neci Documentation

Release 0.1

Alavi Group

February 21, 2010

CONTENTS

CHAPTER

ONE

INTRODUCTION

NECI is a rapidly developing code based on a post Hartree–Fock electronic structure method.

It calculates electron correlation via path-resummations in Slater Determinant space [SumPaper], [StarPaper], [ThomPhDThesis]. More recent work has focused on a novel Quantum Monte Carlo theory, FCIQMC, that obtains the Full Configuration Interaction energy. See [FCIQMC] and [Initiator] for more details.

As a standalone package, NECI can perform calculations on electrons confined to a box, the uniform electron gas and the hubbard model.

NECI can also read in wavefunctions, or a set of integrals based on them, of molecular systems produced by another program (e.g. [DALTON] or [MolPro]) and run calculations using them as the basis for forming the necessary Slater Determinants.

Finally, NECI can also be compiled as a library for integration into existing codes. Currently this has been performed for the [CPMD] or [VASP] plane-wave packages, allowing calculations to be performed on periodic systems.

CHAPTER

TWO

THEORETICAL REVIEW

2.1 Introduction

The energy of a system can be evaluated using a standard statisical mechanics result:

$$E = \frac{\mathrm{Tr}[He^{-\beta H}]}{\mathrm{Tr}[e^{-\beta H}]}$$

We choose to work in a Slater Determinant space, which is, by construction, anti-symmetric. In this space the energy expression becomes:

$$\begin{aligned} E &= \frac{\sum_{\mathbf{i}} \langle D_{\mathbf{i}}|He^{-\beta H}|D_{\mathbf{i}}\rangle}{\sum_{veci}\langle D_{\mathbf{i}}|e^{-\beta H}|D_{\mathbf{i}}\rangle} \\ &= \frac{\sum_{\mathbf{i}} w_{\mathbf{i}}\tilde{E}_{\mathbf{i}}}{\sum_{\mathbf{i}} w_{\mathbf{i}}} \end{aligned}$$

A given term in the numerator is simply the differential of the corresponding term in the denominator. There is a cleaner and more efficient way of evaluating the numerator than differentiation, but we will first turn our attention to the denominator.

We can expand each term into a closed path of P steps through the discrete Slater Determinant space:

$$\begin{aligned} w^{P}_{\mathbf{i},\mathbf{i}_1,\mathbf{i}_2,\cdots,\mathbf{i}_p,\mathbf{i}} &= \sum_{\mathbf{i}_1}\sum_{\mathbf{i}_1}\cdots\sum_{\mathbf{i}_P}\langle D_{\mathbf{i}_1}|e^{\beta H/P}|D_{\mathbf{i}_2}\rangle\langle D_{\mathbf{i}_2}|e^{\beta H/P}|D_{\mathbf{i}_3}\rangle\cdots\langle D_{\mathbf{i}_P}|e^{\beta H/P}|D_{\mathbf{i}_1}\rangle \\ &= \sum_{\mathbf{i}_1}\sum_{\mathbf{i}_1}\cdots\sum_{\mathbf{i}_P}\rho_{\mathbf{i}_1\mathbf{i}_2}\rho_{\mathbf{i}_2\mathbf{i}_3}\cdots\rho_{\mathbf{i}_P\mathbf{i}_1}, \end{aligned}$$

where the ρ matrix consists of elements $\rho_{\mathbf{ij}} = \langle D_{\mathbf{i}}|e^{\beta H/P}|D_{\mathbf{j}}\rangle$.

Each path does not necessarily visit $P-1$ determinants: "hopping" terms are allowed. Due to the ρ matrix being diagonally dominant, paths containing small numbers of unique determinants will tend to have a much greater contribution to the overall energy.

The size of the Slater determinant space grows factorially with the number of electrons and virtual orbitals, making it impossible to sum together all the paths. Furthermore, the sign of a path is an incredibly poorly behaved quantity. It is possible to perform an analytical resummation of the paths into objects we term graphs, where each graph contains paths which only visit the vertices contained within the graph.

Note: To come: pictures of paths –> graph.

The expression for the energy now becomes a sum over Slater determinants and a sum graphs which originate from each Slater determinant:

$$E = \frac{\sum_{\mathbf{i}}\sum_{G} w_{\mathbf{i}}[G]\tilde{E}_{\mathbf{i}}[G]}{\sum_{\mathbf{i}}\sum_{G} w_{\mathbf{i}}[G]}$$

Furthermore, the graphs have a much better sign behaviour: there are many graphs with a definite-positive weight, at least for graphs with less than 5 vertices, which makes a Monte Carlo approach feasible.

The resummation of paths into graphs still leaves a sum that is far too large to be completely evaluated. There are various approximations we can apply.

1. Use a single reference reference approach, i.e. approximate the energy with:

$$E = \frac{\sum_G w_{\mathbf{0}}[G]\tilde{E}_{\mathbf{0}}[G]}{\sum_G w_{\mathbf{0}}[G]}$$

   where $\mathbf{0}$ refers to the reference (i.e. Hartree–Fock) determinant.

   This sum, in general, still contains too many terms (and grows too rapidly with system size) to be of much use.

2. Truncate the sum at a certain graph size (e.g. restrict it to two or three vertices). This approach is referred to as a **VERTEX SUM** approach.

3. Find a large graph that is a good approximation to the ground state and is easy to evaluate. Our current model is the single and double excitation star, which contains all single and double excitations connected to the reference determinant but ignores any connections between the excited determinants. In other words, it couples all the single and double excitations together, but only through the reference determinant. This method is referred to as a **VERTEX STAR** approach. The star contains all graphs in the sum truncated at the two vertex level and much more but at no additional costly integrals to evaluate. This makes it a very attractive approach.

Note: To come: the propogation operator.

2.2 Graph evaluation

There are two approaches to evaluating the weight and energy contribution of a given graph: either by diagonalising the ρ matrix of the graph or by diagonalising the Hamiltonian matrix of the graph.

2.2.1 RHODIAG

Diagonalisation of the ρ matrix is referred to as **RHODIAG** in the input documentation.

The ρ matrix of the graph is the evaluation of the high-temperature thermal density operator on the space of Slater determinants spanned by the graph, or more formally:

$$\rho[G] = \sum_{\mathbf{ij}\in G} |D_{\mathbf{i}}\rangle \rho_{\mathbf{ij}} \langle D_{\mathbf{j}}|$$

We can obtain the eigenvectors and -values, $\{v_k\}$ and $\{\lambda_k\}$ of $\rho[G]$ via matrix diagonalisation, and can then use them to evaluate the weight of the graph:

$$\begin{aligned} w_{\mathbf{i}}[G] &= \langle D_{\mathbf{i}}|e^{-\beta H[G]}|D_{\mathbf{i}}\rangle \\ &= \sum_{kl} \langle D_{\mathbf{i}}|v_k\rangle\langle v_k|e^{-\beta H[G]}|v_l\rangle\langle v_l|D_{\mathbf{i}}\rangle \\ &= \sum_{kl} \langle D_{\mathbf{i}}|v_k\rangle\langle v_k|\lambda_l^P|v_l\rangle\langle v_l|D_{\mathbf{i}}\rangle \\ &= \sum_{kl} \langle D_{\mathbf{i}}|v_k\rangle\lambda_l^P\delta_{kl}\langle v_l|D_{\mathbf{i}}\rangle \\ &= \sum_{k} \lambda_k^P \langle D_{\mathbf{i}}|v_k\rangle\langle v_k|D_{\mathbf{i}}\rangle \end{aligned}$$

where we have applied the identity operator, $\sum_k |v_k\rangle\langle v_k|$ twice and used:

$$e^{-\beta H[G]}|v_l\rangle = \rho[G]^{P-1}\lambda_l|v_l\rangle.$$

In a similar fashion, the energy contribution, $w_{\mathbf{i}}\tilde{E}_{\mathbf{i}}$ can be evaluated:

$$\begin{aligned}
w_{\mathbf{i}}\tilde{E}_{\mathbf{i}} &= \langle D_{\mathbf{i}}|He^{-\beta H[G]}|D_{\mathbf{i}}\rangle \\
&= \sum_{\mathbf{j}\in G}\langle D_{\mathbf{i}}|H|D_{\mathbf{j}}\rangle\langle D_{\mathbf{j}}|e^{-\beta H[G]}|D_{\mathbf{i}}\rangle \\
&= \sum_{\mathbf{j}\in G}\sum_{kl}\langle D_{\mathbf{i}}|H|D_{\mathbf{j}}\rangle\langle D_{\mathbf{j}}|v_k\rangle\langle v_k|e^{-\beta H[G]}|v_l\rangle\langle v_l|D_{\mathbf{i}}\rangle \\
&= \sum_{\mathbf{j}\in G}\sum_{kl}\langle D_{\mathbf{i}}|H|D_{\mathbf{j}}\rangle\langle D_{\mathbf{j}}|v_k\rangle\lambda_l^P\delta_{kl}\langle v_l|D_{\mathbf{i}}\rangle \\
&= \sum_{\mathbf{j}\in G}\sum_{k}\langle D_{\mathbf{i}}|H|D_{\mathbf{j}}\rangle\lambda_k^P\langle D_{\mathbf{j}}|v_k\rangle\langle v_k|D_{\mathbf{i}}\rangle
\end{aligned}$$

The ρ matrix elements can be evaluated using a Taylor expansion with or without a Trotter approximation to improve the accuracy of the expansion.

2.2.2 HDIAG

Alternatively, we can use a slightly simpler approach which avoids having to evaluate ρ matrix by dealing with the Hamiltonian matrix directly. This method is referred to as **HDIAG** in the input documentation. The two approaches give essentially the same result. In an analogous fashion to the application of the rho matrix in the space of the graph, we consider the Hamiltonian to be a propogator acting in the space of the graph:

$$H[G] = \sum_{\mathbf{ij}\in G}|D_{\mathbf{i}}\rangle H_{\mathbf{ij}}\langle D_{\mathbf{j}}|$$

We can evaluate use this to evaluate the weight of the graph:

$$\begin{aligned}
w_{\mathbf{i}}[G] &= \langle D_{\mathbf{i}}|e^{-\beta H[G]}|D_{\mathbf{i}}\rangle \\
&= \sum_{kl}\langle D_{\mathbf{i}}|v_k\rangle\langle v_k|1-\beta H[G]+\frac{\beta^2H[G]^2}{2!}-\frac{\beta^3H[G]^3}{3!}+\cdots|v_l\rangle\langle v_l|D_{\mathbf{i}}\rangle \\
&= \sum_{kl}\langle D_{\mathbf{i}}|v_k\rangle(1-\beta\lambda_l+\frac{\beta^2\lambda_l^2}{2!}-\frac{\beta^3\lambda_l^3}{3!}+\cdots)\delta_{kl}\langle v_l|D_{\mathbf{i}}\rangle \\
&= \sum_{k}e^{-\beta\lambda_k}\langle D_{\mathbf{i}}|v_k\rangle\langle v_k|D_{\mathbf{i}}\rangle
\end{aligned}$$

where now $\{v_k\}$ and $\{\lambda_k\}$ are eigenvectors and -values of the Hamiltonian matrix in the space of the graph.

Similarly, we can obtain the energy contribution of the graph:

$$\begin{aligned}
w_{\mathbf{i}}\tilde{E}_{\mathbf{i}} &= \langle D_{\mathbf{i}}|He^{-\beta H[G]}|D_{\mathbf{i}}\rangle \\
&= \sum_{\mathbf{j}\in G}\langle D_{\mathbf{i}}|H|D_{\mathbf{j}}\rangle\langle D_{\mathbf{j}}|e^{-\beta H[G]}|D_{\mathbf{i}}\rangle \\
&= \sum_{\mathbf{j}\in G}\sum_{kl}\langle D_{\mathbf{i}}|H|D_{\mathbf{j}}\rangle\langle D_{\mathbf{j}}|v_k\rangle\langle v_k|1-\beta H[G]+\frac{\beta^2H[G]^2}{2!}-\frac{\beta^3H[G]^3}{3!}+\cdots|v_l\rangle\langle v_l|D_{\mathbf{i}}\rangle \\
&= \sum_{\mathbf{j}\in G}\sum_{kl}\langle D_{\mathbf{i}}|H|D_{\mathbf{j}}\rangle\langle D_{\mathbf{j}}|v_k\rangle e^{-\beta\lambda_l}\delta_{kl}\langle v_l|D_{\mathbf{i}}\rangle \\
&= \sum_{\mathbf{j}\in G}\sum_{k}\langle D_{\mathbf{i}}|H|D_{\mathbf{j}}\rangle e^{-\beta\lambda_k}\langle D_{\mathbf{j}}|v_k\rangle\langle v_k|D_{\mathbf{i}}\rangle
\end{aligned}$$

CHAPTER

THREE

INSTALLATION

Requirements:

- **git.** The NECI codebase, documentation and test suite are currently distributed only from git repositories hosted on CUC3 workstations (account required).
- **subversion.** Legacy code and a modified version of CPMD are distributed via the wwmm subversion repository (account required), which is kindly hosted by the Unilever Centre.
- LAPACK.
- BLAS.
- FFTW 3.x.

3.1 Download

3.1.1 NECI

The NECI source code can be checked out using:

```
.. code-block:: bash
```

    $ git clone scepter:/home/ajwt3/NECI.git

Various branches also exist, but they may or may not be stable or under active development.

3.1.2 CPMD

Our modified version of CPMD (containing the necessary routines for integration with NECI) can be downloaded from:

```
.. code-block:: bash
```

    $ svn checkout https://wwmm.ch.cam.ac.uk/svn2/groups/alavi/CPMD/branches/QMC/trunk CPMD

Again, there exist various branches and they may or may not be stable.

The default setting used in the compilation scripts assumes that the NECI source is in a subdirectory of the CPMD source directory. This can be changed by using the command line options or setting them in the .compileconf file (see below).

Note that the codebase is not currently under active development.

