   0.393261
   6    13.46   0.546339
  S 2
   1    13.46   0.252036
   2     4.993  0.779385
  S 1 ; 1 1.569  1.0
  S 1 ; 1 0.5800 1.0
  S 1 ; 1 0.1923 1.0
  P 3
   1 35.91  0.040319
   2  8.480 0.243602
   3  2.706 0.805968
  P 1 ; 1 0.9921 1.0
  P 1 ; 1 0.3727 1.0
  P 1 ; 1 0.1346 1.0
  D 1 ; 1 1.654 1.0
  D 1 ; 1 0.469 1.0
  F 1 ; 1 1.093 1.0

Hydrogen   1.0  0.0 0.7993787 0.6359684
  S 3   ! note that this is unscaled
   1 33.64  0.025374
   2  5.058 0.189684
   3  1.147 0.852933
  S 1 ; 1 0.3211 1.0
  S 1 ; 1 0.1013 1.0
  P 1 ; 1 1.407 1.0
```

```
  P 1 ; 1 0.388 1.0
  D 1 ; 1 1.057 1.0

 $end
--- ROHF ORBITALS --- GENERATED AT 17:17:13 CST 28-OCT-1996
E(ROHF)= -55.5836109825, E(NUC)= 7.5835449477, 9 ITERS
 $VEC
 1  1 5.58322965E-01....
...omitted...
69 14 2.48059888E-02....
 $END
```

Geometry Searches and Internal Coordinates

Stationary points are places on the potential energy surface with a zero gradient vector (first derivative of the energy with respect to nuclear coordinates). These include minima (whether relative or global), better known to chemists as reactants, products, and intermediates; as well as transition states (also known as saddle points).

The two types of stationary points have a precise mathematical definition, depending on the curvature of the potential energy surface at these points. If all of the eigenvalues of the hessian matrix (second derivative of the energy with respect to nuclear coordinates) are positive, the stationary point is a minimum. If there is one, and only one, negative curvature, the stationary point is a transition state. Points with more than one negative curvature do exist, but are not important in chemistry. Because vibrational frequencies are basically the square roots of the curvatures, a minimum has all real frequencies, and a saddle point has one imaginary vibrational "frequency".

PC GAMESS locates minima by geometry optimization, as RUNTYP=OPTIMIZE, and transition states by saddle point searches, as RUNTYP=SADPOINT. In many ways these are similar, and in fact nearly identical FORTRAN code is used for both. The term "geometry search" is used here to describe features which are common to both procedures. The input to control both RUNTYPs is found in the $STATPT group.

As will be noted in the symmetry section below, an OPTIMIZE run does not always find a minimum, and a SADPOINT run may not find a transtion state, even though the gradient is brought to zero. You can prove you have located a minimum or saddle point only by examining the local curvatures of the potential energy surface. This can be done by following the geometry search with a RUNTYP=HESSIAN job, which should be a matter of routine.

* * * Quasi-Newton Searches * * *

Geometry searches are most effectively done by what is called a quasi-Newton-Raphson procedure. These methods assume a quadratic potential surface, and require the

exact gradient vector and an approximation to the hessian. It is the approximate nature of the hessian that makes the method "quasi". The rate of convergence of the geometry search depends on how quadratic the real surface is, and the quality of the hessian. The latter is something you have control over, and is discussed in the next section.

PC GAMESS contains different implementations of quasi-Newton procedures for finding stationary points, namely METHOD=NR, RFO, QA, and the seldom used SCHLEGEL. They differ primarily in how the step size and direction are controlled, and how the Hessian is updated. The CONOPT method is a way of forcing a geometry away from a minimum towards a TS. It is not a quasi-Newton method, and is described at the very end of this section.

The NR method employs a straight Newton-Raphson step. There is no step size control, the algorithm will simply try to locate the nearest stationary point, which may be a minimum, a TS, or any higher order saddle point. NR is not intended for general use, but is used by PC GAMESS in connection with some of the other methods after they have homed in on a stationary point, and by Gradient Extremal runs where location of higher order saddle points is common. NR requires a very good estimate of the geometry in order to converge on the desired stationary point.

The RFO and QA methods are two different versions of the so-called augmented Hessian techniques. They both employ Hessian shift parameter(s) in order to control the step length and direction.

In the RFO method, the shift parameter is determined by approximating the PES with a Rational Function, instead of a second order Taylor expansion. For a RUNTYP=SADPOINT, the TS direction is treated separately, giving two shift parameters. This is known as a Partitioned Rational Function Optimization (P-RFO). The shift parameter(s) ensure that the augmented Hessian has the correct eigen-value structure, all positive for a minimum search, and one negative eigenvalue for a TS search. The (P)-RFO step can have any length, but if it exceeds DXMAX, the step is simply scaled down.

In the QA (Quadratic Approximation) method, the shift parameter is determined by the requirement that the step size should equal DXMAX. There is only one shift parameter for both minima and TS searches. Again the augmented Hessian will have the correct structure. There is another way of describing the same algorithm, namely as a minimization on the "image" potential. The latter is known as TRIM (Trust Radius Image Minimization). The working equation is identical in these two methods.

When the RFO steplength is close to DXMAX, there is little difference between the RFO and QA methods. However, the RFO step may in some cases exceed DXMAX significantly, and a simple scaling of the step will usually not produce the best direction. The QA step is the best step on the

hypersphere with radius DXMAX. For this reason QA is the default algorithm.

Near a stationary point the straight NR algorithm is the most efficient. The RFO and QA may be viewed as methods for guiding the search in the "correct" direction when starting far from the stationary point. Once the stationary point is approached, the RFO and QA methods switch to NR, automatically, when the NR steplength drops below 0.10 or DXMAX, whichever is the smallest.

You should read the papers mentioned below in order to understand how these methods are designed to work. The first 3 papers describe the RFO and TRIM/QA algorithms A good but somewhat dated summary of various search procedures is given by Bell and Crighton, and see also the review by Schlegel. Most of the FORTRAN code for geometry searches, and some of the discussion in this section was written by Frank Jensen of Odense University, whose paper compares many of the algorithms implemented in GAMESS (US):

1. J.Baker J.Comput.Chem. 7, 385-395(1986)
2. T.Helgaker Chem.Phys.Lett. 182, 305-310(1991)
3. P.Culot, G.Dive, V.H.Nguyen, J.M.Ghuysen Theoret.Chim.Acta 82, 189-205(1992)
4. H.B.Schlegel J.Comput.Chem. 3, 214-218(1982)
5. S.Bell, J.S.Crighton J.Chem.Phys. 80, 2464-2475(1984).
6. H.B.Schlegel Advances in Chemical Physics (Ab Initio Methods in Quantum Chemistry, Part I), volume 67, K.P.Lawley, Ed. Wiley, New York, 1987, pp 249-286.
7. F.Jensen J.Chem.Phys. 102, 6706-6718(1995).

* * * the Hessian * * *

Although quasi-Newton methods require only an approximation to the true hessian, the choice of this matrix has a great affect on convergence of the geometry search.

There is a procedure contained within PC GAMESS for guessing a diagonal, positive definite hessian matrix, HESS=GUESS. If you are using Cartesian coordinates, the guess hessian is 1/3 times the unit matrix. The guess is more sophisticated when internal coordinates are defined, as empirical rules will be used to estimate stretching and bending force constants. Other force constants are set to 1/4. The diagonal guess often works well for minima, but cannot possibly find transition states (because it is positive definite). Therefore, GUESS may not be selected for SADPOINT runs.

Two options for providing a more accurate hessian are HESS=READ and CALC. For the latter, the true hessian is obtained by direct calculation at the initial geometry, and then the geometry search begins, all in one run. The READ option allows you to feed in the hessian in a $HESS

group, as obtained by a RUNTYP=HESSIAN job. The second procedure is actually preferable, as you get a chance to see the frequencies. Then, if the local curvatures look good, you can commit to the geometry search. Be sure to include a $GRAD group (if the exact gradient is available) in the HESS=READ job so that PC GAMESS can take its first step immediately.

Note also that you can compute the hessian at a lower basis set and/or wavefunction level, and read it into a higher level geometry search. In fact, the $HESS group could be obtained at the semiempirical level. This trick works because the hessian is 3Nx3N for N atoms, no matter what atomic basis is used. The gradient from the lower level is of course worthless, as the geometry search must work with the exact gradient of the wavefunction and basis set in current use. Discard the $GRAD group from the lower level calculation!

You often get what you pay for. HESS=GUESS is free, but may lead to significantly more steps in the geometry search. The other two options are more expensive at the beginning, but may pay back by rapid convergence to the stationary point.

The hessian update frequently improves the hessian for a few steps (especially for HESS=GUESS), but then breaks down. The symptoms are a nice lowering of the energy or the RMS gradient for maybe 10 steps, followed by crazy steps. You can help by putting the best coordinates into $DATA, and resubmitting, to make a fresh determination of the hessian.

The default hessian update for OPTIMIZE runs is BFGS, which is likely to remain positive definite. The POWELL update is the default for SADPOINT runs, since the hessian can develop a negative curvature as the search progresses. The POWELL update is also used by the METHOD=NR and CONOPT since the Hessian may have any number of negative eigenvalues in these cases. The MSP update is a mixture of Murtagh-Sargent and Powell, suggested by Josep Bofill, (J.Comput.Chem., 15, 1-11, 1994). It sometimes works slightly better than Powell, so you may want to try it.

* * * Coordinate Choices * * *

Optimization in cartesian coordinates has a reputation of converging slowly. This is largely due to the fact that translations and rotations are usually left in the problem. Numerical problems caused by the small eigenvalues associated with these degrees of freedom are the source of this poor convergence. The methods in PC GAMESS project the hessian matrix to eliminate these degrees of freedom, which should not cause a problem. Nonetheless, Cartesian coordinates are in general the most slowly convergent coordinate system.

The use of internal coordinates (see NZVAR in $CONTRL as well as $ZMAT) also eliminates the six rotational and translational degrees of freedom. Also, when internal

coordinates are used, the GUESS hessian is able to use empirical information about bond stretches and bends. On the other hand, there are many possible choices for the internal coordinates, and some of these may lead to much poorer convergence of the geometry search than others. Particularly poorly chosen coordinates may not even correspond to a quadratic surface, thereby ending all hope that a quasi-Newton method will converge.

Internal coordinates are frequently strongly coupled. Because of this, Jerry Boatz has called them "infernal coordinates"! A very common example to illustrate this might be a bond length in a ring, and the angle on the opposite side of the ring. Clearly, changing one changes the other simultaneously. A more mathematical definition of "coupled" is to say that there is a large off-diagonal element in the hessian. In this case convergence may be unsatisfactory, especially with a diagonal GUESS hessian, where a "good" set of internals is one with a diagonally dominant hessian. Of course, if you provide an accurately computed hessian, it will have large off-diagonal values where those are truly present. Even so, convergence may be poor if the coordinates are coupled through large 3rd or higher derivatives. The best coordinates are therefore those which are the most "quadratic".

One very popular set of internal coordinates is the usual "model builder" Z-matrix input, where for N atoms, one uses N-1 bond lengths, N-2 bond angles, and N-3 bond torsions. The popularity of this choice is based on its ease of use in specifying the initial molecular geometry. Typically, however, it is the worst possible choice of internal coordinates, and in the case of rings, is not even as good as Cartesian coordinates.

However, PC GAMESS does not require this particular mix of the common types. PC GAMESS' only requirement is that you use a total of 3N-6 coordinates, chosen from these 3 basic types, or several more exotic possibilities. (Of course, we mean 3N-5 throughout for linear molecules). These additional types of internal coordinates include linear bends for 3 collinear atoms, out of plane bends, and so on. There is no reason at all why you should place yourself in a straightjacket of N-1 bonds, N-2 angles, and N-3 torsions. If the molecule has symmetry, be sure to use internals which are symmetrically related.

For example, the most effective choice of coordinates for the atoms in a four membered ring is to define all four sides, any one of the internal angles, and a dihedral defining the ring pucker. For a six membered ring, the best coordinates seem to be 6 sides, 3 angles, and 3 torsions. The angles should be every other internal angle, so that the molecule can "breathe" freely. The torsions should be arranged so that the central bond of each is placed on alternating bonds of the ring, as if they were pi bonds in Kekule benzene. For a five membered ring, we suggest all 5 sides, 2 internal angles, again alternating every other one, and 2 dihedrals to fill in.

The internal angles of necessity skip two atoms where the ring closes. Larger rings should generalize on the idea of using all sides but only alternating angles. If there are fused rings, start with angles on the fused bond, and alternate angles as you go around from this position.

Rings and more especially fused rings can be tricky. For these systems, especially, we suggest the Cadillac of internal coordinates, the "natural internal coordinates" of Peter Pulay. For a description of these, see

P.Pulay, G.Fogarosi, F.Pang, J.E.Boggs,
J.Am.Chem.Soc. 101, 2550-2560 (1979).
G.Fogarasi, X.Zhou, P.W.Taylor, P.Pulay
J.Am.Chem.Soc. 114, 8191-8201 (1992).

These are linear combinations of local coordinates, except in the case of rings. The examples given in these two papers are very thorough.

An illustration of these types of coordinates is given in the example job EXAM25.INP, distributed with PC GAMESS. This is a nonsense molecule, designed to show many kinds of functional groups. It is defined using standard bond distances with a classical Z-matrix input, and the angles in the ring are adjusted so that the starting value of the unclosed OO bond is also a standard value.

Using Cartesian coordinates is easiest, but takes a very large number of steps to converge. This however, is better than using the classical Z-matrix internals given in $DATA, which is accomplished by setting NZVAR to the correct 3N-6 value. The geometry search changes the OO bond length to a very short value on the 1st step, and the SCF fails to converge. (Note that if you have used dummy atoms in the $DATA input, you cannot simply enter NZVAR to optimize in internal coordinates, instead you must give a $ZMAT which involves only real atoms).

The third choice of internal coordinates is the best set which can be made from the simple coordinates. It follows the advice given above for five membered rings, and because it includes the OO bond, has no trouble with crashing this bond. It takes 20 steps to converge, so the trouble of generating this $ZMAT is certainly worth it compared to the use of Cartesians.

Natural internal coordinates are defined in the final group of input. The coordinates are set up first for the ring, including two linear combinations of all angles and all torsions withing the ring. After this the methyl is hooked to the ring as if it were a NH group, using the usual terminal methyl hydrogen definitions. The H is hooked to this same ring carbon as if it were a methine. The NH and the CH2 within the ring follow Pulay's rules exactly. The amount of input is much greater than a normal Z-matrix. For example, 46 internal coordinates are given, which are then placed in 3N-6=33 linear combinations. Note that natural internals tend to be rich in bends, and short

on torsions.

The energy results for the three coordinate systems which converge are as follows:

NSERCH	Cart	good Z-mat	nat. int.
0	-48.6594935049	-48.6594935049	-48.6594935049
1	-48.6800538676	-48.6806631261	-48.6838361406
2	-48.6822702585	-48.6510215698	-48.6874045449
3	-48.6858299354	-48.6882945647	-48.6932811528
4	-48.6881499412	-48.6849667085	-48.6946836332
5	-48.6890226067	-48.6911899936	-48.6959800274
6	-48.6898261650	-48.6878047907	-48.6973821465
7	-48.6901936624	-48.6930608185	-48.6987652146
8	-48.6905304889	-48.6940607117	-48.6996366016
9	-48.6908626791	-48.6949137185	-48.7006656309
10	-48.6914279465	-48.6963767038	-48.7017273728
11	-48.6921521142	-48.6986608776	-48.7021504975
12	-48.6931136707	-48.7007305310	-48.7022405019
13	-48.6940437619	-48.7016095285	-48.7022548935
14	-48.6949546487	-48.7021531692	-48.7022569328
15	-48.6961698826	-48.7022080183	-48.7022570260
16	-48.6973813002	-48.7022454522	-48.7022570662
17	-48.6984850655	-48.7022492840	
18	-48.6991553826	-48.7022503853	
19	-48.6996239136	-48.7022507037	
20	-48.7002269303	-48.7022508393	
21	-48.7005379631		
22	-48.7008387759		
	...		
50	-48.7022519950		

from which you can see that the natural internals are actually the best set. The $ZMAT exhibits upward burps in the energy at step 2, 4, and 6, so that for the same number of steps, these coordinates are always at a higher energy than the natural internals.

The initial hessian generated for these three columns contains 0, 33, and 46 force constants. This assists the natural internals, but is not the major reason for its superior performance. The computed hessian at the final geometry of this molecule, when transformed into the natural internal coordinates is almost diagonal. This almost complete uncoupling of coordinates is what makes the natural internals perform so well. The conclusion is of course that not all coordinate systems are equal, and natural internals are the best. As another example, we have run the ATCHCP molecule, which is a popular geometry optimization test, due to its two fused rings:

H.B.Schlegel, Int.J.Quantum Chem., Symp. 26, 253-264(1992)
T.H.Fischer and J.Almlof, J.Phys.Chem. 96, 9768-9774(1992)
J.Baker, J.Comput.Chem. 14, 1085-1100(1993)

Here we have compared the same coordinate types, using a guess hessian, or a computed hessian. The latter set of runs is a test of the coordinates only, as the initial

hessian information is identical. The results show clearly the superiority of the natural internals, which like the previous example, give an energy decrease on every step:

	HESS=GUESS	HESS=READ
Cartesians	65	41 steps
good Z-matrix	32	23
natural internals	24	13

A final example is phosphinoazasilatrane, with three rings fused on a common SiN bond, in which 112 steps in Cartesian space became 32 steps in natural internals. The moral is:

"A little brain time can save a lot of CPU time."

In late 1998, a new kind of internal coordinate method was included into PC GAMESS. This is the delocalized internal coordinate (DLC) of

J.Baker, A. Kessi, B.Delley
J.Chem.Phys. 105, 192-212(1996)

although as is the usual case, the implementation is not exactly the same. Bonds are kept as independent coordinates, while angles are placed in linear combination by the DLC process. There are some interesting options for applying constraints, and other options to assist the automatic DLC generation code by either adding or deleting coordinates. It is simple to use DLCs in their most basic form:

```
 $contrl nzvar=xx $end
 $zmat   dlc=.true. auto=.true. $end
```

Our initial experience is that the quality of DLCs is not as good as explicitly constructed natural internals, which benefit from human chemical knowledge, but are almost always better than carefully crafted $ZMATs using only the primitive internal coordinates (although we have seen a few exceptions). Once we have more numerical experience with the use of DLC's, we will come back and revise the above discussion of coordinate choices. In the meantime, they are quite simple to choose, so give them a go.

* * * The Role of Symmetry * * *

At the end of a succesful geometry search, you will have a set of coordinates where the gradient of the energy is zero. However your newly discovered stationary point is not necessarily a minimum or saddle point!

This apparent mystery is due to the fact that the gradient vector transforms under the totally symmetric representation of the molecular point group. As a direct consequence, a geometry search is point group conserving. (For a proof of these statements, see J.W.McIver and A.Komornicki, Chem.Phys.Lett., 10,303-306(1971)). In simpler terms, the molecule will remain in whatever point group you select in $DATA, even if the true minimum is in some lower point group. Since a geometry search only explores totally symmetric degrees of freedom, the only way to learn about the curvatures for all degrees of

freedom is RUNTYP=HESSIAN.

As an example, consider disilene, the silicon analog of ethene. It is natural to assume that this molecule is planar like ethene, and an OPTIMIZE run in D2h symmetry will readily locate a stationary point. However, as a calculation of the hessian will readily show, this structure is a transition state (one imaginary frequency), and the molecule is really trans-bent (C2h). A careful worker will always characterize a stationary point as either a minimum, a transition state, or some higher order stationary point (which is not of great interest!) by performing a RUNTYP=HESSIAN.

The point group conserving properties of a geometry search can be annoying, as in the preceeding example, or advantageous. For example, assume you wish to locate the transition state for rotation about the double bond in ethene. A little thought will soon reveal that ethene is D2h, the 90 degrees twisted structure is D2d, and structures in between are D2. Since the saddle point is actually higher symmetry than the rest of the rotational surface, you can locate it by RUNTYP=OPTIMIZE within D2d symmetry. You can readily find this stationary point with the diagonal guess hessian! In fact, if you attempt to do a RUNTYP=SADPOINT within D2d symmetry, there will be no totally symmetric modes with negative curvatures, and it is unlikely that the geometry search will be very well behaved.

Although a geometry search cannot lower the symmetry, the gain of symmetry is quite possible. For example, if you initiate a water molecule optimization with a trial structure which has unequal bond lengths, the geometry search will come to a structure that is indeed C2v (to within OPTTOL, anyway). However, PC GAMESS leaves it up to you to realize that a gain of symmetry has occurred.

In general, Mother Nature usually chooses more symmetrical structures over less symmetrical structures. Therefore you are probably better served to assume the higher symmetry, perform the geometry search, and then check the stationary point's curvatures. The alternative is to start with artificially lower symmetry and see if your system regains higher symmetry. The problem with this approach is that you don't necessarily know which subgroup is appropriate, and you lose the great speedups PC GAMESS can obtain from proper use of symmetry. It is good to note here that "lower symmetry" does not mean simply changing the name of the point group and entering more atoms in $DATA, instead the nuclear coordinates themselves must actually be of lower symmetry.

* * * Practical Matters * * *

Geometry searches do not bring the gradient exactly to zero. Instead they stop when the largest component of the gradient is below the value of OPTTOL, which defaults to a reasonable 0.0001. Analytic hessians usually have

residual frequencies below 10 cm**-1 with this degree of optimization. The sloppiest value you probably ever want to try is 0.0005.

If a geometry search runs out of time, or exceeds NSTEP, it can be restarted. For RUNTYP=OPTIMIZE, restart with the coordinates having the lowest total energy (do a string search on "FINAL"). For RUNTYP=SADPOINT, restart with the coordinates having the smallest gradient (do a string search on "RMS", which means root mean square). These are not necessarily at the last geometry!

The "restart" should actually be a normal run, that is you should not try to use the restart options in $CONTRL (which may not work anyway). A geometry search can be restarted by extracting the desired coordinates for $DATA from the printout, and by extracting the corresponding $GRAD group from the PUNCH file. If the $GRAD group is supplied, the program is able to save the time it would ordinarily take to compute the wavefunction and gradient at the initial point, which can be a substantial savings. There is no input to trigger reading of a $GRAD group: if found, it is read and used. Be careful that your $GRAD group actually corresponds to the coordinates in $DATA, as PC GAMESS has no check for this.

Sometimes when you are fairly close to the minimum, an OPTIMIZE run will take a first step which raises the energy, with subsequent steps improving the energy and perhaps finding the minimum. The erratic first step is caused by the GUESS hessian. It may help to limit the size of this wrong first step, by reducing its radius, DXMAX. Conversely, if you are far from the minimum, sometimes you can decrease the number of steps by increasing DXMAX.

When using internals, the program uses an iterative process to convert the internal coordinate change into Cartesian space. In some cases, a small change in the internals will produce a large change in Cartesians, and thus produce a warning message on the output. If these warnings appear only in the beginning, there is probably no problem, but if they persist you can probably devise a better set of coordinates. You may in fact have one of the two problems described in the next paragraph. In some cases (hopefully very few) the iterations to find the Cartesian displacement may not converge, producing a second kind of warning message. The fix for this may very well be a new set of internal coordinates as well, or adjustment of ITBMAT in $STATPT.

There are two examples of poorly behaved internal coordinates which can give serious problems. The first of these is three angles around a central atom, when this atom becomes planar (sum of the angles nears 360). The other is a dihedral where three of the atoms are nearly linear, causing the dihedral to flip between 0 and 180. Avoid these two situations if you want your geometry search to be convergent.

Sometimes it is handy to constrain the geometry search by freezing one or more coordinates, via the IFREEZ array. For example, constrained optimizations may be useful while trying to determine what area of a potential energy surface contains a saddle point. If you try to freeze coordinates with an automatically generated $ZMAT, you need to know that the order of the coordinates defined in $DATA is