3.2 File structure

3.2.1 NECI

config/ Directory containing configuration files used for building NECI on a variety of platforms and compilers.

docs/ Documentation.

src/ Source files.

tools/ Tools which are useful for running and analysing NECI output. These include a histogramming script and scripts which automate sending a **SOFTEXIT** signal to a running instance of NECI after a set amount of time.

utils/ Useful scripts for compiling and working on the code base.

The following directories are created during compilation

dest/ Compiled objects.

bin/ Executables.

lib/ NECI libraries for linking to VASP/NECI.

The major compiled files are:

NECI/bin/neci.x Standalone neci executable.

NECI/lib/neci-cpmd.a NECI library for integration with the CPMD gamma-point code.

NECI/lib/neci-vasp.a NECI library for integration with the CPMD k-point code.

NECI/lib/neci-cpmd.a NECI library for integration with the VASP gamma-point code.

NECI/lib/neci-vasp.a NECI library for integration with the VASP k-point code.

All of these are actually symbolic links. The binaries and libraries are unique to the optimisation level and configuration used. The above files merely point to the most recent appropriate binary or library.

3.2.2 CPMD

CONFIGURE/ Directory containing the configuration files for a variety of platforms.

dest/ Build directory for gamma-point code.

kdest/ Build directory for k-point code.

gcpmd.x Gamma-point executable of the CPMD-NECI code (links to CPMD/dest/cpmd.x). Must not be used for k-point calculations!

kcpmd.x k-point executable of the CPMD-NECI code (links to CPMD/kdest/cpmd.x). Must not be used for gamma-point calculations!

3.3 Compilation

The settings provided are mainly for CUC3 machines and other computer clusters that the Alavi group has access to and these might need to be adjusted in order to compile in different environments. Most of the time only the paths and flags for the FFTW, LAPACK and BLAS libraries, given in the LFLAGS variable, will need to be adjusted (if that). It is necessary to use the same compiler to compile CPMD as was used to produce the NECI library for CPMD and similarly for integration with VASP.

We have compiled and tested the codebases with the gfortran (4.2 and later), Portland, Pathscale and Intel compilers in 32 and 64 bit.

3.3.1 NECI

To compile NECI run:

```
.. code-block:: bash
```

$ ./tools/mkconfig.py platform $ make

The platform is a filename in the config directory. The mkconfig.py script has some useful options. Run:

```
.. code-block:: bash
```

$ ./tools/mkconfig.py –help

to see information on them.

The objects are compiled to dest/platform/optimised/real (or complex, for compiling the complex version for libraries). The resultant executable, neci.platform.optimised.x, is placed in the exe directory.

If the debug flag [-g] is given to mkconfig.py, then the debug configuration is used and the filenames and paths contain debug rather than optimised.

For convenience, bin/neci.x is a symbolic link to the most recently compiled executable.

Compiling libraries works in much the same way.

There are several goals defined in the makefile. Run

```
$ make help
```

to see the most useful.

Configuration files use a simple ini format and adding a new configuration is easy. See the supplied configurations for examples and/or comments at the start of the mkconfig.py script.

3.3.2 CPMD

For historical reasons, CPMD and NECI are much more closely interwoven than NECI is with VASP.

The repository version of CPMD depends upon NECI, thus NECI must be compiled first. Various scripts take care of this for us.

CPMD and NECI have CONFIGURE subdirectories. Each file in a CONFIGURE subdirector contains the necessary information to compile the code using a certain compiler.

Please note that not all the CPMD configure scripts will work: many of them were supplied with CPMD and have never been used with our modified version. Please only use platforms that exist in the NECI/config directory as well as the CPMD/CONFIGURE directory. It is easy to make your own configuration files using existing ones as a template.

Note that some of the platforms obtain LAPACK and BLAS as part of atlas, ACML or MKL, so this will need to be changed if different source libraries are used.

Two versions of CPMD (and the corresponding NECI library) exist, gcpmd.x (for gamma-point calculations) and kcpmd.x (for k-point calculations), to take advantage of some substantial memory savings when running gamma-point calculations, as all wavefunctions are then real. This is controlled via a C pre-processing statement. As neci.x is only for molecular calculations, it only exists in one form.

runmake.sh is a script for CPMD which controls the process of creating makefiles and compiling the NECI/CPMD hybrid. A “platform” is just the name of one of the configuration files in the CONFIGURE subdirectories.

Run:

```
.. code-block:: bash
```

$ ./runmake.sh -h

to see the various options and default behaviour of the runmake.sh script.

Note that runmake.sh produces new makefiles for CPMD **and** for NECI, and compiles neci.x, the NECI libraries needed for CPMD, and gcpmd.x and kcpmd.x. To aid the speed of recompilation, the real and complex objects are compiled into separate directories.

runmake.sh defaults to compiling the codebases using the Portland 64-bit compiler, if a platform is not specified either via the command line or given in .compileconf, a text file which contains the name of the desired platform. Note that runmake.sh will use the same platform for both the CPMD and NECI makefiles.

The CPMD source directory also contains a controlling Makefile to further help the make process (and generally just acts as a wrapper for the runmake.sh script). Run:

```
make help
```

in each directory to see the various targets available.

To quickest way to compile both CPMD and NECI is to run:

```
[CPMD]$ make all
```

from within the CPMD source directory.

.compileconf

The .compileconf file is not under source code management and allow local defaults to be set.

When running runmake.sh, please note that it uses the CPMD .compileconf information for compiling NECI, rather than the user's default NECI platform. This is to ensure that the same platform is used for both.

The settings in .compileconf are overridden by command line options but override any defaults in the runmake.sh script.

.compileconf in its simplest (and oldest) form simply contains the name of the desired platform, e.g.:

```
PC-ifort64
```

will use the PC-ifort64 platform as the default.

.compileconf can also be used to set local defaults for more variables—see the comments in runmake.sh for more details. Defaults are used for any variables not set in the .compileconf file.

For example, to set different defaults for the platform and the location of the NECI source, .compileconf in the CPMD directory would look like:

```
platform=PC-ifort64
NECIsrc=~/NECI/source
```

3.4 Developing NECI

3.4.1 Default platform

If no platform is given on the command line to the mkconfig.py script, then it looks for a .default file in the config subdirectory, which is used if it exists.

It is thus simple to set a local default platform by creating a symbolic link from the desired platform file to config/default.

3.4.2 Working in the src subdirectory

Note that any goal defined in the Makefile in the root NECI directory can also be run from the src directory (see src/Makefile for how this is achieved).

3.4.3 Adding new files

Just create the new source file in the src subdirectory. The makefile will pick it up automatically. Note that *all* .F, .F90, .c and .C files in src will be compiled and linked to form neci.

3.4.4 Updating dependencies

The dependency list is created automatically if it does not exist using. If this needs to be updated (e.g. due to a new source file or because development results in additional dependencies), run:

```
.. code-block:: bash
```

    $ make depend

Note that this causes an infinite loop if make 3.80 or earlier is used. As an alternative:

```
.. code-block:: bash
```

    $ make rmdeps

deletes all dependency files, which are then automatically regenerated the next time make is run to produce any target. This should be used on older systems.

Note that all platforms share the same set of dependency files.

3.4.5 Testing on different platforms

Because each neci executable is given a file name dependent upon the configuration and optimisation level used, it is easy to test multiple configurations (e.g. parallel and serial) within one local repository.

```
$ ./tools/mkconfig.py -f Makefile.serial PC-ifort64
$ ./tools/mkconfig.py -f Makefile.mpi PC-ifort64-MPI
$ make -f Makefile.serial
$ make -f Makefile.mpi
```

This will compile 2 executables. Recompiling will only involve recompiling any changed files.

3.4.6 tags

Thanks to Alex for pointing this out. It is useful to be able to work on the source code both from the main directory and the src directory and have the same tags file work in both cases.

You can create a tags file by running:

```
$ make tags
```

and tell vim to search the current directory and parent directory (which amounts to working in the main and src directories respectively for our case) by doing:

```
:set tags=./tags,../tags
```

within vim or placing:

```
set tags=./tags,../tags
```

in your $HOME/.vimrc

3.5 testcode

testcode is a set of scripts written by James Spencer that is used to check that our programs produce the same results as they did before. It is useful both for development work, to ensure that regression issues are avoided, and testing successful compilations.

Every night the latest version of the codebase is checked out of the subversion repository and tested against a variety of compilers, giving confidence in the continued stability of the codebase.

The testcode scripts, NECI test suite and NECI/CPMD test suite can be obtained respectively via:

```
.. code-block:: bash
```

$ git clone scepter:/home/jss43/alavi_group/testcode.git $ git clone scepter:/home/jss43/alavi_group/testsuite_neci.git $ git clone scepter:/home/jss43/alavi_group/testsuite_cpmd.git

Please see the testcode documentation for more details.

CHAPTER

FOUR

RUN

4.1 NECI

Note: How to obtain FCIDUMP/density-fitting input files?

```
$ neci.x input_file
```

If no file is given, then it takes input options from STDIN. This is rarely useful, however.

NECI prints output to STDOUT, so output needs to be captured in some way:

```
$ neci.x input_file > output_file
$ neci.x nput_file | tee output_file
```

4.2 CPMD-NECI

The converged Kohn–Sham orbitals obtained from a **OPTIMIZE WAVEFUNCTION** CPMD calculation can be used as input for a NECI calculation.

In contrast to the molecular case, NECI calculations based upon CPMD-generated wavefunctions are called from within CPMD itself. This allows us to take advantage of many routines that CPMD already possesses (FFT routines, initialisation, reading in the wavefunctions etc.).

To run, specify **QMC** in the **&CPMD** section of the CPMD input file. **RESTART WAVEFUNCTIONS OCCUPATION DENSITY COORDINATES LATEST** must also be specified. Running CPMD (assuming it has been correctly compiled with the appropriate NECI library) then calls NECI to read the NECI input file and perform the desired calculation.

For gamma-point calculations:

```
$ gcpmd.x input_file > output_file
```

For k-point calculations:

```
$ kcpmd.x input_file > output_file
```

There are many other appropriate options that can be specified in the CPMD input file rather than the NECI input file. Please see the CPMD manula and the local CPMD documentation detailing addtions the Alavi group has made.

4.3 Interacting with running calculations

It is useful to be able to interact with a running calculation, particularly during iterative and cyclic processes and is not available (nor suitable) for all types of calculation. Currently this functionality is implemented for **FCIMC** calculations.

NECI checks for a file called CHANGEVARS in the working directory of the calculation. CHANGEVARS can be placed on any node when run on multi-node compute servers. The CHANGEVARS file is read and echoed to STDOUT and then deleted so that it does not affect any subsequent calculations.

Valid options to CHANGEVARS are:

EXCITE [excitation_level] Change the maximum excitation level of determinants included in the simulation. A value less than or equal to 0 or greater than the number of electrons in the system sets the maximum excitation level to use the full determinant space.

TRUNCATECAS [OccCASOrbs] [VirtCASOrbs] Change the space used to the specified complete active space (CAS). A value equal to or less than 0 or greater than the number of electrons in the system sets it to the full determinant space.

SOFTEXIT Stop calculation as soon as possible. This facility is useful when running on machines with fixed walltimes, especially when used with the watchdog.py script in the utils/ subdirectory.

WRITEPOPS Print out the current walker population to POPSFILE.

VARYSHIFT Exit fixed shift phase and allow the shift to vary according to [FCIQMC].

NMCYC [ncycles] Change the number of Monte Carlo cycles to perform.

TAU [tau] Change the timestep, tau, for the simulation.

DIAGSHIFT [shift] Change the shift.

SHIFTDAMP [damping] Change the damping parameter used to adjust the shift.

STEPSSHIFT [nsteps] Change the number of Monte Carlo cycles performed between updating the shift.

SINGLESBIAS [bias] Change the bias for generating single excitations over double excitations when using the non-uniform random excitation generator.

ZEROPROJE Rezero the averaged energy estimators. This is useful when the initial value of the energy estimators are a long way from the instantaneous values, causing a slow convergence of the averaged values.

ZEROHIST Rezero the averaged histogramming vectors.

PARTIALLYFREEZE [nPartFrozen nHolesFrozen] Change the maximum number of holes, nHolesFrozen, allowed in the nPartFrozen number of spin-orbitals in the core valence region. Determinants with a larger number of holes in the lowest nPartFrozen spin-orbitals are not considered. See the input option in the **INTEGRALS** section for more details.

PARTIALLYFREEZEVIRT [nVirtPartFrozen nElVirtFrozen] Similar to **PARTIALLYFREEZE**, allow only Slater determinants with at most nElVirtFrozen electrons in the nVirtPartFrozen number of virtual spin-orbitals.

PRINTERRORBLOCKING Print the blocking analysis.

STARTERRORBLOCKING Start the blocking analysis.

RESTARTERRORBLOCKING Restart the blocking analysis.

PRINTSHIFTBLOCKING Print the shift blocking analysis.

RESTARTSHIFTBLOCKING Restart the shift blocking analysis.

EQUILSTEPS [ncycles] Change the number of initial Monte Carlo cycles to ignore in the averaging of the energy and the shift.

STARTHIST Begin histogramming the determinant populations if the tCalcFCIMCPsi is on and the histogramming has been set up.

HISTEQUILSTEPS [ncycles] Change the iteration at which the histogramming begins to the value specified.

TRUNCINITIATOR Expand the CAS calculation to a **TRUNCINITIATOR** calculation if **DELAYTRUNCINITIATOR** is present in the input.

ADDTOINIT [nwalkers] Will change the cutt-off population for which walkers are added to the initiator space. The population must be above specified value.

Many of these options are also valid options in the main input file and are covered in more depth in input.

CHAPTER

FIVE

INPUT OPTIONS

5.1 Overview

The NECI input file is keyword driven and requires a minimal amount of information.

The NECI input file is divided into various sections, or input blocks: system, precalc, calc, integral and logging. Of these, only the system and calc blocks are compulsory: all others are optional. Inside each input block, it is possible to set a variety of options. There are also three types of keywords that exist outside of an input block.

The order of the input blocks is not important (but certain orders are more logical than others), and nor is the order within a block, unless an option is only valid when a logical statement is true, in which case the relevant keyword for the logical statement must precede its related keywords.

General points to note:

- The input file is not case sensitive. In the input documentation, the keywords are given in capitals and **emphasised** for clarity and options or data required are in square brackets.
- Parameters which follow a keyword ought to be on the same line as the keyword,but this isn't a strict requirement.
- A new line is required for each keyword, unless the keyword is an option of another keyword, in which case it ought to be on the same line.
- Blank lines are ignored.
- Comments are enclosed in parentheses.
- Data items are terminated by space or comma.
- Only the variables relevant to the desired run are required.
- Unknown keywords return an error message and stop the run.
- Sensible defaults are set, reducing the amount of information required from the input file. There exist different sets of default options, allowing a large set of variables to be set with one command.

The overall structure, with a reasonably logical layout, is:

TITLE

DEFAULTS

SYSTEM [system type]

[System options]

ENDSYS

PRECALC

[PreCalc options]

ENDPRECALC

CALC

[Calc options]

ENDCALC

INTEGRAL

[Integral options]

ENDINT

LOGGING

[Integral options]

ENDLOG

END

Warning: This is a work in progress. Many places (especially, but not exclusively, where noted) need to be expanded and/or improved.
In addition, the following keywords are valid options, but are *not* documented:

- CALCREALPROD
- CALCRHOPROD
- DELTAH
- DERIV
- DETPOPS
- DIAGSHIFT
- EQUILSTEPS
- EXCHANGE-ATTENUATE
- HAPP
- LINROOTCHANGE
- MAXVERTICES
- MODMPTHEORY
- RESUMFCIMC
- RHOAPP
- RHOELEMS
- SAVEPREVARLOGGING
- SHIFTDAMP
- STARPROD
- STEPSSHIFT
- SUMPRODII

In contrast, the following options are documented, but are *not* valid input options:

- BANDGAP
- EXCHANGE-DAMPING
- STOCHASTICTIME
- MPMODTHEORY
- SAVEPRECALCLOGGING

5.2 Non-block level options

The following options exist outside of any input block:

TITLE Takes the rest of the line as the title and prints it to output. Useful for labelling the output.

DEFAULTS [DEFAULT FEB08] Default: **DEFAULT**. NECI has a default set of defaults (the **DEFAULT** set), which are sensible, safe defaults. The **FEB08** set of defaults reflect furthr work, and change the defaults as follows:

- Fock-Partition-Lowdiag is set in the integral block.

- RhoEpsilon= 10^{-8} in the calc block.
- MCPATHS is set to be on in the logging block.

This can be specified anywhere in the input file outside of an input block. All other options in the input file override the defaults.

END End of input file. Not required, unless there is text after the input (e.g. comments or notes) which is not commented out or if the input file is given via STDIN.

5.3 System

SYSTEM [system type] Starts system block. The system type must be provided and specifies the basis upon which NECI performs a calculation. ORDER is only valid for some system types—see below.

[System options—see below.]

ENDSYS End the system input block.

The available system types fall into three categories:

- Read in data produced by a molecular computational chemistry package:

  READ [ORDER] [NOORDER] Perform a calculation on a (molecular) system based upon reading in the integrals produced by a third-party program from disk.

  GENERIC [ORDER] [NOORDER] Synonym for **READ**.

- Use a model system:

  BOX Run a calculation on electrons confined to a box. See [TwoElBox] for more details.

  HUBBARD Run a Hubbard model calculation.

  UEG Run a uniform electron gas calculation.

- Periodic systems:

  CPMD [ORDER] [NOORDER] Perform a calculation based upon the Kohn–Sham wavefunctions produced by CPMD. Only available in a combined CPMD-NECI executable.

  VASP Perform a calculation based upon the Hartree–Fock wavefunctions produced by VASP. Only available in a combined VASP-NECI executable.

ORDER If **ORDER** is specified directly after **READ**, **GENERIC**, then a quick HF calculation in the space of the orbitals is performed. The orbitals are then reordered according to the HF energies, rather than using the orbital energies read in.

NOORDER Do not re-order the orbitals, but keep them in the order they arrive.

If **CPMD** is followed by **ORDER**, then the CPMD orbitals are ordered, not according to their Kohn–Sham eigenvalues, but instead according to their one-electron energies (i.e. with no exchange or correlation). **ORDER** is not valid for any other system type.

5.3.1 General options

RANLUXLEV [iRanLuxLev] Default=3 Set the random number luxury level.

MERSENNETWIST [OFF] Default=.true.

Use the Mercenne Twister random number generator. This is about 10 times faster than Ranlux. It can be used for FCIMC parallel calculations and random excitation generation using symgenrandexcit2.F90 routines. This is now on by default.

BANDGAP Perform calculations for systems containing NEL, NEL+1, and NEL-1 electrons and extract the band gap energy.

COULOMB [FCOUL] Multiply the strength of the coulomb interaction by FCOUL.

COULOMB-DAMPING ENERGY [μ β] Damp the two-electron coulomb integrals, $\langle ab||cd\rangle$ with factor $f(E_a)f(E_b)f(E_c)f(E_d)$ where $f(E_a) = \mathrm{erfc}(\beta * (E_a - \mu))$. A β of 1 gives a damping range of 2; a β of 40 gives a damping range of 0.05.

COULOMB-DAMPING ORBITAL [ORB β] Damp the coulomb integrals as above, with MU set to be halfway between the energies of ORB and ORB+1.

Note: COULOMB-DAMPING is now disabled [26/7/06].

CSF-OLD [STOT] Default **OFF**. Default STOT=0.

If specified, work in CSFs rather than determinants. CSFs might not function properly for some Monte Carlo, but should work for vertex sums and diagonalization. STOT is twice the magnitude of spin to restrict the resultant space.

This uses the old CSF routines, which cannot make use of random excitation generation.

CSF [STOT] Default **OFF**. Default STOT=0

If specified, work in CSFs rather than determinants. STOT is twice the magnitude of the spin to restrict the resultant space.

It is recommended, although not necessary, to restrict Ms to the same value as the total spin, using the option SPIN-RESTRICT STOT

TRUNCATE-CSF [csf_trunc_level] Default **OFF**

Requires CSF

Use CSFs rather than determinants only for spatial configurations with fewer than csf_trunc_level unpaired electrons. Above this level, transition to using a normal determinential representation.

This avoids the extremely high computation cost of matrix elements for CSFs with large numbers of unpaired electrons, whilst retaining the specified spin structure for the spatial configurations which contribute the most to the final wavefunction.

ELECTRONS [NEL] Specify the number of electrons. Required for all system types apart from CPMD- or VASP-based calculations.

ENERGY-CUTOFF EMax Default off.

Reject basis functions with an (unscaled) energy larger than EMax.

EXCHANGE [ON | OFF] Default **ON**.

Specify whether to include Exchange in the Slater-Condon rules. If off, we are effectively reduced to a using Hartree multi-electron wavefunctions rather than Slater determinants.

UMATEPSILON [UMatEps] Default **OFF**

This works when integrals are read in from an FCIDUMP file. If specified, it provides a cutoff for the magnitude of the two-electron integrals. If the integrals are larger than the size specified, they will be zeroed.

CALCMCSIZESPACE [CalcDetCycles] [CalcDetPrint]

This option will calculation the exact size of the determinant space, including spin- polarization, spatial symmetry, Lz symmetry, and truncation of the excitation level if included, in a MC fashion. It is parallelised, and takes the values (INTEGER*8s) CalcDetCycles - the number of MC cycles per processor, and CalcDetPrint - the number of cycles before the stats of the MC run are printed to a "SpaceMCStats" file. This file gives the iteration number, the number of allowed determinants generated, the fraction of generated determinants which were allowed, and finally, the expected size of the space from the run so far.

CALCEXACTSIZESPACE Default false.

This will calculate the exact size of the symmetry allowed space before any calculations are performed. Only determinants with the same Sz value as the reference are included. This scales badly and is unsuitable for use with large systems.

NONUNIFORMRANDEXCITS [NOSYMGEN | CYCLETHRUORBS] Default false.

These are new excitation generators, currently only interfaced with the parallel FCIMC algorithm. They are generated with normalised probability, but not uniformly. They scale well however at O[N]. NOSYMGEN means that spatial symmetry will not be considered when generating the excitations and cyclethruorbs indicates that only orbitals which are allowed will be randomly selected, although this involves an O[M] loop over the basis and is marginaly slower, but will not need to redraw forbidden orbitals many times. This may be useful for small basis-set sizes with high symmetry.

FAST-EXCITGEN [OFF] Default on. Temporary flag [AJWT 2008/09/22] Used to indicate that if an Abelian symmetry group is present the excitation generators should use optimized routines to take this into account. Not all (i.e. no) excitation generator functions currently work with this. USE WITH CARE This will disable itself if it detects non-abelian symmetry.

> **Warning:** The excitation generators for Abelian symmetries are currently incompatible with density-fitting. Density fitting calculations should use **FAST-EXCITGEN OFF**.

NORENORMRANDEXCITS Default off.

Since we have already calculated the number of excitations possible for each symmetry type, there no need to renormalise all excitations with weight 1. As long as pairs of allowed occupied and virtual orbitals can be chosen without any bias, then we can generate random excitations in O[1] time. This is default off since it will change previous results, however it is strongly recommended to be on for virtually all unweighted MC calculations, since it should speed up generation, especially in low symmetry and/or large systems. However, currently this facility is not possible for use with doubles with abelian symmetry, unless FASTEXCITGEN is OFF, or STORESTATELIST is activated. For single excitations, the list is not needed, and so they will always be chosen faster.

STORESTATELIST Default off.

This indicates that the list of state pairs is stored. This is taken by default to be off, however, for non-abelian symmetry, or if FASTEXCITGEN is OFF, then it will be stored no matter what. The advantage to storing the list is that NORENROMRANDEXCITS can be used with double excitations, leading to quicker generation of determinants if there is no weighting function. However, this can use a not insignificant amount of memory and some of the abelian features in the excitation generator setup are no longer used. It is hoped that soon the ability to generate random unweighted excitations without renormalisation will be available without storage of the state pairs.

ASSUMESIZEEXCITGEN Default off.

This indicates that the size of excitation generator will be calculated on the basis of the upper bound of the memory needed. This means that there is no need to run through the excitations twice to count and then allocate the memory for the excitations. This makes calculation of the excitation generators very much faster. The first entry to symgenexcitit2 will now simply return the maximum size of the excitation generator. This size is actually smaller than the full excitation generators, since various components of the generators is left out, namely: Iterator info, STORE info, nAllowPPS and SymProds arrays. Because of this, the excitation generators are smaller, but also are only useful for random excitation generation. If code which fully enumerates excitations is used with this flag, things will go very wrong.

NEL [NEL] Synonym for **ELECTRONS**.

LZTOT [LzTot] Constrain the z-component of the angular momentum for atomic, diatomic and linear molecules to be LzTot. This means that the orbitals are complex, but the integrals are all real. The orbitals need to be transformed from the HF canonical ones to the complex ones using LzTrans.x (on svn).

NOSYMMETRY Ignore all spatial symmetry information. This does not apply to periodic calculations or the hubbard model.

SPIN-RESTRICT [LMS] Default off. Default LMS=0. Turns spin restriction on, limiting the working space to the z-component of spin being LMS*2.

SYM [l_x, l_y, l_z iSym] Default off.

If specified, limit the working Slater determinant space to the set of determinants with the specified symmetry quantum numbers. The symmetry of a given orbital is specified in one of two ways:

model system calculations: 3 quantum numbers, l_x, l_y, l_z.

molecular or periodic calculations: Symmetry label, iSym, which corresponds to an irreducible representation of the symmetry group.

The symmetry label(s) of each orbital is included in the output, from which the symmetry of the desired set of Slater determinants can be evaluated (albeit in a somewhat laborious manner). All four numbers are required, but only the relevant one(s) are used.

For Abelian symmetry groups, each symmetry is printed out in terms of a propogating vector. Internally an integer label is still used, according to the formula:

$$i_{\mathrm{SYM}} = \sum_{i=1}^{3} p_i * 2^{15^{i-1}}$$

where p_i are the components of the propogating vector.

SYMIGNOREENERGIES When calculating Sym Reps, NECI assumes that orbitals with energies differing by more than 1e-5 do not transform together. Specifying **SYMIGNOREENERGIES** forces NECI to always regard beta/alpha pairs as of the same sym rep (even if they have different actual symmetries). This is mighty dangerous in general, but can be used to perform ROHF and UHF calculations, if orbitals are in paired order.

USEBRILLOUINTHEOREM Apply Brillouin's theorem: the net effect of single-excitations of the Hartree–Fock determinant coupled to the Hartree–Fock determinant is zero, so explicitly exclude such single excitations.

NOBRILLOUINTHEOREM For the FCIMC parallel calculations, brillouins theorem is on by default. To disable this, this keyword is required (for say non-HF orbitals, ROHF orbitals, rotated orbitals...). This is automatically turned on if the **ROHF** or **ROTATEDORBS** keyword is also supplied.

ROTATEORBS [TimeStep] [ConvergedForce] This keyword initiates an iterative rotation of the HF orbitals to find the coefficients that best fit a particular criteria (e.g those which maximise the sum of the <iiii> values). This is followed by two real values, the first indicates the size of the iterative steps, and the second is the force value chosen to indicate convergence. The default time step is 0.01, and convergence value is 0.001. Further options are described below.

SPAWNLISTDETS This means that a file called SpawnOnlyDets will be read in before a spawning calculation, and only the determinants listed in this file will be able to be spawned at. Currently, this only works for FCIMCPar calculation.

ROTATEDORBS This keyword is required in the system block if a ROFCIDUMP file is being read in (after orbital rotation). As the orbitals are no longer the HF orbitals, Brillouin's theorem does not apply, and the projected energy must include contributions from walkers on single (as well as double) excited determinants. NOTE: Currently, if electrons are frozen in a rotation calculation, they are incorporated into the core energy in the ROFCIDUMP file. So the number of electrons specified in an input file which reads in an ROFCIDUMP, needs to be the NEl-No.FrozenEl, and the number frozen in the INTEGRAL block needs to be set to 0. This will hopefully be fixed in the near future.