```
y
y  x r1
y  x r2  x a3
y  x r4  x a5  x w6
y  x r7  x a8  x w9
```

and so on, where y and x are whatever atoms and molecular connectivity you happen to be using.

* * * Saddle Points * * *

Finding minima is relatively easy. There are large tables of bond lengths and angles, so guessing starting geometries is pretty straightforward. Very nasty cases may require computation of an exact hessian, but the location of most minima is straightforward.

In contrast, finding saddle points is a black art. The diagonal guess hessian will never work, so you must provide a computed one. The hessian should be computed at your best guess as to what the transition state (T.S.) should be. It is safer to do this in two steps as outlined above, rather than HESS=CALC. This lets you verify you have guessed a structure with one and only one negative curvature. Guessing a good trial structure is the hardest part of a RUNTYP=SADPOINT!

This point is worth iterating. Even with sophisticated step size control such as is offered by the QA/TRIM or RFO methods, it is in general very difficult to move correctly from a region with incorrect curvatures towards a saddle point. Even procedures such as CONOPT or RUNTYP=GRADEXTR will not replace your own chemical intuition about where saddle points may be located.

The RUNTYP=HESSIAN's normal coordinate analysis is rigorously valid only at stationary points on the surface. This means the frequencies from the hessian at your trial geometry are untrustworthy, in particular the six "zero" frequencies corresponding to translational and rotational (T&R) degrees of freedom will usually be 300-500 cm**-1, and possibly imaginary. The Sayvetz conditions on the printout will help you distinguish the T&R "contaminants" from the real vibrational modes. If you have defined a $ZMAT, the PURIFY option within $STATPT will help zap out these T&R contaminants).

If the hessian at your assumed geometry does not have one and only one imaginary frequency (taking into account that the "zero" frequencies can sometimes be 300i!), then it will probably be difficult to find the saddle point.

Instead you need to compute a hessian at a better guess for the initial geometry, or read about mode following below.

If you need to restart your run, do so with the coordinates which have the smallest RMS gradient. Note that the energy does not necessarily have to decrease in a SADPOINT run, in contrast to an OPTIMIZE run. It is often necessary to do several restarts, involving recomputation of the hessian, before actually locating the saddle point.

Assuming you do find the T.S., it is always a good idea to recompute the hessian at this structure. As described in the discussion of symmetry, only totally symmetric vibrational modes are probed in a geometry search. Thus it is fairly common to find that at your "T.S." there is a second imaginary frequency, which corresponds to a non-totally symmetric vibration. This means you haven't found the correct T.S., and are back to the drawing board. The proper procedure is to lower the point group symmetry by distorting along the symmetry breaking "extra" imaginary mode, by a reasonable amount. Don't be overly timid in the amount of distortion, or the next run will come back to the invalid structure.

The real trick here is to find a good guess for the transition structure. The closer you are, the better. It is often difficult to guess these structures. One way around this is to compute a linear least motion (LLM) path. This connects the reactant structure to the product structure by linearly varying each coordinate. If you generate about ten structures intermediate to reactants and products, and compute the energy at each point, you will in general find that the energy first goes up, and then down. The maximum energy structure is a "good" guess for the true T.S. structure. Actually, the success of this method depends on how curved the reaction path is.

A particularly good paper on the symmetry which a saddle point (and reaction path) can possess is by
P.Pechukas, J.Chem.Phys. 64, 1516-1521(1976)

* * * Mode Following * * *

In certain circumstances, METHOD=RFO and QA can walk from a region of all positive curvatures (i.e. near a minimum) to a transition state. The criteria for whether this will work is that the mode being followed should be only weakly coupled to other close-lying Hessian modes. Especially, the coupling to lower modes should be almost zero. In practise this means that the mode being followed should be the lowest of a given symmetry, or spatially far away from lower modes (for example, rotation of methyl groups at different ends of the molecule). It is certainly possible to follow also modes which do not obey these criteria, but the resulting walk (and possibly TS location) will be extremely sensitive to small details such as the stepsize.

This sensitivity also explain why TS searches often fail, even when starting in a region where the Hessian has the required one negative eigenvalue. If the TS mode is strongly coupled to other modes, the direction of the mode is incorrect, and the maximization of the energy along that direction is not really what you want (but what you get).

Mode following is really not a substitute for the ability to intuit regions of the PES with a single local negative curvature. When you start near a minimum, it matters a great deal which side of the minima you start from, as the direction of the search depends on the sign of the gradient. We strongly urge that you read before trying to use IFOLOW, namely the papers by Frank Jensen and Jon Baker mentioned above, and see also Figure 3 of C.J.Tsai, K.D.Jordan, J.Phys.Chem. 97, 11227-11237 (1993) which is quite illuminating on the sensitivity of mode following to the initial geometry point.

Note that PC GAMESS retains all degrees of freedom in its hessian, and thus there is no reason to suppose the lowest mode is totally symmetric. Remember to lower the symmetry in the input deck if you want to follow non-symmetric modes. You can get a printout of the modes in internal coordinate space by a EXETYP=CHECK run, which will help you decide on the value of IFOLOW.

* * *

CONOPT is a different sort of saddle point search procedure. Here a certain "CONstrained OPTimization" may be considered as another mode following method. The idea is to start from a minimum, and then perform a series of optimizations on hyperspheres of increasingly larger radii. The initial step is taken along one of the Hessian modes, chosen by IFOLOW, and the geometry is optimized subject to the constraint that the distance to the minimum is constant. The convergence criteria for the gradient norm perpendicular to the constraint is taken as 10*OPTTOL, and the corresponding steplength as 100*OPTTOL.

After such a hypersphere optimization has converged, a step is taken along the line connecting the two previous optimized points to get an estimate of the next hyper-sphere geometry. The stepsize is DXMAX, and the radius of hyperspheres is thus increased by an amount close (but not equal) to DXMAX. Once the pure NR step size falls below DXMAX/2 or 0.10 (whichever is the largest) the algorithm switches to a straight NR iterate to (hopefully) converge on the stationary point.

The current implementation always conducts the search in cartesian coordinates, but internal coordinates may be printed by the usual specification of NZVAR and ZMAT. At present there is no restart option programmed.

CONOPT is based on the following papers, but the actual

implementation is the modified equations presented in
Frank Jensen's paper mentioned above.
Y. Abashkin, N. Russo, J.Chem.Phys. 100, 4477-4483(1994).
Y. Abashkin, N. Russo, M. Toscano,
Int.J.Quant.Chem. 52, 695-704(1994).

There is little experience on how this method works in practice, experiment with it at your own risk!

IRC methods

The Intrinsic Reaction Coordinate (IRC) is defined as the minimum energy path connecting the reactants to products via the transition state. In practice, the IRC is found by first locating the transition state for the reaction. The IRC is then found in halves, going forward and backwards from the saddle point, down the steepest descent path in mass weighted Cartesian coordinates. This is accomplished by numerical integration of the IRC equations, by a variety of methods to be described below.

The IRC is becoming an important part of polyatomic dynamics research, as it is hoped that only knowledge of the PES in the vicinity of the IRC is needed for prediction of reaction rates, at least at threshhold energies. The IRC has a number of uses for electronic structure purposes as well. These include the proof that a certain transition structure does indeed connect a particular set of reactants and products, as the structure and imaginary frequency normal mode at the saddle point do not always unambiguously identify the reactants and products. The study of the electronic and geometric structure along the IRC is also of interest. For example, one can obtain localized orbitals along the path to determine when bonds break or form.

The accuracy to which the IRC is determined is dictated by the use one intends for it. Dynamical calculations require a very accurate determination of the path, as derivative information (second derivatives of the PES at various IRC points, and path curvature) is required later. Thus, a sophisticated integration method (such as AMPC4 or RK4), and small step sizes (STRIDE=0.05, 0.01, or even smaller) may be needed. In addition to this, care should be taken to locate the transition state carefully (perhaps decreasing OPTTOL by a factor of 10), and in the initiation of the IRC run. The latter might require a hessian matrix obtained by double differencing, certainly the hessian should be PURIFY'd. Note also that EVIB must be chosen carefully, as decribed below.

On the other hand, identification of reactants and products allows for much larger step sizes, and cruder integration methods. In this type of IRC one might want to be careful in leaving the saddle point (perhaps STRIDE should be reduced to 0.10 or 0.05 for the first few steps away from the transition state), but once a few points have been taken, larger step sizes can be employed. In general, the defaults in the $IRC group are set up for this latter,

cruder quality IRC. The STRIDE value for the GS2 method can usually be safely larger than for other methods, no matter what your interest in accuracy is.

The simplest method of determining an IRC is linear gradient following, PACE=LINEAR. This method is also known as Euler's method. If you are employing PACE=LINEAR, you can select "stabilization" of the reaction path by the Ishida, Morokuma, Komornicki method. This type of corrector has no apparent mathematical basis, but works rather well since the bisector usually intersects the reaction path at right angles (for small step sizes). The ELBOW variable allows for a method intermediate to LINEAR and stabilized LINEAR, in that the stabilization will be skipped if the gradients at the original IRC point, and at the result of a linear prediction step form an angle greater than ELBOW. Set ELBOW=180 to always perform the stabilization.

A closely related method is PACE=QUAD, which fits a quadratic polynomial to the gradient at the current and immediately previous IRC point to predict the next point. This pace has the same computational requirement as LINEAR, and is slightly more accurate due to the reuse of the old gradient. However, stabilization is not possible for this pace, thus a stabilized LINEAR path is usually more accurate than QUAD.

Two rather more sophisticated methods for integrating the IRC equations are the fourth order Adams-Moulton predictor-corrector (PACE=AMPC4) and fourth order Runge-Kutta (PACE=RK4). AMPC4 takes a step towards the next IRC point (prediction), and based on the gradient found at this point (in the near vincinity of the next IRC point) obtains a modified step to the desired IRC point (correction). AMPC4 uses variable step sizes, based on the input STRIDE. RK4 takes several steps part way toward the next IRC point, and uses the gradient at these points to predict the next IRC point. RK4 is the most accurate integration method implemented in PC GAMESS, and is also the most time consuming.

The Gonzalez-Schlegel 2nd order method finds the next IRC point by a constrained optimization on the surface of a hypersphere, centered at 1/2 STRIDE along the gradient vector leading from the previous IRC point. By construction, the reaction path between two successive IRC points is thus a circle tangent to the two gradient vectors. The algorithm is much more robust for large steps than the other methods, so it has been chosen as the default method. Thus, the default for STRIDE is too large for the other methods. The number of energy and gradients need to find the next point varies with the difficulty of the constrained optimization, but is normally not very many points. Be sure to provide the updated hessian from the previous run when restarting PACE=GS2.

The number of wavefunction evaluations, and energy gradients needed to jump from one point on the IRC to the next

point are summarized in the following table:

PACE	# energies	# gradients	
LINEAR stabilized	1	1	
LINEAR	3	2	
QUAD	1	1	(+ reuse of historical gradient)
AMPC4	2	2	(see note)
RK4	4	4	
GS2	2-4	2-4	(equal numbers)

Note that the AMPC4 method sometimes does more than one correction step, with each such corection adding one more energy and gradient to the calculation. You get what you pay for in IRC calculations: the more energies and gradients which are used, the more accurate the path found.

A description of these methods, as well as some others that were found to be not as good is geven by Kim Baldridge and Lisa Pederson, Pi Mu Epsilon Journal, 9, 513-521 (1993).

* * *

All methods are initiated by jumping from the saddle point, parallel to the normal mode (CMODE) which has an imaginary frequency. The jump taken is designed to lower the energy by an amount EVIB. The actual distance taken is thus a function of the imaginary frequency, as a smaller FREQ will produce a larger initial jump. You can simply provide a $HESS group instead of CMODE and FREQ, which involves less typing. To find out the actual step taken for a given EVIB, use EXETYP=CHECK. The direction of the jump (towards reactants or products) is governed by FORWRD. Note that if you have decided to use small step sizes, you must employ a smaller EVIB to ensure a small first step. The GS2 method begins by following the normal mode by one half of STRIDE, and then performing a hypersphere minimization about that point, so EVIB is irrelevant to this PACE.

The only method which proves that a properly converged IRC has been obtained is to regenerate the IRC with a smaller step size, and check that the IRC is unchanged. Again, note that the care with which an IRC must be obtained is highly dependent on what use it is intended for.

A small program which converts the IRC results punched to file IRCDATA into a format close to that required by the POLYRATE VTST dynamics program written in Don Truhlar's group is available. For a copy of this IRCED program, contact Mike Schmidt. The POLYRATE program must be obtained from the Truhlar group.

Some key IRC references are:

K.Ishida, K.Morokuma, A.Komornicki
J.Chem.Phys. 66, 2153-2156 (1977)
K.Muller

Angew.Chem., Int.Ed.Engl.19, 1-13 (1980)
M.W.Schmidt, M.S.Gordon, M.Dupuis
J.Am.Chem.Soc. 107, 2585-2589 (1985)
B.C.Garrett, M.J.Redmon, R.Steckler, D.G.Truhlar,
K.K.Baldridge, D.Bartol, M.W.Schmidt, M.S.Gordon
J.Phys.Chem. 92, 1476-1488(1988)
K.K.Baldridge, M.S.Gordon, R.Steckler, D.G.Truhlar
J.Phys.Chem. 93, 5107-5119(1989)
C.Gonzales, H.B.Schlegel
J.Chem.Phys. 90, 2154-2161(1989)

The IRC discussion closes with some practical tips:

The $IRC group has a confusing array of variables, but fortunately very little thought need be given to most of them. An IRC run is restarted by moving the coordinates of the next predicted IRC point into $DATA, and inserting the new $IRC group into your input file. You must select the desired value for NPOINT. Thus, only the first job which initiates the IRC requires much thought about $IRC.

The symmetry specified in the $DATA deck should be the symmetry of the reaction path. If a saddle point happens to have higher symmetry, use only the lower symmetry in the $DATA deck when initiating the IRC. The reaction path will have a lower symmetry than the saddle point whenever the normal mode with imaginary frequency is not totally symmetric. Be careful that the order and orientation of the atoms corresponds to that used in the run which generated the hessian matrix.

If you wish to follow an IRC for a different isotope, use the $MASS group. If you wish to follow the IRC in regular Cartesian coordinates, just enter unit masses for each atom. Note that CMODE and FREQ are a function of the atomic masses, so either regenerate FREQ and CMODE, or more simply, provide the correct $HESS group.

Gradient Extremals

This section of the manual, as well as the source code to trace gradient extremals was written by Frank Jensen of Odense University.

A Gradient Extremal (GE) curve consists of points where the gradient norm on a constant energy surface is stationary. This is equivalent to the condition that the gradient is an eigenvector of the Hessian. Such GE curves radiate along all normal modes from a stationary point, and the GE leaving along the lowest normal mode from a minimum is the gentlest ascent curve. This is not the same as the IRC curve connecting a minimum and a TS, but may in some cases be close.

GEs may be divided into three groups: those leading to dissociation, those leading to atoms colliding, and

those which connect stationary points. The latter class allows a determination of many (all?) stationary points on a PES by tracing out all the GEs. Following GEs is thus a semi-systematic way of mapping out stationary points. The disadvantages are:

i) There are many (but finitely many!) GEs for a large molecule.
ii) Following GEs is computationally expensive.
iii) There is no control over what type of stationary point (if any) a GE will lead to.

Normally one is only interested in minima and TSs, but many higher order saddle points will also be found. Furthermore, it appears that it is necessary to follow GEs radiating also from TSs and second (and possibly also higher) order saddle point to find all the TSs.