ROHF This is to be used when we are reading in integrals from an FCIDUMP interface for a *restricted* open-shell system. Without this keyword, ROHF and UHF are treated the same and the integral file and calculations are performed on spin-orbitals. However, for ROHF, this results in a duplication in the storage of the integrals, since integrals of the same spatial orbitals are stored multiple times. With this option, the integrals for ROHF systems are stored as spatial orbitals, not spin orbtials, which leads to a ~16x memory saving! The results should be unchanged by this option, and the integral file can remain in spin-orbitals. A word of warning is that with ROHF systems, the fock eigenvalues for the orbitals are different between alpha and beta spins, but with this, the eigenvalues are written out as the same (the value of the alpha one). This means that the

eigenvalues cannot be trusted and values derived from them will be wrong (such as the chemical potential which is printed out.)

5.3.2 Read options

BINARY Read in an unformatted FCIDUMP file containing the molecular integrals.

DensityFitted Read in a set of density fitted coefficients and coulomb integrals from files SAV_DFaSOL and SAV_Ta_INT (generated by [CamCasp]). One-electron integrals are read in from HONEEL, which also contains $\langle ij|ij\rangle$ and $\langle ij|ji\rangle$ integrals (generated by readintOCC.x—a local package).

RIIntegrals Read in Rsolution of the identity (much the same as Density Fitting) integrals from RIINTDUMP (these are generated by Q-Chem). One-electron and HF eigenvalues are taken from the FCIDUMP file (as well as two-index two-electron integrals).

STARSTORE [BINARY] Only the integrals required for a double-excitation star calculation are read in from an FCIDUMP. The one-electron integrals, which we call TMAT elements, are stored as integrals involving spatial orbitals, meaning that UHF is no longer available. In addition, only non-zero one-electron integrals i are stored. The memory required to store the coulomb integrals is massively reduced, from $\frac{M^4}{8}$ to just $\frac{N^2M^2}{2}$, where M and N are the total number of orbitals and the number of occupied orbitals respecitvely. We only store the $\langle ij|ab\rangle$ integrals in the UMAT array, where i and j are occupied, as well as the $\langle ii|jj\rangle$ and $\langle ij|ij\rangle$ integrals over all states in the UMAT2D array. Can only be used for the 2-vertex sum and the 2-vertex star calculations. If **BINARY** is also specfied, then an unformatted FCIDUMP file is used.

STORE-AS-EXCITATIONS Store determinants as a 4-integer list of orbitals excited from, and orbitals excited to, in comparison to the reference determinant, rather than as an n-electron list of the occupied orbitals in the determinant. This means that the scaling is reduced to N^2M^2 rather than N^3M^2, as we run through the list for each excitation. Currently only working for the 2-vertex star Fock-Partition-Lowdiag calculations.

READCACHEINTS Default=.false.

This means that the FCIDUMP file will be read in the integrals in it will be cached. This means that less space should be used for storage of the integrals, however, it will be slower since the integrals will need to be binary searched.

5.3.3 Model system options

The following apply to electron in a box, Hubbard model and uniform electron gas calculations, unless otherwise noted.

BOXSIZE [A [BOA COA]] Required for **UEG** and **BOX** calculations. BOA and COA optional. Default BOA=COA=1.

Set lattice constants a, b and c respectively, where b and c are defined as a ratio of a.

CELL [NMAXX NMAXY NMAXZ] Maximum basis functions for each dimension. For **HUBBARD** and **UEG**, functions range from -NMAXi to NMAXi, but for **BOX**, they range from 1 to NMAXi, where i=X,Y,Z.

5.3.4 Box options

ALPHA [α] Sets TALPHA=.true. and defines α.

Integrate out the Coulomb singularity by performing part in real space and part in Fourier space, with the division according to the screening parameter α. See [TwoElBox].

MESH [NMSH] Default NMSH=32.

Number of mesh points used for calculating integrals.

5.3.5 Hubbard options

B [BHUB] Default=0.

Sets B (hopping or kinetic energy) parameter for the Hubbard model.

U [UHUB] Default=0.

Sets U (on-site repulsion) parameter for the Hubbard model.

REAL Set Hubbard model to be in real space.

APERIODIC Hubbard model is set to be not periodic.

TILT [ITILTX ITILTY] Default off.

The Hubbard model is tilted and the unit vectors are (x,y)=(ITILTX,ITILTY) and (-y,x). Require x $\geq$ y.

5.3.6 UEG options

EXCHANGE-CUTOFF [R_c] Use the method detailed in [AttenEx] for calculating the exchange integrals.

Sets cutoff distance R_c for the exchange electron-electron potential. If R_c is not explicitly set, it will be set to be equivalent to a sphere of the same volume as the cell, $R_c = (\frac{\Omega}{4\pi/3})^{1/3}$.

EXCHANGE-ATTENUATE [R_c] Use an expoentially attenuated exchange Sets attenuation parameter R_c for the exchange electron-electron potential. If R_c is not explicitly set, it will be set to be equivalent to a sphere of the same volume as the cell, $R_c = (\frac{\Omega}{4\pi/3})^{1/3}$.

EXCHANGE-DAMPING [R_c] Sets cutoff parameter R_c for attenuated or cutoff potential $V(r) = \frac{\mathrm{erfc}(r/R_c)}{r}$. If R_c is not explicitly set, it will be set to be equivalent to a sphere of the same volume as the cell, $R_c = (\frac{\Omega}{4\pi/3})^{1/3}$.

5.3.7 Orbital rotation options

The minimum keywords required for this method are the above described **ROTATEORBS**, the type of rotation / localisation, and an orthonormalisation method.

Type of rotation / localisation:

OFFDIAGSQRDMIN [OffDiagWeight](optional) This method finds the linear combinations of the HF orbitals that most effectively minimise the sum of the <ij|kl>^2 values, where i,j,k,l are spin orbitals and i.ne.k, and j.ne.l.

OFFDIAGSQRDMAX [OffDiagWeight](optional) This method finds the linear combinations of the HF orbitals that most effectively maximise the sum of the <ij|kl>^2 values, where i,j,k,l are spin orbitals and i.ne.k, and j.ne.l.

OFFDIAGMIN [OffDiagWeight](optional) This method finds the linear combinations that minimise the <ij|kl> integrals (without squaring).

OFFDIAGMAX [OffDiagWeight](optional) This method finds the linear combinations that maximise the <ij|kl> integrals (without squaring).

DOUBEXCITEMIN [OffDiagWeight](optional) This method finds the linear combination that minimise the antisymmetrised double excitation hamiltonian elements, <ij|kl> - <ij||lk>.

HFSINGDOUBEXCMAX This minimises the square of the four index integrals corresponding to single or double excitations from the HF determinant. I.e. Integrals of the form <ij|kl> where i and j are orbitals occupied in the HF determinant, and either k and l are both virtual, or k=i or l=j, but not both at once.

VIRTCOULOMBMAX This maximises the sum of the <ij|ij> terms where i and j are both virtual spatial orbitals.

VIRTEXCHANGEMIN This minimises the sum of the <ij|ji> terms where i and j are both virtual spatial orbitals.

ERLOCALIZATION [ERWeight](optional) This method performs a Edmiston-Reudenberg localisation. It finds the coefficients that maximise the sum of the self-repulsion (<ii|ii>) terms. In reality we minimise -<ii|ii> to keep the code consistent.

The presence of both the **ERLOCALIZATION** keyword together with one of the first three options finds the coefficients that both maximise the <ii|ii> terms and also fit the chosen off diagonal criteria. The contribution from each method can be adjusted by weighting the effect of either force. In the absence of values for ERWeight and/or OffDiagWeight, the defaults of 1.0 each will be used. These weights are also redundant if only one of the keywords is present.

ONEPARTORBENMAX [Alpha] This maximises the sum of (e_i - e_min)^alpha, where e_i are the fock, one particle orbital energies (e_i = <i|h|i> + sum_j [<ij||ij>]), and e_min is currently the energy of the HF LUMO. Alpha is a real value specified in the input, with a default value of 1.0.

ONEELINTMAX This maximises the sum of the <i|h|i>, one electron integral values.

HIJSQRDMIN This minimises the square of the one electron integrals, <i|h|j>. Currently i can be occupied or virtual, but j can only be virtual, i=<j.

DIAGONALIZEHIJ This option takes the <i|h|j> matrix of one electron integrals in the HF orbital basis and diagonalises it. It then uses the eigenvectors as the transformation matrix to form a set of new orbitals which have a diagonal <i|h|j> matrix. This is the extreme case of minimising the off diagonal <i|h|j> matrix elements.

READINTRANSMAT With this option, a TRANSFORMMAT file must be provided which contains the transformation matrix to be used for the orbital rotation. When this keyword is present, the coefficient matrix is simply read in from the file, and used to transform the relevant integrals and print a new ROFCIDUMP file.

USEMP2VDM With this option, the MP2 variational density matrix is calculated and then used to transform the orbitals and produce a new ROFCIDUMP file. The MP2VDM is given as follows: MP2VDM = D2_ab = sum_ijc [t_ij^ac (2 t_ij^bc - t_ji^bc)] Where: t_ij^ac = - < a c | i j > / (E_a - E_i + E_b - E_j) Ref: J. Chem. Phys. 131, 034113 (2009) (note Eqn 1 is mis-printed, the cb indices should be bc). Having calculated the MP2VDM matrix, this is diagonalised. The eigenvectors correspond to the transformation matrix, which produce orbitals whose occupation numbers are given by the respective eigenvalues. These eigenvalues ideally decay exponentially, so in principle we may remove some of the low occupancy virtual orbitals without loosing much accuracy in the energy calculation. This truncation of the virtuals is done using the Logging option **TRUNCROFCIDUMP** [NoFrozenVirt].

USECINATORBS This option is similar to **USEMP2VDM** except that the one electron reduced density matrix is used instead of the MP2VDM to transform the orbitals. The 1-RDM has the form: < Psi | a_p+ a_q | Psi >, where a_q is an annihilation and a_p+ the creation operator acting on a determinant in Psi. In order to form this one electron reduced density matrix, we must first find Psi within the required truncation. This is done by performing a spawning calculation and histogramming the occupation at the determinants. The required histogramming is automatically turned on by using the **USECINATORBS** keyword, and at the end of the spawning, the 1-RDM is found from the amplitudes. The orbitals are then rotated using this matrix, and a ROFCIDUMP file of the resulting approximate natural orbitals is printed. The level of natural orbitals found is controlled by truncation of the excitation level in the spawning calculation. E.g. an excite 2 calculation results in the CISD natural orbitals etc.

Each of these methods may be applied for both the cases where symmetry as kept and broken. This is controlled by the absence or presence of the NOSYMMETRY keyword respectively. Also, the default option is to mix all orbitals (occupied and virtual) together.

Orthonormalisation methods:

SHAKE [ShakeConverged] This will use the shake algorithm to iteratively enforce orthonormalisation on the rotation coefficients. Convergence is reached once the sum of the orthonormality constraints is reduced to below the ShakeConverged value.

SHAKEAPPROX This keyword is likely to be used when the matrix inversion required in the full shake algorithm cannot be performed. It initiates an approximatation to the method which treats each con-

straint in succession, and finds an appropriate lambda for each in turn. This clearly converges more slowely than the full method in which all constraints are dealt with simultaneously.

SHAKEITER [ShakeIterMax] The presence of this keyword overrides the ConvergenceLimit specified with the **SHAKE** keyword, and instead the shake algorithm is applied for the number of iterations given by ShakeIterMax. It seems that with only a few iterations, although the coefficients do not remain completely orthonormal at every rotation step, orthonormality is eventually imposed throughout the course of the run.

SHAKEDELAY [ShakeStart] This option sets the shake orthonormalisation algorithm to only kick in after a certain number of rotation iterations, specified by ShakeStart. This potentially allows a large shift in the coefficients away from their starting point before orthonormalisation is enforced.

LAGRANGE This option can only be used if **ROTATEORBS** is specified, and will try to maintain orthonormality of the orbitals via a lagrange multiplier force, rather than an explicit reorthogonalization step each iteration.

Additional options:

ROITERATION [ROIterMax] Much like **SHAKEITER**, the presence of this keyword overrides the convergence criteria on the force, and instead runs for the number of iterations specified here. Note: A **SOFTEXIT** is also an option in this method.

SPINORBS Default=.false. This option ensures that spin orbitals are used in the rotation - as is required for open shell systems for example. IF UHF=.true. is present in the FCIDUMP file, this will be turned on automatically, but in special cases where this is not present and we still want to use spin orbitals, this keyword should be used.

SEPARATEOCCVIRT If present, this keyword separates the orbitals into occupied and virtual and does not allow mixing between the two. This has the advantage of keeping the energy of the reference determinant the same as the HF.

ROTATEOCCONLY This option allows mixing amongst the occupied orbitals only, while keeping the virtual the HF.

ROTATEVIRTONLY This option allows mixing amongst the virtual orbitals only.

USEHFORBS This is mostly used for debugging as it just uses the identity matrix to 'transform' the HF orbitals. Can be compared to original HF results.

Note: With this method come logging options **ROFCIDUMP**, **ROHISTOGRAM**, and **ERHISTOGRAM**.

5.4 PreCalc

Note: George, please improve! My interpretations also need to be checked...

PRECALC Start pre-calculation block. This chooses which weighting parameters to use in Monte Carlo calculation , in order to give minimum variance. This is an optional input block, and is not required if the default parameters are to be used, or are specified explicitly in the **CALC** input block. Currently, only the **IMPORTANCE** parameter, the C **EXCITWEIGHTING** parameter, and the a & b optimal parameters are searched for simultaneously, optimised and parsed through the the main program.

[PreCalc options—see below.]

ENDPRECALC End the precalc input block.

5.4.1 PreCalc Options

Note: George: What on earth are the following:

- A,B etc parameters.
- 1.

- U matrix.

VERTEX [HDIAG RHODIAG] [SUM MC] Similar to the methods section in the **CALC** block, specify the method to use at the next vertex level (the first entry is for the second vertex level) in searching for the best parameters, using the hamiltonian matrix diagonaltisation (**HDIAG**) or the ρ matrix diagonalisation (**RHODIAG**). After this is specified, on the same line, specify whether to calculate the expected variance using the full sum at this vertex level (**SUM**), or using a Monte Carlo sum (**MC**).

Currently, only the **HDIAG** routine works when performing a MC expected variance, though the diagonalisation of the *rho* matrix now works with the full sum.

For example:

```
VERTEX HDIAG SUM
XXX
XXX
VERTEX HDIAG MC
XXX
```

where XXX are the vertex options (see below).

If no further options are specified for a given vertex level, the optimum values of all the **EXCITWEIGHTING** variables will be found, but not used in the main program.

GRIDVAR [A_ExcitFromStart] [A_ExcitFromEnd] [A_ExcitFromStep] [B_ExcitToStart] [B_ExcitToEnd] [B_ExcitToStep] Produce a 3D map of the variance landscape, but do not explicitly calculate the minimum. Vaules for the A parameter start, end, and step must be specified, followed by the same for the B parameter.

LINEVAR [G_VMC_PIStart] [G_VMC_PIEnd] [G_VMC_PIStep] Same as GRIDVAR, but produce a 1D line for one variable - currently only working for the **IMPORTANCE** parameter and the U-matrix element parameter. Shows change in expected variance over the designated range of values.

MEMORISE All the graphs, their excitation generators, weights, energies, and unbiased probabilities should be stored in the memory. Speeds up the calculation of the variance by around twofold. However, this cannot be used for large systems. ~31000 4v graphs, or 22000 3v graphs was the maximum for the nitrogen dimer with the VQZ basis .

Currently only available for **MC** precalculations. If this is not set, then only the first node excitation generators are stored—there should be enough memory for this

PREGRAPHEPSILON [PREWEIGHTEPS] Default 10^{-8}.

Gives the threshold above which the weight of a graph must be if they are to be included in the full precalc variance calculation.

If the graph weight is below the threshold, then the probability of obtaining the graph does not need to be calculated, and so the optimisation routine is faster (but less accurate) at higher thresholds.

TOTALERROR [Desired Error from main calculation] Calculates the required number of cycles so that the final error from the Monte Carlo calculation is equal to the error specified.

Only valid if the highest vertex level in the precalculation stage is the same as the highest vertex level in the main MC calculation, i.e. the highest MC vertex level is independently optimised.

The optimum vertex splitting is also calculated.

TOLERANCE [TOLERANCE] Default 0.1.

The fractional precision to which the optimum parameter is obtained, using the minimisation method.

TRUECYCLES [No.of cycles] Specify the total number of MC cycles that we want to use in the main calculation. The precalculation stage will then automatically split the cycles between the vertex levels in the main calculation, according to how the use statements indicate the parameters are to be split.

The cycles are then split so to best minimise the overall variance of the run, assuming that the variance of the whole run is simply the sum of the variances of the individual vertex results.

5.4.2 Vertex options

For the specified vertex level:

CYCLES [nCYCLES] Specify the number of graphs generated in the MC algorithm for each expected variance calculation in the parameter minimisation algorithm.

Applies only to a vertex level which evaluates the expected variance using the MC algorithm in the **VERTEX** line.

NONE Perform no calculations or optimisations.

UEPSILON [UEPSILON] Default UEPSILON is 0.

Find the optimum C **EXCITWEIGHTING** coefficient and pass through to the main program, unless setting C to be zero changes the expected variance by an amount less than UEPSILON, in which case C is set to be zero.

The calculation of the U matrix elements can be time-consuming in a real MC simulation, yet can have a negligible effect on the final result. In these cases, setting the C coefficient to be zero makes the full MC simulation much faster.

With the the default value of UEPSILON, the optimum value of C will always be used in the main program.

FINDC Find the optimum C **EXCITWEIGHTING** parameter.

The C **EXCITWEIGHTING** parameter. will only be found if this flag is set, or if a **UEPSILON** is set.

By default, the optimisation algoritm will only seek to find the values which give the minimum expected variance by varying the A and B **EXCITWEIGHTING** parameters, (or the parameters in the weighting scheme specified).

FINDD Find the optimum D **EXCITWEIGHTING** parameter for this vertex level (g_VMC_ExcitToWeight2).

USED Pass the optimum D **EXCITWEIGHTING** parameter found at this vertex level through to the main calculation.

FINDIMPORT Run the optimisation algorithm for the IMPORTANCE parameter.

The optimised value will be found printed out, but will not be passed through to the main calculation.

Can only be set for vertex levels of three or higher for obvious reasons.

Note: And the obvious reasons are?

USEIMPORT Find the optimal **IMPORTANCE** parameter, and use in the main calculation.

As for **FINDIMPORT**, can only be set for vertex levels of three or higher.

USE [MC_VERTEX_LEVEL_1] [MC_VERTEX_LEVEL_2] ... Use in the main calculation the parameters calculated from the specified precalc vertex levels, rather than any other, are to be passed through and used in the main program when performing a MC at one of the vertex levels specified. Any given vertex level can only be specified once in all the **USE** statements.

Vertex levels in the main calculation which are not specified in one of the precalc **USE** statements, will use the parameters which are given in the **CALC** block of the input file.

A **USE** statement on its own will only calculate the A and B **EXCITWEIGHTING** parameters (or any of the other weighting scheme parameters specified) and use them for all MC vertex levels in the main calculation, unless **FINDC** or **UEPSILON** is specified, in which case for the C parameter to also be used.

Note: This is somewhat confusing. How does it fit in with USEIMPORT etc.?

5.5 Calc

CALC Start calculation block. This chooses what calculation to do.

[Calculation options—see below.]

ENDCALC End the calculation input block.

5.5.1 General options

ALLPATHS Choose all determinants (i.e. set NPATHS = -1).

BETA [BETA] Set β.

BETAOVERP [BETAP] Default= 1.d-4.

Set β/P.

DELTABETA [DBETA] Set $\delta\beta$. If given a negative value, calculate it exactly.

Note: What is this used for?

DETINV [DETINV] Specify the root determinant for which the complete vertex series is worked out, using the determinant index obtained from a previous calculation. If **DETINV** is negative, the NPATHS calculations are started at this determinant.

EXCITE [ICILEVEL] Default 0.

Excitiation level at which to truncate determinant list. If ICILEVEL=0 then all determinants are enumerated. This also works for FCIMC calculations.

EXCITATIONS [OLD NEW] For generation of up to double excitations use the old (completely reliable), or new (faster, but does not work for more than 2-vertex level SUMS) routine

Note: You can now use the **NEW** routines for all methods, right? What is the difference between **NEW** and **OLD**? (If it doesn't say, how else can a user make an informed decision as to which to use?)

EXCITATIONS [SINGLES DOUBLES] Default is to use all excitations.

Restricts determinants which are allowed to be connected to the reference determinant to be either single or double excitations of the reference determinant.

Applies only to the **VERTEX** [**SUM STAR**] **NEW** methods.

HAMILTONIAN [STAR] Store the Hamiltonian. This is defaulted to ON if **ENERGY** is set, but can be used without **ENERGY**.

STAR Only the connections between the root determinant and its excitations should be included in the Hamiltonian and not off-diagonal elements between excited determinants.

MAXVERTICES [MAXVERTICES] Give the vertex level of the calculation. Cannot be used in conjunction with a **METHODS** block.

CONSTRUCTNATORBS Calculates the 1 electron reduced density matrix (1-RDM) as a FCIMC calculation progresses. At the end of the iterations, this matrix is diagonalised to get linear combinations of the HF orbitals which approximate the natural orbitals. The occupation numbers (e-values) of these are printed in the OUTPUT file. This is now a very old option, a much more efficient equivalent has been added under the **ROTATEORBS** option. See **USECINATORBS** in the system file.

METHOD [Method option(s)] Specify the method for a graph theory calculation. See Method options for the available methods.

Can only be specified once if used outside of the methods block, in which case the given method is applied to all vertex levels.

METHODS Begin a methods block. This allows a different method for each vertex level. Each vertex level can contain **EXCITATIONS**, **VERTICES**, **CYCLES** and **CALCVAR** keywords. Each **METHOD** line and the options that follow it detail the calculation type for the next vertex level, with the first **METHOD** line used for the the second-vertex level, unless over-ridden with the **VERTICES** option.

The block terminates with **ENDMETHODS**.

For example:

```
METHODS
   METHOD VERTEX SUM NEW
   EXCITATIONS DOUBLES
   METHOD VERTEX STAR POLY
   EXCITATIONS SINGLES
   VERTICES 2
ENDMETHODS
```

sets the first method, at the two-vertex level, to be a complete 2-vertex sum of only doubles, and the second method, overriden to be also at the two-vertex level, to be a vertex star of singles.

Similarly:

```
METHODS
   METHOD VERTEX SUM NEW
   METHOD VERTEX SUM MC
   [Monte Carlo options]
ENDMETHODS
```

performs a full sum at the two-vertex level and a Monte Carlo calculation at the three-vertex level.

ENDMETHODS Terminate a methods block.

PATHS [option] Select the number of determinants taken to be the root of the graph. Usually set to 1. Valid options:

NPATHS Choose the first NPATHS determinants and calculate RHOPII etc.

ALL Choos all determinants (same as ALLPATHS).

ACTIVE Choose only the active space of determinants: the degenerate set containing the highest energy electron.

ACTIVE ORBITALS nDown nUp Set the active space to be nDown and nUp orbitals respectively from the Fermi level

ACTIVE SETS nDown nUp Set the active space to be nDown and nUp degenerate sets respectively from the Fermi level

RHOEPSILON [RHOEPSILON] Set the minimum significant value of an element in the rho matrix as a fraction of the maximum value in the rho matrix. Matrix elements below this threshold are set to be 0.

STARCONVERGE [STARCONV] Default 1.d-3.

Set the convergence criteria for whether a roots to the star graph is significant.

TROTTER Default.

Perform a Trotter decomposition to evaluate the rho matrix elements.

TIMESTEPS [I_P] Set P, the timesteps into which $e^{-\beta H}$ is split. Automatically sets $\beta/P = 0$ (as required) but returns an error message if **BETAOVERP** is also used.

WORKOUT [NDETWORK] Sets the number of determinants which are worked out exactly.

Note: What is this used for?

VERTICES Only available inside a methods block.

By default, each method takes a number of vertices corresponding to its index within the methods block, the first methods corresponding to the 2-vertex level, the second to the 3-vertex level, and so on. **VERTICES** overrides this, and allows the vertex level of each method to be explicitly specified, enabling, for example, the 2-vertex level to be split up and the contributions from single and double excitations of the reference determinant to be handled separately.

5.5.2 Method options

VERTEX SUM [OLD NEW HDIAG] [SUB2VSTAR] [LOGWEIGHT] Calculate the vertex sum approximation.

OLD Diagonalise the ρ matrix using the original method.

NEW Diagonalise the ρ matrix using a more modern, more efficient method. Recommended.

HDIAG Diagonalise the Hamiltonian matrix instead of the rho matrix in order to calculate the weight and energy contribution of each graph.

SUB2VSTAR Remove paths which were present in the 2-vertex star for each graph. If this is specified for ANY vertex level, it applies to all **SUM** and MC vertex levels.

LOGWEIGHT Form Q as a multiplication of factors from graphs. This results in the quantity $\log w$ being used instead of w, which also translates to the energy expression only involving $\tilde{E}$ not weights. Hopefully this is size-consistent.

> **Warning:** **SUB2VSTAR** and **LOGWEIGHT** are experimental options.

VERTEX [MC MCMETROPOLIS MCDIRECT MCMP] [HDIAG] Perform a Monte Carlo calculation.

MCDIRECT Perform direct stochastic sampling for the graph theory vertex sum method, dividing each freshly generated graph by its normalized generation probability.

If **MULTIMCWEIGHT** is specified then the sampling generates graphs from all weighted levels using the weighting - a single MC calculation is performed.

If **MULTIMCWEIGHT** is not specified (default), a separate MC calculation is performed at each vertex level. Combined statistics are printed.

> **Warning:** **MULTIMCWEIGHT** is not documented. Use with great caution.

MCMP Perform direct stochastic sampling, as in **MCDIRECT**, but for the Moller–Plesset method.

MC or MCMETROPOLIS Perform Metropolis Monte Carlo.

This may be performed in a number of ways. The way is chosen by the location of the **VERTEX MC** command.

Warning: The following options appear in INPUT_DOC but, however, are incredibly poorly documented. In particular:

- No detail on the arguments the options take (e.g. **BIAS**).
- Some options documented don't exist (e.g. **SINGLE**, **BIAS**, **MULTI**, **STOCHASTICTIME**).
- Sufficient tests are not present in the test suite.

Do not use.
The "options" are:

```
 **STOCHASTICTIME**
     may also be specified to perform stochastic
     time simulations with a given **BIAS**

**SINGLE**
    MC is performed at a single vertex level using a composite
    1-vertex graph containing a full sum previously performed.

**BIAS**
    is used to choose whether a step selects a composite
    (all lower levels) or a normal (this level) graph.  Stochastic
    time MC is performed. This can only be specified in the
    **METHODS** section, and only at the last vertex level.
    Uses **EXCITWEIGHTING** for excitation generation weighting
    and **IMPORTANCE** for graph generation weighting

**MULTI**
    MC is performed at a multiple vertex levels, but still
    using a composite 1-vertex graph containing a full sum
    previously performed. MULTI should be specified in all the
    (contiguous) vertex levels to be included (not composited)
    in the MC.  **BIAS** is used to choose whether a step
    selects a composite (all lower levels) or a normal (the
    **MULTI** levels) graph.  **MULTIMCWEIGHT** is specified
    for each **MULTI** level, and gives a relative weighting
    of selecting the vertex level graphs once a non-composite
    graph is chosen.  Stochastic time MC is performed.
    This can only be specified in the **METHODS** section.
    Once **MULTI** has been specified, it must be specified
    on all subsequent vertex levels in a **METHODS** section.
    Uses **EXCITWEIGHTING** for excitation generation weighting
    and **IMPORTANCE** for graph generation weighting

**FULL**
    Does  MC at all levels using BIAS to bias the levels,
    **EXCITWEIGHTING** for excitation generation, and
    **IMPORTANCE** to for graph generation weighting.  This is
    only available *WITHOUT* a **METHODS** section. If **HDIAG**
    is specified, the H-diagonalizing routine is used, otherwise,
    the rho-diagonalizer is used.  **HDIAG** is automatically
    specified for **MCMP**.
```

VERTEX SUM READ Read in from pre-existing MCPATHS file for that vertex level. Only really useful in a **METHODS** section.