A rather complete map of the extremals for the H2CO potential surface is available in a paper which explains the points just raised in greater detail:

K.Bondensgaard, F.Jensen,
J.Chem.Phys. 104, 8025-8031(1996).

An earlier paper gives some of the properties of GEs:

D.K.Hoffman, R.S.Nord, K.Ruedenberg,
Theor. Chim. Acta 69, 265-279(1986).

There are two GE algorithms in PC GAMESS, one due to Sun and Ruedenberg (METHOD=SR), which has been extended to include the capability of locating bifurcation points and turning points, and another due to Jorgensen, Jensen, and Helgaker (METHOD=JJH):

J. Sun, K. Ruedenberg, J.Chem.Phys. 98, 9707-9714(1993)
P. Jorgensen, H. J. Aa. Jensen, T. Helgaker
Theor. Chim. Acta 73, 55 (1988).

The Sun and Ruedenberg method consist of a predictor step taken along the tangent to the GE curve, followed by one or more corrector steps to bring the geometry back to the GE. Construction of the GE tangent and the corrector step requires elements of the third derivative of the energy, which is obtained by a numerical differentiation of two Hessians. This puts some limitations on which systems the GE algorithm can be used for. First, the numerical differentiation of the Hessian to produce third derivatives means that the Hessian should be calculated by analytical methods, thus only those types of wavefunctions where this is possible can be used. Second, each predictor/corrector step requires at least two Hessians, but often more. Maybe 20-50 such steps are necessary for tracing a GE from one stationary point to the next. A systematic study of all the GE radiating from a stationary point increases the work by a factor of ~2*(3N-6). One should thus be prepared to invest at least hundreds, and more likely thousands, of Hessian calculations. In other words, small systems, small basis sets, and simple wave-functions.

The Jorgensen, Jensen, and Helgaker method consists of taking a step in the direction of the chosen Hessian

eigenvector, and then a pure NR step in the perpendicular modes. This requires (only) one Hessian calculation for each step. It is not suitable for following GEs where the GE tangent forms a large angle with the gradient, and it is incapable of locating GE bifurcations.

Although experience is limited at present, the JJH method does not appear to be suitable for following GEs in general (at least not in the current implementation). Experiment with it at your own risk!

The flow of the SR algorithm is as follows: A predictor geometry is produced, either by jumping away from a stationary point, or from a step in the tangent direction from the previous point on the GE. At the predictor geometry, we need the gradient, the Hessian, and the third derivative in the gradient direction. Depending on HSDFDB, this can be done in two ways. If .TRUE. the gradient is calculated, and two Hessians are calculated at SNUMH distance to each side in the gradient direction. The Hessian at the geometry is formed as the average of the two displaced Hessians. This corresponds to a double-sided differentiation, and is the numerical most stable method for getting the partial third derivative matrix. If HSDFDB = .FALSE., the gradient and Hessian are calculated at the current geometry, and one additional Hessian is calculated at SNUMH distance in the gradient direction. This corresponds to a single-sided differentiation. In both cases, two full Hessian calculations are

necessary, but HSDFDB = .TRUE. require one additional wavefunction and gradient calculation. This is usually a fairly small price compared to two Hessians, and the numerically better double-sided differentiation has therefore been made the default.

Once the gradient, Hessian, and third derivative is available, the corrector step and the new GE tangent are constructed. If the corrector step is below a threshold, a new predictor step is taken along the tangent vector. If the corrector step is larger than the threshold, the correction step is taken, and a new micro iteration is performed. DELCOR thus determines how closely the GE will be followed, and DPRED determine how closely the GE path will be sampled.

The construction of the GE tangent and corrector step involve solution of a set of linear equations, which in matrix notation can be written as Ax=B. The A-matrix is also the second derivative of the gradient norm on the constant energy surface.

After each corrector step, various things are printed to monitor the behavior: The projection of the gradient along the Hessian eigenvalues (the gradient is parallel to an eigenvector on the GE), the projection of the GE tangent along the Hessian eigenvectors, and the overlap of the Hessian eigenvectors with the mode being followed from the previous (optimzed) geometry. The sign of these

overlaps are not significant, they just refer to an arbitrary phase of the Hessian eigenvectors.

After the micro iterations has converged, the Hessian eigenvector curvatures are also displayed, this is an indication of the coupling between the normal modes. The number of negative eigenvalues in the A-matrix is denoted the GE index. If it changes, one of the eigenvalues must have passed through zero. Such points may either be GE bifurcations (where two GEs cross) or may just be "turning points", normally when the GE switches from going uphill in energy to downhill, or vice versa. The distinction is made based on the B-element corresponding to the A-matrix eigenvalue = 0. If the B-element = 0, it is a bifurcation, otherwise it is a turning point.

If the GE index changes, a linear interpolation is performed between the last two points to locate the point where the A-matrix is singular, and the corresponding B-element is determined. The linear interpolation points will in general be off the GE, and thus the evaluation of whether the B-element is 0 is not always easy. The program additionally evaluates the two limiting vectors which are solutions to the linear sets of equations, these are also used for testing whether the singular point is a bifurcation point or turning point.

Very close to a GE bifurcation, the corrector step become numerically unstable, but this is rarely a problem in practice. It is a priori expected that GE bifurcation will occur only in symmetric systems, and the crossing GE will break the symmetry. Equivalently, a crossing GE may be encountered when a symmetry element is formed, however such crossings are much harder to detect since the GE index does not change, as one of the A-matrix eigenvalues merely touches zero. The program prints an message if the absolute value of an A-matrix eigenvalue reaches a minimum near zero, as such points may indicate the passage of a bifurcation where a higher symmetry GE crosses. Run a movie of the geometries to see if a more symmetric structure is passed during the run.

An estimate of the possible crossing GE direction is made at all points where the A-matrix is singular, and two perturbed geometries in the + and - direction are written out. These may be used as predictor geometries for following a crossing GE. If the singular geometry is a turning point, the + and - geometries are just predictor geometries on the GE being followed.

In any case, a new predictor step can be taken to trace a different GE from the newly discovered singular point, using the direction determined by interpolation from the two end point tangents (the GE tangent cannot be uniquely determined at a bifurcation point). It is not possible to determine what the sign of IFOLOW should be when starting off along a crossing GE at a bifurcation, one will have to try a step to see if it returns to the bifurcation point or not.

In order to determine whether the GE index change it is necessary to keep track of the order of the A-matrix eigenvalues. The overlap between successive eigenvectors are shown as "Alpha mode overlaps".

Things to watch out for:

1) The numerical differentiation to get third derivatives requires more accuracy than usual. The SCF convergence should be at least 100 times smaller than SNUMH, and preferably better. With the default SNUMH of 10**(-4) the SCF convergence should be at least 10**(-6). Since the last few SCF cycles are inexpensive, it is a good idea to tighten the SCF convergence as much as possible, to maybe 10**(-8) or better. You may also want to increase the integral accuracy by reducing the cutoffs (ITOL and ICUT) and possibly also try more accurate integrals (INTTYP=HONDO). The CUTOFF in $TRNSFM may also be reduced to produce more accurate Hessians. Don't attempt to use a value for SNUMH below 10**(-6), as you simply can't get enough accuracy. Since experience is limited at present, it is recommended that some tests runs are made to learn the sensitivity of these factors for your system.

2) GEs can be followed in both directions, uphill or downhill. When stating from a stationary point, the direction is implicitly given as away from the stationary point. When starting from a non-stationary point, the "+" and "-" directions (as chosen by the sign of IFOLOW) refers to the gradient direction. The "+" direction is along the gradient (energy increases) and "-" is opposite to the gradient (energy decreases).

3) A switch from one GE to another may be seen when two GE come close together. This is especially troublesome near bifurcation points where two GEs actually cross. In such cases a switch to a GE with -higher- symmetry may occur without any indication that this has happened, except possibly that a very large GE curvature suddenly shows up. Avoid running the calculation with less symmetry than the system actually has, as this increases the likelihood that such switches occuring. Fix: alter DPRED to avoid having the predictor step close to the crossing GE.

4) "Off track" error message: The Hessian eigenvector which is parallel to the gradient is not the same as the one with the largest overlap to the previous Hessian mode. This usually indicate that a GE switch has occured (note that a switch may occur without this error message), or a wrong value for IFOLOW when starting from a non-stationary point. Fix: check IFOLOW, if it is correct then reduce DPRED, and possibly also DELCOR.

5) Low overlaps of A-matrix eigenvectors. Small overlaps may give wrong assignment, and wrong conclusions about GE index change. Fix: reduce DPRED.

6) The interpolation for locating a point where one of the A-matrix eigenvalues is zero fail to converge. Fix: reduce DPRED (and possibly also DELCOR) to get a shorther (and better) interpolation line.

7) The GE index changes by more than 1. A GE switch may have occured, or more than one GE index change is located between the last and current point. Fix: reduce DPRED to sample the GE path more closely.

8) If SNRMAX is too large the algorithm may try to locate stationary points which are not actually on the GE being followed. Since GEs often pass quite near a stationary point, SNRMAX should only be increased above the default 0.10 after some consideration.

Continuum solvation methods

In a very thorough 1994 review of continuum solvation models, Tomasi and Persico divide the possible approaches to the treatment of solvent effects into four categories:

a) virial equations of state, correlation functions
b) Monte Carlo or molecular dynamics simulations
c) continuum treatments
d) molecular treatments

The Effective Fragment Potential method, documented in the following section of this chapter, falls into the latter category, as each EFP solvent molecule is modeled as a distinct object. This section describes the two continuum models which are implemented in the standard version of PC GAMESS.

Continuum models typically form a cavity of some sort containing the solute molecule, while the solvent outside the cavity is thought of as a continuous medium and is categorized by a limited amount of physical data, such as the dielectric constant. The electric field of the charged particles comprising the solute interact with this background medium, producing a polarization in it, which in turn feeds back upon the solute's wavefunction.

* * *

A simple continuum model is the Onsager cavity model, often called the Self-Consistent Reaction Field, or SCRF model. This represents the charge distribution of the solute in terms of a multipole expansion. SCRF usually uses an idealized cavity (spherical or ellipsoidal) to allow an analytic solution to the interaction energy between the solute multipole and the multipole which this induces in the continuum. This method is implemented in PC GAMESS in the simplest possible fashion:

i) a spherical cavity is used
ii) the molecular electrostatic potential of the solute is represented as a dipole only, except a monopole is also included for an ionic solute.

The input for this implementation of the Kirkwood-Onsager model is provided in $SCRF.

Some references on the SCRF method are

1. J.G.Kirkwood J.Chem.Phys. 2, 351 (1934)
2. L.Onsager J.Am.Chem.Soc. 58, 1486 (1936)
3. O.Tapia, O.Goscinski Mol.Phys. 29, 1653 (1975)
4. M.M.Karelson, A.R.Katritzky, M.C.Zerner
   Int.J.Quantum Chem., Symp. 20, 521-527 (1986)
5. K.V.Mikkelsen, H.Agren, H.J.Aa.Jensen, T.Helgaker
   J.Chem.Phys. 89, 3086-3095 (1988)
6. M.W.Wong, M.J.Frisch, K.B.Wiberg
   J.Am.Chem.Soc. 113, 4776-4782 (1991)
7. M.Szafran, M.M.Karelson, A.R.Katritzky, J.Koput,
   M.C.Zerner J.Comput.Chem. 14, 371-377 (1993)
8. M.Karelson, T.Tamm, M.C.Zerner
   J.Phys.Chem. 97, 11901-11907 (1993)

The method is very sensitive to the choice of the solute RADIUS, but not very sensitive to the particular DIELEC of polar solvents. The plots in reference 7 illustrate these points very nicely. The SCRF implementation in PC GAMESS is Zerner's Method A, described in the same reference. The total solute energy includes the Born term, if the solute is an ion. Another limitation is that a solute's electrostatic potential is not likely to be fit well as a dipole moment only, for example see Table VI of reference 5 which illustrates the importance of higher multipoles. Finally, the restriction to a spherical cavity may not be very representative of the solute's true shape. However, in the special case of a roundish molecule, and a large dipole which is geometry sensitive, the SCRF model may include sufficient physics to be meaningful:

M.W.Schmidt, T.L.Windus, M.S.Gordon
J.Am.Chem.Soc. 117, 7480-7486(1995).

* * *

A much more sophisticated continuum method, named the Polarizable Continuum Model, is also available. The PCM method places a solute in a cavity formed by a union of spheres centered on each atom. PCM also includes a more exact treatment of the electrostatic interaction with the surrounding medium, as the electrostatic potential of the solute generates an 'apparent surface charge' on the cavity's surface. The computational procedure divides this surface into small tesserae, on which the charge (and contributions to the gradient) are evaluated. Typically the spheres defining the cavity are taken to be 1.2 times the van der Waals radii. A technical difficulty caused by the penetration of the solute charge density outside this cavity is dealt with by a renormalization. The solvent is characterized by its dielectric constant, surface tension, size, density, and so on. Procedures are provided not only for the computation of the electrostatic interaction of the solute with the apparent surface charges, but also for the cavitation energy, and dispersion and repulsion contributions to the solvation free energy.

The main input group is $PCM, with $PCMCAV providing auxiliary cavity information. If any of the optional energy computations are requested in $PCM, the additional

input groups $NEWCAV, $DISBS, or $DISREP may be required.

Solvation of course affects the non-linear optical properties of molecules. The PCM implementation extends RUNTYP=TDHF to include solvent effects. Both static and frequency dependent hyperpolarizabilities can be found. Besides the standard PCM electrostatic contribution, the IREP and IDP keywords can be used to determine the effects of repulsion and dispersion on the polarizabilities.

Due to its sophistication, users of the PCM model are strongly encouraged to read the primary literature:

General papers on the PCM method:
1) S.Miertus, E.Scrocco, J.Tomasi
Chem.Phys. 55, 117-129(1981)
2) J.Tomasi, M.Persico Chem.Rev. 94, 2027-2094(1994)
3) R.Cammi, J.Tomasi J.Comput.Chem. 16, 1449-1458(1995)

The GEPOL method for cavity construction:
4) J.L.Pascual-Ahuir, E.Silla, J.Tomasi, R.Bonaccorsi
J.Comput.Chem. 8, 778-787(1987)

Charge renormalization (see also ref. 3):
5) B.Mennucci, J.Tomasi J.Chem.Phys. 106, 5151-5158(1997)

Derivatives with respect to nuclear coordinates:
(energy gradient and hessian)
6) R.Cammi, J.Tomasi J.Chem.Phys. 100, 7495-7502(1994)
7) R.Cammi, J.Tomasi J.Chem.Phys. 101, 3888-3897(1995)
8) M.Cossi, B.Mennucci, R.Cammi
J.Comput.Chem. 17, 57-73(1996)

Derivatives with respect to applied electric fields:
(polarizabilities and hyperpolarizabilities)
9) R.Cammi, J.Tomasi
Int.J.Quantum Chem. Symp. 29, 465-474(1995)
10) R.Cammi, M.Cossi, J.Tomasi
J.Chem.Phys. 104, 4611-4620(1996)
11) R.Cammi, M.Cossi, B.Mennucci, J.Tomasi
J.Chem.Phys. 105, 10556-10564(1996)
12) B. Mennucci, C. Amovilli, J. Tomasi
J.Chem.Phys. submitted.

Cavitation energy:
13) R.A.Pierotti Chem.Rev. 76, 717-726(1976)
14) J.Langlet, P.Claverie, J.Caillet, A.Pullman
J.Phys.Chem. 92, 1617-1631(1988)

Dispersion and repulsion energies:
15) F.Floris, J.Tomasi J.Comput.Chem. 10, 616-627(1989)
16) C.Amovilli, B.Mennucci
J.Phys.Chem. B101, 1051-1057(1997)

At the present time, the PCM model in PC GAMESS has the following limitations:

a) SCFTYP=RHF and MCSCF, only.
b) point group symmetry is switched off internally during PCM.
c) The PCM model does not run in parallel.
d) electric field integrals at normals to the surface elements are stored on disk, even during DIRSCF runs. The file size may be considerable.
e) nuclear derivatives are limited to gradients, although theory for hessians is given in Ref. 7.

The calculation shown on the next page illustrates the use of most PCM options. Since methane is non-polar, its internal energy change and the direct PCM electrostatic interaction is smaller than the cavitation, repulsion, and dispersion corrections. Note that the use of ICAV, IREP, and IDP are currently incompatible with gradients, so a reasonable calculation sequence might be to perform the geometry optimization with PCM electrostatics turned on, then perform an additional calculation to include the other solvent effects, adding extra functions to improve the dispersion correction.