VERTEX STAR [ADDSINGLES COUNTEXCITS] [star method] [OLD NEW [H0]] Construct a single and double excitation star from all determinants connected to the root (ignoring connections between those dets). See [StarPaper] for more details.

ADDSINGLES Extend the star graph approach.

Add the single exctitaions which are en-route to each double excitation to that double excitation as spokes, and prediagonalize the mini-star centred on each double excitation. For example, if the double

excitation is (ij->ab), then singles (i->a),(i->b),(j->a) and (j->b) are created in a star with (ij->ab), the result diagonalized, and the eigenvalues and vectors used to create a new spoke of the main star graph.

Only works with **NEW**.

COUNTEXCITS Run through all the symmetry allowed excitations first and count the connected determinants on the star. Enables the memory requirements to be reduced as only connected determinants need to be stored. However, the time taken is increased, as it is necessary to run through all determinants in the star twice. Especially useful for large systems with memory restraints, when density fitting has necessarily turned off symmetry. Also useful if a **RHOEPSILON** has been set to a large value so that many of the symmetry allowed excitations will be counted as disconnected.

Note: Useful for periodic calculations? Does it need just the symmetry info or the transition matrix elements as well?

OLD Use a pre-generated list of determinants using the excitation routine version specified in **EXCITATIONS OLD** or **EXCITATIONS NEW**.

NEW Generate determinants on the fly without storing them, using the **NEW** excitation routine. Much more memory efficient.

NEW H0 Use the zeroth order N-particle Hamiltonian (shifted such that $H^0_{ii} = H_{ii}$) rather than the fully interacting Hamiltonian to generate the roots of the polynomial.

Note: And you'd want to use **NEW H0** why exactly?

The available star methods are:

DIAG Perform a complete diagonalization on the resultant matrix. This can be very slow. However, by specifying **LANCZOS** in the **CALC** block, you can do a Lanczos diagonalisation, which scales much better. **EIGENVALUES** can also be specify to only evaluate the first few eigenvalues.

POLY Use the special properties of the matrix to find the roots of the polynomial and uses them to calculate the relevant values. This is order Ngraph^2.

Note: Ngraph==nDets?

POLYMAX Similar to **POLY** but only finds the highest root of the polynomial, so is order Ngraph. It can be used when P is very large (i.e. β is very large, e.g. 40).

POLYCONVERGE Similar to **POLY** but adds i out of N λ_i roots, such that $(N-i)\lambda_i^P < 10^{-3}$, i.e. we evaluate enough roots such that a very conservative error estimate of the contribution of the remaining roots is negligible.

POLYCONVERGE2 Similar to **POLYCONVERGE** but requires $w(1..i)(N-i)\lambda_i^P < 10^{-3}$, where $w(1..i)$ is the cumulative sum of λ_i^P, which should be a better estimate of the convergence.

The following are experimental star methods:

MCSTAR Use a basic implementation of the spawning algorithm in order to sample the star graph stochastically. The sampling uses elements of the Hamiltonian matrix rather than the rho matrix, so there will be some differences in the converged energy compared to a **VERTEX STAR NEW** calculation.

Many of the **FCIMC** options are also available with MCStar, and there are also some extra one.

NODAL Prediagonalise a completely connected set of virtuals for each set of occupied (i,j) spin-orbitals. The diagonalised excitations are then solved as a star graph. Must be used with **NEW**.

STARSTARS Use an approximation that the change of eigenvalues and the first element of the eigenvectors of the star graph is linear with respect to multiplying the diagonal elements by a constant. Once this scaling is found, all stars of stars are prediagonalised, and reattached to the original graph. This results in N^2 scaling, where N is the number of excitations.

TRIPLES Prediagonalise an excited star of triple excitations from each double excitation, reattach the eigenvectors, and solves the complete star. Currently only available with '**NEW**', '**COUNTEXCITS**' and '**DIAG**'.

Experimental methods

VERTEX FCIMC [MCDIFFUSION] [RESUMFCIMC] [SERIAL] Perform Monte Carlo calculations over pure determinant space, which is sampled using a series of 'particles' (or 'walkers').

The walkers are not necessarily unique and must be sorted at every iteration. Each walker has its own excitation generator.

MCDIFFUSION is a completely particle-conserving diffusion algorithm and is much more experimental.

FCIMC and **MCDETS** calculations share many of the same options (see Walker Monte Carlo options, below).

RESUMFCIMC creates graphs out of connected determinants, and applies the H-matrix successively in order to achieve a local spawning algorithm. This reduces to the original spawning algorithm when **GRAPHSIZE** is 2 and **HAPP** is 1. Uses many of the same options as **FCIMC**.

SERIAL will force NECI to run the serial FCIMC code (which differs substantially from the parallel) even if the code was compiled in parallel.

VERTEX CCMC [FCI] [EXACTCLUSTER] [AMPLITUDE] [EXACTSPAWN] [BUFFER] Perform Monte Carlo calculations over coupled cluster excitation space, which is sampled using a series of 'particles' (or 'walkers').

The walkers are not necessarily unique and must be sorted at every iteration. Each walker has its own excitation generator. **DIRECTANNIHILATION** (in CALC) and **NONUNIFORMRANDEXCITS** (in the SYSTEM section) must also be specified.

If **FCI** is specified, then the code runs an equivalent of the **VERTEX FCIMC** for testing (by only allowing clusters of up to a single excitor, and using (1+T)|D_0> as the wavefunction rather than exp(T)|D_0>

EXACTCLUSTER is an exponentially scaling (with number of walkers) algorithm for testing the stochastic sampling, and explicitly attempts spawning from all clusters in the space.

AMPLITUDE will enumerate the whole of the allowed space, and assign a floating-point amplitude to each excitor. These amplitudes are stochastically sampled (**INITWALKERS** times per MC cycle), and used to propagate the CCMC.

EXACTSPAWN causes spawning to be done exactly - i.e. all allowed connected determinants from a given cluster are spawned to.

BUFFER will accumulate all collapsed cluster selections in a buffer and then do spawnings from that. When using **EXACTCLUSTER** this is much more efficient.

Extremely experimental.

VERTEX GRAPHMORPH [HDIAG] Set up an initial graph and systematically improve it, by applying the rho matrix of the graph and its excitations as a propagator on the largest eigenvector of the graph. From this, an improved graph is stochastically selected, and the process is repeated, lowering the energy. If **HDIAG** is specified, it is the hamiltonian matrix elements which determine the coupling between determinants, and it is the hamiltonian matrix which is diagonalised in each iteration in order to calculate the energy.

Note: **GRAPHMORPH** has not been tested with complex wavefunctions. It will almost certainly not work for them.

VERTEX MCDETS Perform Monte Carlo calculations over pure determinant space, which is sampled using a series of 'particles' (or 'walkers').

MCDETS is similar to **FCIMC** but maintains at most one 'particle' at each determinant, which may then contain subparticles (which correspond to the individual 'walkers' in **FCIMC**), in a binary tree. This makes some efficiency savings where the same information about a determinant is not duplicated.

FCIMC and **MCDETS** calculations share many of the same options (see Walker Monte Carlo options, below).

VERTEX RETURNPATHMC Use a spawning algorithm which is constrained in three ways:

1. a particle can only be spawned where it will increase its excitation level with respect to the reference determinant or back to where it was spawned from.
2. they will spawn back to where their parents were spawned from with probability PRet, which is specified using **RETURNBIAS**.
3. length of spawning chain must be less than the maximum length given by **MAXCHAINLENGTH**.

Note: How can a particle be restricted to spawning to spawning at most back to where it was spawned from *and* have a probability of spawning back to where its parent was spawed from? Documentation *must* be clearer.

This attempts to circumvent any sign problem in the double excitations and the HF, and hopefully this will result in a more stable MC algorithm. It remains to be seen if this approach is useful. Should revert to the star graph in the limit of the return bias tending to 1 or the length of the spawn chain tending to 1.

Note: **FCIMC**, **GRAPHMORPH**, **MCDETS** and **RETURNPATHMC** have not been tested with complex wavefunctions. It will almost certainly not work for them.

All four are experimental options under development.

5.5.3 Walker Monte Carlo options

The following options are applicable for both the **FCIMC** and **MCDETS** methods:

Note: I have made some guesses on the following option names. Clearly some keys are broken on George's keyboard. Specifically:

```
StepsSft --> STEPSSHIFT
SftDamp  --> SHIFTDAMP
DiagSft  --> DIAGSHIFT
```

I also had to guess about **BINCANCEL**. It seems to be a **FCIMC** option, but was placed with **MCSTAR** (and was with all the **VERTEX STAR** methods).

This section needs to be extended substantially.

DIAGSHIFT [DiagSft] Set the initial value of the diagonal shift.

INITWALKERS [nWalkers]

Default 3000.

Set the initial population of walkers. For CCMC Amplitude, this is the number of samples of the amplitude distribution taken each MC step

NMCYC [NMCYC] Set the total number of timesteps to take.

SHIFTDAMP [SftDamp] Damping factor of the change in shift when it is updated. <1 means more damping.

STEPSSHIFT [StepsSft] Default 100.

Set the number of steps taken before the diagonal shift is updated.

TAU [TIMESTEP] Default 0.0.

For FCIMC, this can be considered the timestep of the simulation. It is a constant which will increase/decrease the rate of spawning/death for a given iteration.

The following options are only available in **FCIMC** calculations:

READPOPS Read the initial walker configuration from the file POPSFILE. **DIAGSHIFT** and **INITWALKERS** given in the input will be overwritten with the values read in form POPSFILE.

SCALEWALKERS [fScaleWalkers] Scale the number of walkers by fScaleWalkers, after having read in data from POPSFILE.

STARTMP1 Set the initial configuration of walkers to be proportional to the MP1 wavefunction. The shift will also now be set to the MP2 correlation energy. This also works in CCMC Amplitude

GROWMAXFACTOR [GrowMaxFactor] Default 9000.

Set the factor by which the initial number of particles are allowed to grow before they are culled.

CULLFACTOR [CullFactor] Default 5.

Set the factor by which the total number of particles is reduced once it reaches the GrowMaxFactor limit

EQUILSTEPS [NEquilSteps] Default 0 This indicates the number of cycles which have to pass before the energy of the system from the doubles population is counted

SHIFTEQUILSTEPS [NShiftEquilSteps] Default 1000 This gives the number of iterations after the shift is allowed to change before the shift contributes to the average value printed in column 10. The default is 1000 to simply leave out the first few values where the shift is dropping (usually from 0).

RHOAPP [RhoApp] This is for resummed FCIMC, it indicates the number of propagation steps around each subgraph before particles are assigned to the nodes

SIGNSHIFT This is for FCIMC and involves calculating the change in shift depending on the absolute value of the sum of the signs of the walkers. This should hopefully mean that annihilation is implicitly taken into account. Results were not too good.

Note: details? Why "not good"?

HFRETBIAS [PRet] This is a simple guiding function for FCIMC - if we are at a double excitation, then we return to the HF determinant with a probability PRet. This is unbiased by the acceptance probability of returning to HF.

This is not available in the parallel version.

EXCLUDERANDGUIDE This is an alternative method to unbias for the HFRetBias. It invloves disallowing random excitations back to the guiding function (HF Determinant).

This is not available in the parallel version.

PROJECTE-MP2 This will find the energy by projection of the configuration of walkers onto the MP1 wavefunction. DEVELOPMENTAL and possibly not bug-free.

This is not available in the parallel version.

FIXPARTICLESIGN This uses a modified hamiltonian, whereby all the positive off-diagonal hamiltonian matrix elements are zero. Instead, their diagonals are modified to change the on-site death rate. Particles now have a fixed (positive) sign which cannot be changed and so no annihilation occurs. Results were not good - this was intended for real-space MC, where large regions of connected space were all of the same sign. This is not the case here.

This is not available in the parallel version.

STARTSINGLEPART This will start the simulation with a single positive particle at the HF, and fix the shift at its initial value, until the number of particles gets to the INITPARTICLES value.

MEMORYFACPART [MemoryFacPart] Default 10.D0

MemoryFacPart is the factor by which space will be made available for extra walkers compared to InitWalkers.

MEMORYFACANNIHIL [MemoryFacAnnihil] Default 10.D0

MemoryFacAnnihil is a parallel FCIMC option - it is the factor by which space will be made available for annihilation arrays compared to InitWalkers. This generally will need to be larger than memoryfacpart, because the parallel annihilation may not be exactly load-balanced because of non-uniformity in the wavevector and the hashing algorithm. This will tend to want to be larger when it is running on more processors.

MEMORYFACSPAWN [MemoryFacSpawn] Default 0.5

A parallel FCIMC option for use with **ROTOANNIHILATION**. This is the factor by which space will be made available for spawned particles each iteration. Several of these arrays are needed for the annihilation process. With **ROTOANNIHILATION**, **MEMORYFACANNIHIL** is redundant, but **MEMORYFACPART** still need to be specified.

ANNIHILATEONPROCS Default false

A Parallel FCIMC option. With this, particles are annihilated separately on each node. This should mean less annihilation occurs, but it is effectivly running nProcessor separate simulations. If there are enough particles, then this should be sufficient. Less memory is required, since no hashes need to be stored. Also, no communication is needed, so the routine should scale better as the number of walkers grows.

ROTOANNIHILATION Default false

A parallel FCIMC option which is a different - and hopefully better scaling - algorithm. This is substantially different to previously. It should involve much less memory. **MEMORYFACANNIHIL** is no longer needed (**MEMORYFACPART** still is), and you will need to specify a **MEMORYFACSPAWN** since newly spawned walkers are held on a different array each iteration. Since the newly-spawned particles are annihilated initially among themselves, you can still specify **ANNIHILATEATRANGE** as a keyword, which will change things.

FIXSHELLSHIFT [ShellFix] [FixShift] Default 0,0.D0

An FCIMC option. With this, the shift is fixed at a value given here, but only for the excitation levels at a value of ShellFix or lower. This will almost definitly give the wrong answers for both the energy and the shift, but may be of use in equilibration steps to maintain particle density at low excitations, before writing out the data and letting the shift change.

FIXKIISHIFT FixedKiiCutoff FixShift

Another fixed shift based approximation method for FCIMC in parallel. However, rather than fixing the shift based on an excitation level, it is now fixed according to the Kii value. Determinants lower in energy than FixedKiiCutoff will have their shifts fixed to the value given.

FIXCASSHIFT [OccCASorbs] [VirtCASorbs] [FixShift] Default 0 0 0.D0

A third fixed shift approximation method for FCIMC in parallel. In this option, an active space is chosen containing a number of highest occupied spin orbitals (OccCASorbs) and a number of lowest unoccupied spin orbitals (VirtCASorbs). The shift is then fixed (at FixShift) for determinants with excitations within this space only. I.e. determinants for which the spin orbitals lower in energy than the active space are completely occupied and those higher in energy are completely unoccupied.

SINGLESBIAS [SinglesBias] Default 1.D0

This represents the factor to which singles are biased towards over double excitations from a determinant. This works with the NONUNIFORMRANDEXCITS excitation generators for FCIMC code. Normally, the pDoubles is given by the number of doubles divided by the total excitations from HF. Now, the number of singles in the total excitations term is multiplied by SinglesBias. Alternatively, SinglesBias can be set to less than 1 to bias towards doubles.

FINDGROUNDDET Default=.false.

A parallel FCIMC option. If this is on, then if a determinant is found with an energy lower than the energy of the current reference determinant, the energies are rezeroed and the reference changed to the new determinant. For a HF basis, this cannot happen, but with rotated orbital may be important.

DEFINEDET [DefDet(NEl)] Default=.false.

A parallel FCIMC option. This allows the reference determinant to be chosen based on that specified in the input with this keyword - rather than the default HF determinant chosen according to the energies of the orbitals. The determinant is specified by a series of NEl integers (separated by spaces) which represent the occupied spin orbitals.

DIRECTANNIHILATION Default=.false.

A parallel FCIMC option. This annihilation algorithm has elements in common with rotoannihilation and the default annihilation, but should be faster and better scaling than both of these, with respect to the number of processors. There are no explicit loops over processors, and newly-spawned particles are sent directly to their respective processors.

ANNIHILATEEVERY [iAnnInterval] Default=1

A parallel FCIMC option which can only be used with default annihilation algorihtm. This will mean that annihilation will only occur every iAnnInterval iterations.

ANNIHILATDISTANCE [Lambda] Default=0.D0

A serial FCIMC option. Here, walkers i and j have the chance to annihilate each other as long as they are on connected determinants. They will annihilate with probability given by -Lambda*Hij*(Si*Sj). This is hoped to increase annihilation and allow fewer particles to be needed to sample the space correctly. When Lambda=0.D0, it should be equivalent to the original annihilation algorithm. Warning - this is much slower than normal annihilation.

ANNIHILATERANGE [OFF] Default=.true.

A parallel FCIMC option. This is a slightly different annihilation algorithm, where only one sort of the full set of particles is needed. This should greatly reduce the time needed for annihilation of large numbers of particles. However, the load-balancing across processors may not be so good. This option is now on by default and can only be switched off via the input file by specifying **OFF** after the keyword.

LOCALANNIHIL [Lambda]

A parallel FCIMC option. An additional diagonal death rate is included at the annihilation stage for particles which are only singly occupied. The probability of death is given by Tau*EXP(-Lambda*ExcitDensity) where ExcitDensity is the approximate density of particles in the excitation level of the particle. This should raise death through this local annihilation, and hence keep the shift at a more resonable value in the undersampled regime. This will hopefully mean that a more accurate energy value can be obtained by removing the random killing of particles which arises from such a low shift value.

This is now commented out in the code

UNBIASPGENINPROJE Default false

An FCIMC serial option. Here, the acceptance probabilities are not unbiased for the probability of generating the excitation. Instead, the unbiasing occurs when the walker contributes to the energy estimator. This should reduce the variance for the energy estimator.

GLOBALSHIFT OFF Default true

This option can only be turned off by specifying **OFF**

A parallel FCIMC option. It is generally recommended to have this option on. This will calculate the growth rate of the system as a simple ratio of the total walkers on all processors before and after update cycle, rather than a weighted average. This however is incompatable with culling, and so is removed for update cycles with this in. This should be more stable than the default version and give a more reliable shift estimator for large systems.

MAGNETIZE [NoMagDets] [BField] Default false

This is a parallel FCIMC option. It chooses the largest weighted MP1 components and records their sign. If then a particle occupies this determinant and is of the opposite sign, it energy, i.e. diagonal matrix element is raised by an energy given by BField. First parameter is an integer indicating the number of determinants to 'magnetize', and the second is a real giving the amount the energy of a particle should be raised if it is of an opposite sign.

MAGNETIZESYM [NoMagDets] [BField] Default false

A parallel FCIMC option. Similar to the MAGNETIZE option (same arguments), but in addition to the energy being raised for particles of the opposite sign, the energy is lowered by the same amount for particles of 'parallel' sign.

GRAPHSIZE [NDets] In ResumFCIMC, this is the number of connected determinants to form the graph which you take as your sumsystem for the resummed spawning. Must have an associated RhoApp.

HAPP [HApp] Default 1.

In ResumFCIMC, this indicates the number of local applications of the hamiltonian to random determinants before the walkers are assigned according to the resultant vector.

NOBIRTH Force the off-diagonal H matrix elements to become zero, and hence obtain an exponential decay of the initial populations on the determinants, at a rate which can be exactly calculated and compared against.

This is no longer functional, but commented out in the code.

MCDIFFUSE [Lambda] Default 0.0.

Set the amount of diffusion compared to spawning in the **FCIMC** algorithm.

This is no longer functional and commented out in the code.

FLIPTAU [FlipTauCyc] Default: off.

Cause time to be reversed every FlipTauCyc cycles in the **FCIMC** algorithm. This might help with under-sampling problems.

This is no longer functional and commented out in the code.

NON-PARTCONSDIFF Use a seperate partitioning of the diffusion matrices, in which the antidiffusion matrix (+ve connections) create a net increase of two particles.

This is no longer functional and commented out in the code.

FULLUNBIASDIFF Fully unbias for the diffusion process by summing over all connections.

This is no longer functional and commented out in the code.

NODALCUTOFF [NodalCuttoff] Constrain a determinant to be of the same sign as the MP1 wavefunction at that determinant, if the normalised component of the MP1 wavefunction is greater than the NodalCutoff value.

This is no longer functional and commented out in the code.

NOANNIHIL Remove the annihilation of particles on the same determinant step.

REGENDIAGHELS Default .false. This is a parallel FCIMC option, which means that the diagonal hamiltonian matrix element for each particle is calculated on the fly, rather than stored with the particle. This will free up more memory, but will probably lead to slightly slower calculations.

REGENEXCITGENS This option will regenerate the excitation generator for each particle, every time a new random excitation is wanted. This is MUCH slower for the same number of particles (10x?). However, this frees up a lot more memory to store more particles.

PRINTDOMINANTDETS [NoDeterminants] [MinExcLevel] [MaxExcLevel] Default=.false.

This is a parallel FCIMC option. With this keyword, at the end of a calculation a DOMINANTDETS file is printed containing the NoDeterminants most populated determinants between excitation levels of MinExcLevel and MaxExcLevel (inclusive). This must be used with rotoannihilation.

PRINTDOMSPINCOUPLED [OFF] Default=.true.

This a parallel FCIMC option to go with the one above. It takes the list of dominant determinants chosen based on their populations and adds to the list all the spin coupled determinants that are not already there. This prevents any spin contamination when we truncate the available determinants. This is automatically on, but can be turned off using this keyword followed by OFF.

SPAWNDOMINANTONLY Default=.false.

This is a parallel FCIMC option. It takes a DOMINANTDETS file (printed using the above keywords) and reads it in at the beginning of the calculation. During the calculation, if a walker is to be spawned with an excitation level of those printed in DOMINANTDETS, this is only allowed if the determinant is in the list of dominant determinants. This does not allow truncation of the doubles, and must be used with rotoannihilation.

STARMINORDETERMINANTS Default=.false.

This is a parallel FCIMC option. It goes along with the **SPAWNDOMINANTONLY** keyword. If this is present, spawning to determinants not in the dominant list is done with a star approximation. That is, spawning onto minor determinants is allowed, but these walkers are only allowed to spawn back to the parent from which they came. The walkers undergo death and annihilation like usual (however, the walkers for annihilation are chosen randomly as they differ depending on their parent).

FINDGUIDINGFUNCTION [iGuideDets] Default=.false. [100]

This is a parallel FCIMC option. At the end of a spawning calculation, the iGuideDets most populated determinants are found and these and their final walker populations (with sign) are printed out (in order of their bit strings) to a file named GUIDINGFUNC to be used in the subsequent calculation.

USEGUIDINGFUNCTION [iInitGuideParts] Default=.false.

This is a parallel FCIMC option. This option takes the GUIDINGFUNC file produced in a previous calculation and reads in the guiding (or annihilating) function from it. The population on the HF determinant in this guiding function is then set to be iInitGuideParts, and the remaining determinants are populated based on their occupations from the previous calculation (in GUIDINGFUNC) relative to the HF determinant. The function of this guiding function is then to sit in the background of a calculation, able to annihilate walkers, but unable to itself spawn of have its walkers die. Assuming the GUIDINGFUNC from the previous calculation has the correct nodal structure, this guiding function should serve to instantly remove walkers spawned with the incorrect sign.

TRUNCATECAS [OccCASOrbs] [VirtCASOrbs] This is a parallel FCIMC option, whereby the space will be truncated according to the specified CAS. The arguments indicate the active electrons, and then the number of active virtual orbitals. These values can be dynamically updated throughout the simulation via use of the CHANGEVARS facility.

TRUNCINITIATOR This is a parallel FCIMC option. The keyword requires an initiator space to first be defined (usually via **TRUNCATECAS**, but could be by **EXCITE**). This is then a variation on a kind of CAS-star approach. Spawning is subject to the contraint that walkers spawned from determinants outside the active space only live if they are being spawned onto determinants that are already occupied. If walkers spawned on a new determinant have non-initiator parents, these spawns are 'aborted'. A special case is if in the same iteration walkers are spawned on a new determinant both from inside and outside the active space - in this case we treat the active space to have spawned a second earlier, the determinant is then treated as occupied and the non-active space walkers are allowed to live (providing they are the same sign of course). NOTE: This is currently only possible using **DIRECTANNIHILATION**.

DELAYTRUNCINITIATOR [IterTruncInit] This goes with the above. This allows us to first start with an active space only calculation and then at some iteration (given by IterTruncInit), to expand to the **TRUNCINITIATOR** method. The beginning of the **TRUNCINITIATOR** method may also be started dynamically by putting TRUNCINITIATOR in the CHANGEVARS file. At the moment, when this happens, tau is also reduced by a factor of 10. This should maybe be played with at some stage though.

KEEPDOUBSPAWNS This keyword goes along with the above **TRUNCINITIATOR**. This is an extra exception which means that if two determinant spawn on the same determinant with the same sign, they are allowed to live no matter where they came from. This is different from the original case where if both of these had come from non-initiator determinants, they would have both been killed.

ADDTOINITIATOR [InitiatorWalkerNo] This is again an addition to the above few options. In this case, if a determinant outside the initiator space builds up a significant population (greater than InitiatorWalkerNo), it is treated as being in the initiator space and may spawn on occupied or unoccupied determinants as it likes. This is reassessed at each iteration however, so determinant may move in and out of the initiator space as the populations vary.

INCLDOUBSINITIATOR This keyword also goes with **TRUNCINITIATOR**, and is a parallel FCIMC option. When it is present, all doubly excited determinants are included in the initiator space, and are allowed to spawn as usual.

The following option are only available in **MCSTAR** calculations:

BINCANCEL This is a seperate method to cancel down to find the residual walkers from a list, involving binning the walkers into their determinants. This has to refer to the whole space, and so is much slower. See also

the **WAVEVECTORPRINT** and **POPSFILE** options in the **LOGGING** block.

STARORBS [iStarOrbs] [NORETURN | ALLSPAWNSTARDETS] Default=.false. , NORETURN = OFF

A parallel FCIMC option. Star orbs means that determinants which contain these orbitals can only be spawned at from the HF determinant, and conversly, can only spawn back at the HF determinant. iStarOrbs is the integer variable which decides how many orbitals are in this high- energy space, and take the iStarOrbs number of highest energy orbitals to construct it. **NORETURN** is an optional keyword specifier. If it is specified, then any excitations from the HF to these high-energy determinants (doubles) are left to die and cannot respawn back to the HF determinant. **ALLSPAWNSTARDETS** is another optional keyword, which means that all particles can spawn at determinants with star orbitals, and once there, annihilation can occur. However, they cannot respawn anywhere else and are left there to die.

EXCITETRUNCSING [iHightExcitsSing] Default=.false.

This is a parallel FCIMC option, where excitations between determinants where at least one of the determinants is above iHighExcitsSing will be restricted to be single excitations.