```
!  calculation of CH4 (metano) in PCM water.
!  This input reproduces the data in Table 2, line 6, of
!  C.Amovilli, B.Mennucci J.Phys.Chem. B101, 1051-7(1997)
!
!  The gas phase FINAL energy is  -40.2075980280
!  The FINAL energy in PCM water= -40.2143590161
!                                                    (lit.)
!  FREE ENERGY IN SOLVENT      = -25234.89 KCAL/MOL
!  INTERNAL ENERGY IN SOLVENT  = -25230.64 KCAL/MOL
!  DELTA INTERNAL ENERGY       =       .01 KCAL/MOL ( 0.0)
!  ELECTROSTATIC INTERACTION   =      -.22 KCAL/MOL (-0.2)
!  PIEROTTI CAVITATION ENERGY  =      5.98 KCAL/MOL ( 6.0)
!  DISPERSION FREE ENERGY      =     -6.00 KCAL/MOL (-6.0)
!  REPULSION FREE ENERGY       =      1.98 KCAL/MOL ( 2.0)
!  TOTAL INTERACTION           =      1.73 KCAL/MOL ( 1.8)
!  TOTAL FREE ENERGY IN SOLVENT= -25228.91 KCAL/MOL
!
 $contrl scftyp=rhf runtyp=energy $end
 $guess  guess=huckel $end
 $system memory=300000 $end
!    the W1 basis input here exactly matches HONDO's DZP
 $DATA
CH4...gas phase geometry...in PCM water
Td

Carbon      6.
   DZV
   D 1 ; 1 0.75 1.0

Hydrogen    1.  0.6258579976  0.6258579976  0.6258579976
   DZV 0 1.20 1.15  ! inner and outer scale factors
   P 1 ; 1 1.00 1.0

 $END
!    reference cited used value for H2O's solvent radius
!    which differs from the built in constants.
 $PCM    ICOMP=2 IREP=1 IDP=1 ICAV=1
```

```
         SOLVNT=WATER RSOLV=1.35 $END
 $NEWCAV IPTYPE=2 ITSNUM=540 $END
!    dispersion W2 basis uses exponents which are
!    1/3 of smallest exponent in W1 basis of $DATA.
 $DISBS  NADD=11 NKTYP(1)=0,1,2, 0,1, 0,1, 0,1, 0,1
         XYZE(1)=0.0,0.0,0.0, 0.0511
                 0.0,0.0,0.0, 0.0382
                 0.0,0.0,0.0, 0.25
         1.1817023, 1.1817023, 1.1817023,  0.05435467
         1.1817023, 1.1817023, 1.1817023,  0.33333333
        -1.1817023, 1.1817023,-1.1817023,  0.05435467
        -1.1817023, 1.1817023,-1.1817023,  0.33333333
         1.1817023,-1.1817023,-1.1817023,  0.05435467
         1.1817023,-1.1817023,-1.1817023,  0.33333333
        -1.1817023,-1.1817023, 1.1817023,  0.05435467
        -1.1817023,-1.1817023, 1.1817023,  0.33333333 $end
```

The Effective Fragment Potential Method

The basic idea behind the effective fragment potential (EFP) method is to replace the chemically inert part of a system by EFPs, while performing a regular ab initio calculation on the chemically active part. Here "inert" means that no covalent bond breaking process occurs. This "spectator region" consists of one or more "fragments", which interact with the ab initio "active region" through non-bonded interactions, and so of course these EFP interactions affect the ab initio wavefunction. A simple example of an active region might be a solute molecule, with a surrounding spectator region of solvent molecules represented by fragments. Each discrete solvent molecule is represented by a single fragment potential, in marked contrast to continuum models for solvation.

The quantum mechanical part of the system is entered in the $DATA group, along with an appropriate basis. The EFPs defining the fragments are input by means of a $EFRAG group, one or more $FRAGNAME groups describing each fragment's EFP, and a $FRGRPL group. These groups define non-bonded interactions between the ab initio system and the fragments, and between the fragments. The former interactions enter via one-electron operators in the ab initio Hamiltonian, while the latter interactions are treated by analytic functions. The only electrons explicitly treated (e.g. with basis functions used to expand occupied orbitals) are those in the active region, so there are no new two electron terms. Thus the use of EFPs leads to significant time savings compared to full ab initio calculations on the same system.

At ISU, the EFPs are currently used to model RHF/DZP water molecules in order to study aqueous solvation effects, for example references 1,2,3. Our co-workers at NIST have also used EFPs to model parts of enzymes, see reference 4.

*** Terms in an EFP ***

The non-bonded interactions currently implemented are:

1) Coulomb interaction. The charge distribution of the fragments is represented by an arbitrary number of charges, dipoles, quadrupoles, and octupoles, which interact with the ab initio hamiltonian as well as with multipoles on other fragments. It is possible to input a screening term that accounts for the charge penetration. Typically the multipole expansion points are located on atomic nuclei and at bond midpoints.

2) Dipole polarizability. An arbitrary number of dipole polarizability tensors can be used to calculate the induced dipole on a fragment due to the electric field of the ab initio system as well as all the other fragments. These induced dipoles interact with the ab initio system as well as the other EFPs, in turn changing their electric fields. All induced dipoles are therefore iterated to self-consistency. Typically the polarizability tensors are located at the centroid of charge of each localized orbital of a fragment.

3) Repulsive potential. Two different forms for the repulsive potentials are used: one for ab initio-EFP repulsion and one for EFP-EFP repulsion. The form of the potentials is empirical, and consists of distributed gaussian or exponential functions, respectively. The primary contribution to the repulsion is the quantum mechanical exchange repulsion, but the fitting technique used to develop this term also includes the effects of charge transfer. Typically these fitted potentials are located on atomic nuclei within the fragment.

*** Constructing an EFP Using PC GAMESS ***

RUNTYP=MOROKUMA assists in the decomposition of inter-molecular interaction energies into electrostatic, polarization, charge transfer, and exchange repulsion contributions. This is very useful in developing EFPs since potential problems can be attributed to a particular term by comparison to these energy components for a particular system.

A molecular multipole expansion can be obtained using $ELMOM. A distributed multipole expansion can be obtained by either a Mulliken-like partitioning of the density (using $STONE) or by using localized molecular orbitals ($LOCAL: DIPDCM and QADDCM). The molecular dipole polarizability tensor can be obtained during a Hessian run ($CPHF), and a distributed LMO polarizability expression is also available ($LOCAL: POLDCM).

The repulsive potential is derived by fitting the difference between ab initio computed intermolecular interaction energies, and the form used for Coulomb and polarizability interactions. This difference is obtained at a large number of different interaction geometries, and is then fitted. Thus, the repulsive term is implicitly a function of the choices made in representing the Coulomb and polarizability terms. Note that PC GAMESS currently does not provide a way to obtain these repulsive potential, or

the charge interpenetration screening parameters.

Since you cannot develop all terms necessary to define a new EFP's $FRAGNAME group using PC GAMESS, in practice you will be limited to using the internally stored H2OEF2 potential mentioned below.

*** Current Limitations ***

1. The energy and energy gradient are programmed, which permits RUNTYP=ENERGY, GRADIENT, and numerical HESSIAN. The necessary modifications to use the EFP gradients while moving on the potential surface are programmed for RUNTYP=OPTIMIZE, SADPOINT, and IRC (see reference 3), but the other gradient based potential surface explorations such as DRC are not yet available. Finally, RUNTYP=PROP is also permissible.

2. The ab initio system must be treated with RHF, ROHF, UHF, or the open shell SCF wavefunctions permitted by the GVB code. The correlated methods in PC GAMESS (MP2 and CI) should not be used, since the available H2OEF2 potential was derived at the RHF level, and therefore does not contain dispersion terms. A correlated computation on the ab initio system without these terms in the EFP will probably lead to unphysical results. MCSCF and GVB-PP represent a gray area, but Mo Krauss has obtained some results for a solute described by an MCSCF wavefunction in which the EFP results agree well with fully ab initio computations (in structures and interaction energies).

3. EFPs can move relative to the ab initio system and relative to each other, but the internal structure of an EFP is frozen.

4. The boundary between the ab initio system and the EFPs must not be placed across a chemical bond.

5. Calculations must be done in C1 symmetry at present. Enter NOSYM=1 in $CONTRL to accomplish this.

6. Reorientation of the fragments and ab initio system is not well coordinated. If you are giving Cartesian coordinates for the fragments (COORD=CART in $EFRAG), be sure to use $CONTRL's COORD=UNIQUE option so that the ab initio molecule is not reoriented.

7. If you need IR intensities, you have to use NVIB=2. The potential surface is usually very soft for EFP motions, and double differenced Hessians should usually be obtained.

*** Practical hints for using EFPs ***

At the present time, we have only one EFP suitable for general use. This EFP models water, and its numerical parameters are internally stored, using the fragment name

H2OEF2. These numerical parameters are improved values over the H2OEF1 set which were presented and used in reference 2, and they also include the improved EFP-EFP repulsive term defined in reference 3. The H2OEF2 water EFP was derived from RHF/DH(d,p) computations on the water dimer system. When you use it, therefore, the ab initio part of your system should be treated at the SCF level, using a basis set of the same quality (ideally DH(d,p), but probably other DZP sets such as 6-31G(d,p) will give good results as well). Use of better basis sets than DZP with this water EFP has not been tested.

As noted, effective fragments have frozen internal geometries, and therefore only translate and rotate with respect to the ab initio region. An EFP's frozen coordinates are positioned to the desired location(s) in $EFRAG as follows:

a) the corresponding points are found in $FRAGNAME.
b) Point -1- in $EFRAG and its FRAGNAME equivalent are made to coincide.
c) The vector connecting -1- and -2- is aligned with the corresponding vector connecting FRAGNAME points.
d) The plane defined by -1-, -2-, and -3- is made to coincide with the corresponding FRAGNAME plane.

Therefore the 3 points in $EFRAG define only the relative position of the EFP, and not its internal structure. So, if the "internal structure" given by points in $EFRAG differs from the true values in $FRAGNAME, then the order in which the points are given in $EFRAG can affect the positioning of the fragment. It may be easier to input water EFPs if you use the Z-matrix style to define them, because then you can ensure you use the actual frozen geometry in your $EFRAG. Note that the H2OEF2 EFP uses the frozen geometry r(OH)=0.9438636, a(HOH)=106.70327, and the names of its 3 fragment points are ZO1, ZH2, ZH3.

The translations and rotations of EFPs with respect to the ab initio system and one another are automatically quite soft degrees of freedom. After all, the EFP model is meant to handle weak interactions! Therefore the satisfactory location of structures on these flat surfaces will require use of a tight convergence on the gradient: OPTTOL=0.00001 in the $STATPT group.

*** References ***

The first of these is more descriptive, and the second has a very detailed derivation of the method. The latest EFP developments are discussed in the 3rd paper.

1. "Effective fragment method for modeling intermolecular hydrogen bonding effects on quantum mechanical calculations"
   J.H.Jensen, P.N.Day, M.S.Gordon, H.Basch, D.Cohen, D.R.Garmer, M.Krauss, W.J.Stevens in "Modeling the Hydrogen Bond" (D.A. Smith, ed.) ACS Symposium Series 569, 1994, pp 139-151.
2. "An effective fragment method for modeling solvent effects in quantum mechanical calculations".

P.N.Day, J.H.Jensen, M.S.Gordon, S.P.Webb,
W.J.Stevens, M.Krauss, D.Garmer, H.Basch, D.Cohen
J.Chem.Phys. 105, 1968-1986(1996).
3. "The effective fragment model for solvation: internal
rotation in formamide"
W.Chen, M.S.Gordon, J.Chem.Phys., 105, 11081-90(1996)
4. "Transphosphorylation catalyzed by ribonuclease A:
Computational study using ab initio EFPs"
B.D.Wladkowski, M. Krauss, W.J.Stevens,
J.Am.Chem.Soc. 117, 10537-10545(1995).
5. "A study of aqueous glutamic acid using the effective
fragment potential model"
P.N.Day, R.Pachter J.Chem.Phys. 107, 2990-9(1997)
6. "Solvation and the excited states of formamide"
M.Krauss, S.P.Webb J.Chem.Phys. 107, 5771-5(1997)
7. "Study of small water clusters using the effective
fragment potential method"
G.N.Merrill, M.S.Gordon J.Phys.Chem.A 102, 2650-7(1998)
8. "Menshutkin Reaction"
S.P.Webb, M.S.Gordon J.Phys.Chem.A in press
9. "Solvation of Sodium Chloride: EFP study of NaCl(H2O)n"
C.P.Petersen, M.S.Gordon J.Phys.Chem.A 103, 4162-6(1999)

MOPAC calculations within PC GAMESS

Parts of MOPAC 6.0 have been included in PC GAMESS so that the PC GAMESS user has access to three semiempirical wavefunctions: MNDO, AM1 and PM3. These wavefunctions are quantum mechanical in nature but neglect most two electron integrals, a deficiency that is (hopefully) compensated for by introduction of empirical parameters. The quantum mechanical nature of semiempirical theory makes it quite compatible with the ab initio methodology in PC GAMESS. As a result, very little of MOPAC 6.0 actually is incorporated into PC GAMESS. The part that did make it in is the code that evaluates

1) the one- and two-electron integrals,
2) the two-electron part of the Fock matrix,
3) the cartesian energy derivatives, and
4) the ZDO atomic charges and molecular dipole.

Everything else is actually PC GAMESS: coordinate input (including point group symmetry), the SCF convergence procedures, the matrix diagonalizer, the geometry searcher, the numerical hessian driver, and so on. Most of the output will look like an ab initio output.

It is extremely simple to perform one of these calculations. All you need to do is specify GBASIS=MNDO, AM1, RM1, or PM3 in the $BASIS group. Note that this not only picks a particular Slater orbital basis, it also selects a particular "hamiltonian", namely a particular parameter set.

MNDO, AM1, RM1, and PM3 will not work with every option in PC GAMESS. Currently the semiempirical wavefunctions support SCFTYP=RHF, UHF, and ROHF in any combination with

RUNTYP=ENERGY, GRADIENT, OPTIMIZE, SADPOINT, HESSIAN, and IRC. Note that all hessian runs are numerical finite differencing. The MOPAC CI and half electron methods are not supported.

Because the majority of the implementation is PC GAMESS rather than MOPAC you will notice a few improvments. Dynamic memory allocation is used, so that PC GAMESS uses far less memory for a given size of molecule. The starting orbitals for SCF calculations are generated by a Huckel initial guess routine. Spin restricted (high spin) ROHF can be performed. Converged SCF orbitals will be labeled by their symmetry type. Numerical hessians will make use of point group symmetry, so that only the symmetry unique atoms need to be displaced. Infrared intensities will be calculated at the end of hessian runs.
Orbitals and hessians are punched out for convenient reuse in subsequent calculations.

MOPAC parameters exist for the following elements. The quote means that these elements are treated as "sparkles" rather than as atoms with genuine basis functions. For MNDO:

```
 H
Li  *          B  C  N  O  F
Na' *         Al Si  P  S Cl
 K' * ...  Zn  * Ge  *  * Br
Rb' * ...   *  * Sn  *  *  I
*   * ...  Hg  * Pb  *
```

For AM1:

```
 H
 *  *          B  C  N  O  F
Na' *         Al Si  P  S Cl
 K' * ...  Zn  * Ge  *  * Br
Rb' * ...   *  * Sn  *  *  I
*   * ...  Hg  *  *  *
```

For PM3:

```
 H
 *  Be         *  C  N  O  F
Na' Mg        Al Si  P  S Cl
 K' * ...  Zn Ga Ge As Se Br
Rb' * ...  Cd In Sn Sb Te  I
*   * ...  Hg Tl Pb Bi
```

Semiempirical calculations are very fast. One of the motives for the MOPAC implementation within PC GAMESS is to take advantage of this speed. Semiempirical models can rapidly provide reasonable starting geometries for ab initio optimizations. Semiempirical hessian matrices are obtained at virtually no computational cost, and may help dramatically with an ab initio geometry optimization. Simply use HESS=READ in $STATPT to use a MOPAC $HESS group in an ab initio run.

It is important to exercise caution as semiempirical methods can be dead wrong! The reasons for this are bad parameters (in certain chemical situations), and the underlying minimal basis set. A good question to ask before using MNDO, AM1 or PM3 is "how well is my system modeled with an ab initio minimal basis sets, such as STO-3G?" If the answer is "not very well" there is a good chance that a semiempirical description is equally poor.

--------- ----------

These two papers are of general interest:
A.D.Buckingham, J.Chem.Phys. 30, 1580-1585(1959).
D.Neumann, J.W.Moskowitz J.Chem.Phys. 49, 2056-2070(1968).

All units are derived from the atomic units for distance and the monopole electric charge, as given below.

distance - 1 au = 5.291771E-09 cm

monopole - 1 au = 4.803242E-10 esu
1 esu = sqrt(g-cm**3)/sec

dipole - 1 au = 2.541766E-18 esu-cm
1 Debye = 1.0E-18 esu-cm

quadrupole - 1 au = 1.345044E-26 esu-cm**2
1 Buckingham = 1.0E-26 esu-cm**2

octupole - 1 au = 7.117668E-35 esu-cm**3

electric potential - 1 au = 9.076814E-02 esu/cm

electric field - 1 au = 1.715270E+07 esu/cm**2
1 esu/cm**2 = 1 dyne/esu

electric field gradient- 1 au = 3.241390E+15 esu/cm**3

The atomic unit for electron density is electron/bohr**3 for the total density, and 1/bohr**3 for an orbital density.

The atomic unit for spin density is excess alpha spins per unit volume, h/4*pi*bohr**3. Only the expectation value is computed, with no constants premultiplying it.

IR intensities are printed in Debye**2/amu-Angstrom**2. These can be converted into intensities as defined by Wilson, Decius, and Cross's equation 7.9.25, in km/mole, by multiplying by 42.255. If you prefer 1/atm-cm**2, use a conversion factor of 171.65 instead. A good reference for deciphering these units is A.Komornicki, R.L.Jaffe J.Chem.Phys. 1979, 71, 2150-2155. A reference showing how IR intensities change with basis improvements at the HF level is Y.Yamaguchi, M.Frisch, J.Gaw, H.F.Schaefer, J.S.Binkley, J.Chem.Phys. 1986, 84, 2262-2278.

Localization tips

Three different orbital localization methods are implemented in PC GAMESS. The energy and dipole based methods normally produce similar results, but see M.W.Schmidt, S.Yabushita, M.S.Gordon in J.Chem.Phys., 1984, 88, 382-389 for an interesting exception. You can find references to the three methods at the beginning of this chapter.

The method due to Edmiston and Ruedenberg works by maximizing the sum of the orbitals' two electron self repulsion integrals. Most people who think about the different localization criteria end up concluding that this one seems superior. The method requires the two electron integrals, transformed into the molecular orbital basis. Because only the integrals involving the orbitals to be localized are needed, the integral transformation is actually not very time consuming.

The Boys method maximizes the sum of the distances between the orbital centroids, that is the difference in the orbital dipole moments.

The population method due to Pipek and Mezey maximizes a certain sum of gross atomic Mulliken populations. This procedure will not mix sigma and pi bonds, so you will not get localized banana bonds. Hence it is rather easy to find cases where this method give different results than the Ruedenberg or Boys approach.

PC GAMESS will localize orbitals for any kind of RHF, UHF, ROHF, or MCSCF wavefunctions. The localizations will automatically restrict any rotation that would cause the energy of the wavefunction to be changed (the total wavefunction is left invariant). As discussed below, localizations for GVB or CI functions are not permitted.

The default is to freeze core orbitals. The localized valence orbitals are scarcely changed if the core orbitals are included, and it is usually convenient to leave them out. Therefore, the default localizations are: RHF functions localize all doubly occupied valence orbitals. UHF functions localize all valence alpha, and then all valence beta orbitals. ROHF functions localize all valence doubly occupied orbitals, and all singly occupied orbitals, but do not mix these two orbital spaces. MCSCF functions localize all valence MCC type orbitals, and localize all active orbitals, but do not mix these two orbital spaces. To recover the invariant MCSCF function, you must be using a FORS=.TRUE. wavefunction, and you must set GROUP=C1 in $DRT, since the localized orbitals possess no symmetry.

In general, GVB functions are invariant only to localizations of the NCO doubly occupied orbitals. Any pairs must be written in natural form, so pair orbitals cannot be localized. The open shells may be degenerate, so in general these should not be mixed. If for some reason you feel you must localize the doubly occupied space, do a RUNTYP=PROP job. Feed in the GVB orbitals, but tell the program it is SCFTYP=RHF, and enter a negative ICHARG so that PC GAMESS thinks all orbitals occupied in the GVB are occupied in this fictitous RHF. Use NINA or NOUTA to localize the desired doubly occupied orbitals. Orbital localization is not permitted for CI, because we cannot imagine why you would want to do that anyway.

Boys localization of the core orbitals in molecules having elements from the third or higher row almost never

succeeds. Boys localization including the core for second row atoms will often work, since there is only one inner shell on these. The Ruedenberg method should work for any element, although including core orbitals in the integral transformation is more expensive.

The easiest way to do localization is in the run which generates the wavefunction, by selecting LOCAL=xxx in the $CONTRL group. However, localization may be conveniently done at any time after determination of the wavefunction, by executing a RUNTYP=PROP job. This will require only $CONTRL, $BASIS/$DATA, $GUESS (pick MOREAD), the converged $VEC, possibly $SCF or $DRT to define your wavefunction, and optionally some $LOCAL input.

There is an option to restrict all rotations that would mix orbitals of different symmetries. SYMLOC=.TRUE. yields only partially localized orbitals, but these still possess symmetry. They are therefore very useful as starting orbitals for MCSCF or GVB-PP calculations. Because they still have symmetry, these partially localized orbitals run as efficiently as the canonical orbitals. Because it is much easier for a user to pick out the bonds which are to be correlated, a significant number of iterations can be saved, and convergence to false solutions is less likely.

* * *

The most important reason for localizing orbitals is to analyze the wavefunction. A simple example is to make contour plots of the resulting orbitals with the PLTORB graphics codes, or perhaps to read the localized orbitals in during a RUNTYP=PROP job to examine their Mulliken populations. MCSCF localized orbitals can be input to a CI calculation, to generate the corresponding density matrix, which contains much useful information about electron populations and chemical bonding. For example,

J.Am.Chem.Soc., 104, 960-967 (1982),
J.Am.Chem.Soc., 113, 5231-5243 (1991),
Theoret.Chim.Acta, 83, 57-68 (1992).

In addition, the energy of your molecule can be partitioned over the localized orbitals so that you may be able to understand the origin of barriers, etc. This analysis can be made for the SCF energy, and also the MP2 correction to the SCF energy, which requires two separate runs. An explanation of the method, and application to hydrogen bonding may be found in J.H.Jensen, M.S.Gordon, J.Phys.Chem. 1995, 99, 8091-8107.

Analysis of the SCF energy is based on the localized charge distribution (LCD) model: W.England and M.S.Gordon, J.Am.Chem.Soc. 93, 4649-4657 (1971). This is implemented for RHF and ROHF wavefunctions, and it requires use of the Ruedenberg localization method, since it needs the two electron integrals to correctly compute energy sums. All orbitals must be included in the localization, even the cores, so that the total energy is reproduced.

The LCD requires both electronic and nuclear charges to be partitioned. The orbital localization automatically accomplishes the former, but division of the nuclear charge may require some assistance from you. The program attempts to correctly partition the nuclear charge, if you select the MOIDON option, according to the following: a Mulliken type analysis of the localized orbitals is made. This determines if an orbital is a core, lone pair, or bonding MO. Two protons are assigned to the nucleus to which any core or lone pair belongs. One proton is assigned to each of the two nuclei in a bond. When all localized orbitals have been assigned in this manner, the total number of protons which have been assigned to each nucleus should equal the true nuclear charge.

Many interesting systems (three center bonds, back-bonding, aromatic delocalization, and all charged species) may require you to assist the automatic assignment of nuclear charge. First, note that MOIDON reorders the localized orbitals into a consistent order: first comes any core and lone pair orbitals on the 1st atom, then any bonds from atom 1 to atoms 2, 3, ..., then any core and lone pairs on atom 2, then any bonds from atom 2 to 3, 4, ..., and so on. Let us consider a simple case where MOIDON fails, the ion NH4+. Assuming the nitrogen is the 1st atom, MOIDON generates

```
 NNUCMO=1,2,2,2,2
   MOIJ=1,1,1,1,1
          2,3,4,5
    ZIJ=2.0,1.0,1.0,1.0,1.0,
            1.0,1.0,1.0,1.0
```

The columns (which are LMOs) are allowed to span up to 5 rows (the nuclei), in situations with multicenter bonds. MOIJ shows the Mulliken analysis thinks there are four NH bonds following the nitrogen core. ZIJ shows that since each such bond assigns one proton to nitrogen, the total charge of N is +6. This is incorrect of course, as indeed will always happen to some nucleus in a charged

molecule. In order for the energy analysis to correctly reproduce the total energy, we must ensure that the charge of nitrogen is +7. The least arbitrary way to do this is to increase the nitrogen charge assigned to each NH bond by 1/4. Since in our case NNUCMO and MOIJ and much of ZIJ are correct, we need only override a small part of them with $LOCAL input:

```
   IJMO(1)=1,2,  1,3,  1,4,  1,5
   ZIJ(1)=1.25, 1.25, 1.25, 1.25
```

which changes the charge of the first atom of orbitals 2 through 5 to 5/4, changing ZIJ to

```
    ZIJ=2.0,1.25,1.25,1.25,1.25,
            1.0, 1.0, 1.0, 1.0
```

The purpose of the IJMO sparse matrix pointer is to let you give only the changed parts of ZIJ and/or MOIJ.

Another way to resolve the problem with NH4+ is to change one of the 4 equivalent bond pairs into a "proton". A "proton" orbital AH treats the LMO as if it were a lone pair on A, and so assigns +2 to nucleus A. Use of

a "proton" also generates an imaginary orbital, with zero electron occupancy. For example, if we make atom 2 in NH4+ a "proton", by

```
     IPROT(1)=2
     NNUCMO(2)=1
     IJMO(1)=1,2,2,2   MOIJ(1)=1,0   ZIJ(1)=2.0,0.0
```

the automatic decomposition of the nuclear charges will be

```
     NNUCMO=1,1,2,2,2,1
       MOIJ=1,1,1,1,1,2
              3,4,5
        ZIJ=2.0,2.0,1.0,1.0,1.0,1.0
                    1.0,1.0,1.0
```

The 6th column is just a proton, and the decomposition will not give any electronic energy associated with this "orbital", since it is vacant. Note that the two ways we have disected the nuclear charges for NH4+ will both yield the correct total energy, but will give very different individual orbital components. Most people will feel that the first assignment is the least arbitrary, since it treats all four NH bonds equivalently.

However you assign the nuclear charges, you must ensure that the sum of all nuclear charges is correct. This is most easily verified by checking that the energy sum equals the total SCF energy of your system.

As another example, H3PO is studied in EXAM26.INP. Here the MOIDON analysis decides the three equivalent orbitals on oxygen are O lone pairs, assigning +2 to the oxygen nucleus for each orbital. This gives Z(O)=9, and Z(P)=14. The least arbitrary way to reduce Z(O) and increase Z(P) is to recognize that there is some backbonding in these "lone pairs" to P, and instead assign the nuclear charge of these three orbitals by 1/3 to P, 5/3 to O.

Because you may have to make several runs, looking carefully at the localized orbital output before the correct nuclear assignments are made, there is an option to skip directly to the decomposition when the orbital localization has already been done. Use

```
   $CONTRL RUNTYP=PROP
   $GUESS  GUESS=MOREAD  NORB=
   $VEC containing the localized orbitals!
   $TWOEI
```

The latter group contains the necessary Coulomb and exchange integrals, which are punched by the first localization, and permits the decomposition to begin immediately.

SCF level dipoles can also be analyzed using the DIPDCM flag in $LOCAL. The theory of the dipole analysis is given in the third paper of the LCD sequence. The following list includes application of the LCD analysis to many problems of chemical interest:

W.England, M.S.Gordon J.Am.Chem.Soc. 93, 4649-4657 (1971)
W.England, M.S.Gordon J.Am.Chem.Soc. 94, 4818-4823 (1972)
M.S.Gordon, W.England J.Am.Chem.Soc. 94, 5168-5178 (1972)

M.S.Gordon, W.England Chem.Phys.Lett. 15, 59-64 (1972)
M.S.Gordon, W.England J.Am.Chem.Soc. 95, 1753-1760 (1973)
M.S.Gordon J.Mol.Struct. 23, 399 (1974)
W.England, M.S.Gordon, K.Ruedenberg,
Theoret.Chim.Acta 37, 177-216 (1975)
J.H.Jensen, M.S.Gordon, J.Phys.Chem. 99, 8091-8107(1995)
J.H.Jensen, M.S.Gordon, J.Am.Chem.Soc. 117, 8159-8170(1995)
M.S.Gordon, J.H.Jensen, Acc.Chem.Res. 29, 536-543(1996)

* * *

It is also possible to analyze the MP2 correlation correction in terms of localized orbitals, for the RHF case. The method is that of G.Peterssen and M.L.Al-Laham, J.Chem.Phys., 94, 6081-6090 (1991). Any type of localized orbital may be used, and because the MP2 calculation typically omits cores, the $LOCAL group will normally include only valence orbitals in the localization. As mentioned already, the analysis of the MP2 correction must be done in a separate run from the SCF analysis, which must include cores in order to sum up to the total SCF energy.

* * *

Typically, the results are most easily interpreted by looking at "the bigger picture" than at "the details". Plots of kinetic and potential energy, normally as a function of some coordinate such as distance along an IRC, are the most revealing. Once you determine, for example, that the most significant contribution to the total energy is the kinetic energy, you may wish to look further into the minutia, such as the kinetic energies of individual localized orbitals, or groups of LMOs corresponding to an entire functional group.

Transition Moments and Spin-Orbit Coupling

PC GAMESS can compute transition moments and oscillator strengths for the radiative transition between two CI wavefunctions. The moments are computed using both the "length (dipole) form" and "velocity form". PC GAMESS can also compute the one electron portion of the "microscopic Breit-Pauli spin orbit operator". The two electron terms can be approximately accounted for by means of effective nuclear charges.

The orbitals for the CI can be one common set of orbitals used by all CI states. If one set of orbitals is used, the transition moment or spin-orbit coupling can be found for any type of CI wavefunction PC GAMESS can compute.

Alternatively, two sets of orbitals (obtained from separate MCSCF orbital optimizations) can be used. Two separate CIs will be carried out (in 1 run). The two MO sets must share a common set of frozen core orbitals, and the CI -must- be of the complete active space type. These restrictions are needed to leave the CI wavefunctions

invariant under the necessary transformation to corresponding orbitals. The non-orthogonal procedure implemented in PC GAMESS is a GUGA driven equivalent to the method of Lengsfield, et al. Note that the FOCI and SOCI methods described by these workers are not available in PC GAMESS.

If you would like to use separate orbitals for the states, use the FCORE option in $MCSCF in a SCFTYP=MCSCF optimization of the orbitals. Typically you would optimize the ground state completely, and use these MCSCF orbitals in an optimization of the excited state, under the constraint of FCORE=.TRUE. The core orbitals of such a MCSCF optimization should be declared MCC, rather than FZC, even though they are frozen.

In the case of transition moments either one or two CI calculations are performed, necessarily on states of the same multiplicity. Thus, only a $CIDRT1 is read. A spin-orbit coupling run almost always does two or more CI calculations, as the states to be coupled are usually of different multiplicities. So, spin-orbit runs might read only $CIDRT1, but normally read several, $CIDRT1, $CIDRT2, ... The first CI calculation, defined by $CIDRT1, must be for the state of lower spin multiplicity, with $CIDRT2, $CIDRT3, ... being successively higher multiplicities.

You will probably have to lower the symmetry in $CIDRT1 and $CIDRT2 to C1. You may use full spatial symmetry in the CIDRT groups only if the two states happen to have the same total spatial symmetry.

The spin-orbit operator contains a one electron term arising from the Pauli's reduction of the hydrogenic Dirac equation to one-component form, and a two electron term added by Breit. At present, the code for treating the full Breit-Pauli operator is limited to consideration of only singlet states with one triplet state. The active space for METHOD=BREIT is limited to 10 active orbitals on 32 bit machines and about 16 on 64 bit machines.

As an approximation, the nuclear charge appearing in the one electron term can be regarded as an empirical scale factor, compensating for the omission of the two electron operator. This is METHOD=ZEFF, and is general both to any number of active orbitals or spin muptiplicities. The values of ZEFF may be very different from the true nuclear charge if ECP basis sets are in use, see the two references mentioned below.

For Pauli-Breit runs you can several options control the form factors calculation. For large active spaces you may want to precalculate the form factors and save them to disk by using the ACTION option. In case if you do not provide enough storage for the form factors sorting then some extra disk space will be used; the extra disk space can be eliminated if you set SAVDSK=.TRUE. (the amount of savings depends on the active space and memory provided, it some cases it can decrease the disk space up to one order

of magnitude). The form factors are in binary format, and so can be transfered between computers only if they have compatible binary files. There is a built-in check for consistency of a restart file DAFL30 with the current run parameters.

The transition moment and spin orbit coupling driver is a rather restricted path through PC GAMESS, in that

1) Give SCFTYP=NONE. $GUESS is not read, as the program expects to MOREAD the orbitals $VEC1 group, and perhaps $VEC2,... groups. It is not possible to reorder MOs.

2) $CIINP is not read. The CI is hardwired to consist of CIDRT generation, integral transformation/sorting, Hamiltonian generation, and diagonalization. This means $CIDRT1 (and maybe $CIDRT2,...), $TRANS, $CISORT, $GUGEM, and $GUGDIA input is read, and acted upon.

3) The density matrices are not generated, and so no properties (other than the transition moment or the spin-orbit coupling) are computed.

4) There is no restart capability provided.

5) CIDRT input is given in $CIDRT1 and maybe $CIDRT2.

6) IROOTS will determine the number of CI states in each CI for which the properties are calculated. Use NSTATE to specify the number of CI states for the CI Hamiltonian diagonalisation. Sometimes the CI convergence is assisted by requesting more roots to be found in the diagonalization than you want to include in the property calculation.

7) The spin-orbit integrals permit the basis to be
METHOD=Zeff: s,p,d,f but not l,g (l means sp)
METHOD=Breit: s,p,d,l but not f,g

Reference for separate orbital optimization:
1. B.H.Lengsfield, III, J.A.Jafri, D.H.Phillips, C.W.Bauschlicher, Jr. J.Chem.Phys. 74,6849-6856(1981)

References for transition moments:
2. F.Weinhold, J.Chem.Phys. 54,1874-1881(1970)
3. C.W.Bauschlicher, S.R.Langhoff Theoret.Chim.Acta 79:93-103(1991)
4. "Intramediate Quantum Mechanics, 3rd Ed." Hans A. Bethe, Roman Jackiw Benjamin/Cummings Publishing, Menlo Park, CA (1986), chapters 10 and 11.
5. S.Koseki, M.S.Gordon J.Mol.Spectrosc. 123, 392-404(1987)

References for Zeff spin-orbit coupling, and ZEFTYP values:
6. S.Koseki, M.W.Schmidt, M.S.Gordon J.Phys.Chem. 96, 10768-10772 (1992)
7. S.Koseki, M.S.Gordon, M.W.Schmidt, N.Matsunaga J.Phys.Chem. 99, 12764-12772 (1995)
8. N.Matsunaga, S.Koseki, M.S.Gordon J.Chem.Phys. 104, 7988-7996 (1996)

9. S.Koseki, M.W.Schmidt, M.S.Gordon
   J.Phys.Chem.A 102, 10430-10435 (1998)

References for full Breit-Pauli spin-orbit coupling:
10. T.R.Furlani, H.F.King
    J.Chem.Phys. 82, 5577-5583 (1985)
11. H.F.King, T.R.Furlani
    J.Comput.Chem. 9, 771-778 (1988)

* * *

Special thanks to Bob Cave and Dave Feller for their
assistance in performing check spin-orbit coupling runs
with the MELDF programs. Special thanks to Tom Furlani
for contributing his 2e- spin-orbit code and answering
many questions about its interface.

Here is an example. Note that you must know what you are
doing with term symbols, J quantum numbers, point group
symmetry, and so on in order to make skillful use of this
part of the program.

```
!  Compute the splitting of the famous sodium D line.
!
!  The two SCF energies below give an excitation energy
!  of 16,044 cm-1 to the 2-P term.  The computed spin-orbit
!  levels are at RELATIVE E=-10.269 and 5.135 cm-1, which
!  means the 2-P level interval is 15.404 cm-1.
!
!  Charlotte Moore's Atomic Energy Levels, volume 1, gives
!  the experimental 2-P interval as 17.1963, the levels are
!  at 2-S-1/2=0.0, 2-P-1/2=16,956.183, 2-P-3/2=16,973.379
!
```

1. generate ground state 2-S orbitals by conventional ROHF.
   the energy of the ground state is -161.8413919816

```
--- $contrl scftyp=rohf mult=2 $end
--- $system kdiag=3 memory=300000 $end
--- $guess  guess=huckel $end
--- $basis  gbasis=n31 ngauss=6 $end
```

2. generate excited state 2-P orbitals, using a state-averaged
   SCF wavefunction to ensure radial degeneracy of the 3p shell
   is preserved. The open shell SCF energy is -161.7682895801.
   The computation is both spin and space restricted open shell
   SCF on the 2-P Russell-Saunders term. Starting orbitals are
   reordered orbitals from step 1.

```
--- $contrl scftyp=gvb mult=2 $end
--- $system kdiag=3 memory=300000 $end
--- $guess  guess=moread norb=13 norder=1 iorder(6)=7,8,9,6 $end
--- $basis  gbasis=n31 ngauss=6 $end
--- $scf    nco=5 nseto=1 no(1)=3 rstrct=.true. couple=.true.
---             f(1)=  1.0  0.16666666666667
---         alpha(1)=  2.0  0.33333333333333  0.0
---          beta(1)= -1.0 -0.16666666666667  0.0 $end
```

3. compute spin orbit coupling in the 2-P term. The use of

C1 symmetry in $CIDRT ensures that all three spatial CSFs
are kept in the CI function. In the preliminary CI, the
spin function is just the alpha spin doublet, and all three
roots should be degenerate, and furthermore equal to the
GVB energy at step 2. The spin-orbit coupling code uses
both doublet spin functions with each of the three spatial
wavefunctions, so the spin-orbit Hamiltonian is a 6x6 matrix.
The two lowest roots of the full 6x6 spin-orbit Hamiltonian
are the doubly degenerate 2-P-1/2 level, while the other
four roots are the degenerate 2-P-3/2 level.

```
 $contrl scftyp=none cityp=guga runtyp=spinorbt mult=2 $end
 $system memory=500000 $end
 $gugdia nstate=3 $end
 $transt numvec=1 numci=1 nfzc=5 nocc=8 iroots=3 zeff=10.04 $end
 $cidrt1 group=c1 fors=.true. nfzc=5 nalp=1 nval=2 $end

 $data
Na atom...2-P excited state...6-31G basis, typed w/o L shells.
Dnh 2

Na 11.0
  s 6 1
    1 9993.2 0.00193766
    2 1499.89 0.0148070
    3 341.951 0.0727055
    4 94.6796 0.252629
    5 29.7345 0.493242
    6 10.0063 0.313169
  s 6 1
    1 150.963 -0.00354208
    2 35.5878 -0.0439588
    3 11.1683 -0.109752
    4 3.90201 0.187398
    5 1.38177 0.646699
    6 0.466382 0.306058
  p 6 1
    1 150.963 0.00500166
    2 35.5878 0.0355109
    3 11.1683 0.142825
    4 3.90201 0.338620
    5 1.38177 0.451579
    6 0.466382 0.273271
  s 3 1
    1 0.497966 -0.248503
    2 0.0843529 -0.131704
    3 0.0666350 1.233520
  p 3 1
    1 0.497966 -0.0230225
    2 0.0843529 0.950359
    3 0.0666350 0.0598579
  s 1 1
    1 0.0259544 1.0
  p 1 1
    1 0.0259544 1.0

 $end

--- GVB ORBITALS --- GENERATED AT  7:46:08 CST 30-MAY-1996
```

```
Na atom...2-P excited state
E(GVB)=     -161.7682895801, 5 ITERS
 $VEC1
...orbitals from step 2 go here...
 $END
```