EXPANDFULLSPACE [iFullSpaceIter] Default=0

This is a parallel FCIMC option. When this is set, the space initially is truncated at excitation level of ICIlevel, set by the value of the EXCITE parameter, or the CAS space given by TRUNCATECAS. If EXPANDFULLSPACE is set, then the system will continue to be truncated, but will expand to the full space after iteration iFullSpaceIter. Hopefully expanding the space in this way will allow quicker convergence, without needing to do this dynamically through the use of CHANGEVARS which may be difficult for long/queued jobs.

INITAMPLITUDE dInitAmplitude

For CCMC Amplitude, this is the initial amplitude in the Hartree-Fock determinant, and normalization factor for the wavefunction. Default 1.0

CLUSTERSIZEBIAS dProbSelNewExcitor

For CCMC Amplitude, this is the probability that the cluster selection algorithm terminates after each addition of an excitor. Larger values will bias towards smaller cluster. Default 0.7 (Range 0-1)

NSPAWNINGS nSpawnings

For CCMC, this is the number of spawnings attempted from each cluster (unless **EXACTSPAWN** is specified). Default 1

SPAWNPROP For Amplitude CCMC use NSPAWNINGS as a total number of spawnings, and distribute them according to the Amplitudes of clusters.

5.5.4 Return Path Monte Carlo options

MAXCHAINLENGTH [CLMAX] Set the maximum allowed chain length before a particle is forced to come back to its origin.

RETURNBIAS [PRet] Set the bias at any point to spawn at the parent determinant.

5.5.5 Perturbation theory options

MPTHEORY [ONLY] In addition to doing a graph theory calculation, calculate the Moller–Plesset energy to the same order as the maximum vertex level from the reference determinant (e.g. with 2-vertex sum the MP2 energy is obtained, with 3-vertex the MP3 energy etc. Within the **VERTEX SUM** hierarchy, this will only work with **VERTEX SUM HDIAG**. In the **VERTEX MC** hierarchy, do a Moller–Plesset calculation instead of a path-integral one. Requires **HDIAG**, and **BIAS**=0.D0. Can be used without a **METHODS** section. If a **METHODS** section is needed to specify different numbers of cycles at each level, then **MCDIRECTSUM** must also be set, either in the main block of the **CALC**, or by using **VERTEX MCDIRECT** instead of **VERTEX MC**. Note that the MP2 energy can be obtained in conjunction with a **VERTEX STAR** calculation.

ONLY Run only a MP2 calculation. This is only available when compiled in parallel. The only relevant **CALC** options are the **EXCITATIONS** options: all other **CALC** keywords are ignored or over-ridden. No **LOGGING** options are currently applicable.

Whilst in principle integrals are only used once, this optimal algorithm is not currently implemented. The speed of a **CPMD**-based calculation thus benefits from having a **UMatCache** of non-zero size.

> **Warning:** It is currently assumed that the calculation is restricted.

EPSTEIN-NESBET Apply Epstein–Nesbet perturbation theory, rather than Moller–Plesset. Only works for **VERTEX SUM NEW** and **VERTEX SUM HDIAG** and only at the 2-vertex level.

LADDER Use ladder diagram perturbation theory, rather than Moller–Plesset. The energy denominator is $E_0 - E_I + |H_{0I}|^2$. Only works for **VERTEX SUM NEW** and **VERTEX SUM HDIAG** and only at the 2-vertex level.

MODMPTHEORY Perform a hybrid of Epstein–Nesbet and Moller–Plesset theory, which includes only the $\langle ij||ijket + \langle ab||abket$ terms in the denominator. Only works for **VERTEX SUM NEW** and **VERTEX SUM HDIAG** and only at the 2-vertex level.

5.5.6 Diagonalisation options

Options for performing a full diagonalisation in the space of the full basis of spin orbitals.

> **Warning:** This quickly becomes prohibitively expensive as system size increases.

ACCURACY [B2L] Desired level of accuracy for Lanczos routine.

BLOCK [ON OFF] Default off.

Determines whether the Hamiltonian is calculated for each block or not. This only works for **COMPLETE**.

BLOCKS [NBLK] Set number of blocks used in Lanczos diagonalisation.

COMPLETE Perform a full diagonalisation working out all eigenvectors and eigenvalues. if **HAMILTONIAN** is **OFF**, discard the eigenvectors and eigenvalues after having used them for calculation. Relevant options are **HAMILTONIAN** and **BLOCK**.

Note: When would it be advantageous to save the eigenvalues and -vectors are a diagonalisation?

EIGENVALUES [NEVAL] Required number of eigenvalues.

ENERGY Calculate the energy by diagonalising the Hamiltonian matrix. Requires one of **COMPLETE**, **LANCZOS**, or **READ** to be set.

Exact E(Beta) is printed out as:

$$\text{E(Beta)} = \frac{\sum_\alpha E_\alpha e^{-\beta E_\alpha}}{\sum_\alpha e^{-\beta E_\alpha}}$$

The result will, of course, change depending upon the symmetry subspace chosen for diagonalization for finite temperatures.

The diagonalization procedure creates a list of determinants, which is printed out to the DETS file.

The weight, w_i and weighted energy, $w_\text{i}\tilde{E}_\text{i}$ are also calculated for all NPATH determinants.

Note: **ENERGY** was documented twice in the INPUT_DOC file. This is not particularly helful...

I have (hopefully) combined them correctly.

JUSTFINDDETS

This is an option to be used in conjunction with **ENERGY** and exact diagonalization methods. If specified, the diagonalization routines will just enumerate all the determinants and will not try to form the hamiltonian or diagonalize it. No energy will therefore be found, but enumerating all the determinants can be useful for histogramming methods in FCIMC methods.

KRYLOV [NKRY] Set number of Krylov vectors.

LANCZOS Perform a Lanczos block diagonalisation on the Hamiltonian matrix.

Relevant parameters are **BLOCKS**, **KRYLOV**, **ACCURACY**, **STEPS** and **EIGENVALUES**.

READ Read in eigenvectors and eigenvalues of the Hamiltonian matrix from a previous calculation.

STEPS [NCYCLE] Set the number of steps used in the Lanzcos diagonalisation.

5.5.7 Graph morphing options

A new approach developed by George Booth. Take an initial starting graph and over many iterations allow the determinants contained within the graph to change, so that the resultant graph is a better approximation to the true ground state.

GRAPHBIAS [GraphBias] If at each iteration the graph is being completely renewed, then this bias specifies the probability that an excitation of the previous graph is selected to try and be attached, rather than one of the determinants in the previous graph.

GRAPHSIZE [NDets] Specify the number of determinants in the graph to morph.

GROWGRAPHSEXPO [GrowGraphsExpo] Default is 2.D0.

The exponent to which the components of the excitations vector and the eigenvector are raised in order to turn them into probabilities. The higher the value, the more that larger weighted determinants will be favoured, though this might result in the graph growing algorithm getting stuck in a region of the space.

GROWINITGRAPH Grow the initial graph non-stochastically from the excitations of consecutive determinants.

INITSTAR Set up the completely connected two-vertex star graph, and use as the starting point for the morphing.

Automatically changes the NDets parameter to reflect the number of double excitations in the system.

ITERATIONS [Iters] The number of graph morphing iterations to perform.

MAXEXCIT [iMaxExcitLevel] Limit the size of the excitation space by only allowing excitations out to iMaxExcitLevel away from HF reference determinant.

MCEXCITS [NoMCExcits] Stochastically sample the space of excitations from each determinant in the graph with NoMCExcits determinants chosen per determinant. For the FCIMC code, this represents the number of attempted spawns per iteration in the spawning step.

MOVEDETS [NoMoveDets] Grow the graphs using an alternative Monte Carlo, where a number of determiants are deleted from the previous graph and reattached elsewhere in the graph in a stochastic manner, according to the probabilities given by the application of the rho propagator to the eigenvector of the previous graph.

NOSAMEEXCIT Ignore the connections between determinants which are of the same excitation level in comparison to the reference determinant. Currently only available in conjunction with **INITSTAR**, so the starting graph is simply the doubles star graph (with no cross connections).

ONEEXCITCONN Grow the graph by attaching only determinants which differ by one excitation level to the connecting vertex in the previous graph. Currently not implemented with MoveDets.

SINGLESEXCITSPACE Restrict the space into which the current graph is allowed to morph to just single excitations of the determinants in the current graph. This should reduce the scaling of the algorithm.

5.5.8 Monte Carlo options

Options for performing a Monte Carlo calculation on a vertex sum (as specified in the **METHODS** section).

The Monte Carlo routines have only ever been tested for molucular and model systems and probably are not currently functional for **CPMD** or **VASP** based calculations.

See the reports by Ramin Ghorashi ([RGPtIII]) and George Booth ([GHBCPGS]).

CALCVAR Only available for performing full vertex sums using the **HDIAG** formulation to evaluate the thermal density matrix elements.

Calculate a theoretical approximation to the expected variance if a non-stochastic MC run were to be performed, with the parameters given, at the chosen vertex level. Currently the expected variance is sent to STOUT as a full variance for the total energy ratio. Causes the calculation to take longer since the generation probabilities of the graphs must all be calculated. The sum over graphs of the generation probabilities is also printed out for each vertex level. This should equal 1, since we are working with normalised probabilities.

POSITION [IOBS JOBS KOBS] Sets the position of the reference particle.

CIMC Perform a configuration interation space Monte Carlo.

BETAEQ [BETAEQ] Default is set to be β, as set above.

Set β to have a different value for the equilibriation steps.

Note: What are the equilibriation steps?

BIAS [G_VMC_FAC] Default 16.

Vertex level bias for **FULL MC**. Positive values bias toward larger graphs, negative values towards smaller graphs.

For **SINGLE** and **MULTI** level MC (using a composite 1-vertex graph containing a full sum previously performed), this is the probability of generating a graph which is not the composite graph. The default is invalid, and this must be set manaully. Stochastic time MC is used. If BIAS is negative, then | BIAS | is used, but stochastic-time MC is not performed.

Note: BIAS seems to do two very different things if it is set to a negative value. Please clarify.

DETSYM [MDK(I), I=1,4] The symmetry of the **CIMC** determinant.

Note: Specify the symmetry how?

Note: If any if the **CIMC** options are set without **CIMC** being specified, the code will return an error and exit.**

EQSTEPS [IEQSTEPS] The number of equilibriation sets for the CI space Monte Carlo routine.

GRAPHEPSILON [GRAPHEPSILON] Default 0.0.

The minimum significant value of the weight of a graph.

Ignore the contributions to the weight and $\tilde{E}$ of all graphs with a weight that is smaller in magnitude than GRAPHEPSILON.

IMPORTANCE [G_VMC_PI] Ddefault 0.95.

Set the generation probability for the MC routine. This is the probability that new determinants are excitations of the pivot, i.

MCDIRECTSUM Perform Monte Carlo on graphs summing in energies weighted with the weight/generation probability of the graph.

PGENEPSILON [PGENEPSILON] Default 0.0.

Set the minimum significant value of the generation probability of a graph.

Because for larger graphs, the calculation of the generation probability is subject to numerical truncation errors, generation probabilities which are lower than a certain value are unreliable, and can cause the Monte Carlo algorithm to get stuck: if a graph had a very small generation probability, it would be difficult for a Monte Carlo run to accept a move to a different graph. If the magnitude of the generation probability of a graph is smaller than PGENEPSILON, then a new graph is generated.

Setting this too high could cause problems in the graph generation phase, so NECI will exit with an error if it generates 10000 successive graphs each with generation probabilities below PGENEPSILON.

SEED [G_VMC_SEED] Default -7.

Set the random seed required for the Monte Carlo calculations.

STEPS [IMCSTEPS] Set the number of steps for the CI space Monte Carlo routine.

VVDISALLOW Disallow V-vertex to V'-vertex transitions in stochastic time Monte Carlo: i.e. allow only transitions to graphs of the same size.

Weighting schemes

By default the vertex sum Monte Carlo algorithm selects excitations with no bias. The variance of a Monte Carlo calculation can be reduced by preferentially selecting for certin types of excitation.

EXCITWEIGHTING [g_VMC_ExcitFromWeight g_VMC_ExcitToWeight G_VMC_EXCITWEIGHT] [g_VMC_ExcitToW
Default 0.d0 (unweighted) for all values.

A weighting factor for the generation of random excitations in the vertex sum Monte Carlo. A parameter set to zero has a corresponding weighting factor of 1.

For generating an excitation from occupied spin orbitals i and j to unoccupied spin orbitals k and l:

- **the probability of choosing pair (ij) is proportional to**

$$e^{(E_i+E_j)\text{g\_VMC\_ExcitFromWeight}}.$$

- **the probability of choosing pair (kl) is proportional to**

$$e^{-(E_k+E_l)\text{g\_VMC\_ExcitToWeight}} e^{|\langle ij|U|kl\rangle|*\text{G\_VMC\_EXCITWEIGHT}} |E_i + E_j - E_k - E_l|^{\text{g\_VMC\_ExcitToWeight2}}.$$

POLYEXCITWEIGHT [g_VMC_ExcitFromWeight g_VMC_PolyExcitToWeight1 g_VMC_PolyExcitToWeight2 G_VMC_
Default 0.0 for all values (i.e. unweighted: all weighting factors are set to 1).

Weighting system for the choice of virtual orbitals in the excitations.

The probability of choosing the pair of spin orbitals, kl, to excite to is set to be constant for $E_k + E_l$ is less than g_VMC_PolyExcitToWeight1. For higher energy virtual orbitals, the weighting applied is a decaying polynomial which goes as:

$$(E_k + E_l - \text{g\_VMC\_PolyExcitToWeight1} + 1)^{-\text{g\_VMC\_PolyExcitToWeight2}}$$

g_VMC_PolyExcitToWeight1 is constrained to be not more than the energy of the highest virtual orbital.

POLYEXCITBOTH [g_VMC_PolyExcitFromWeight1 g_VMC_PolyExcitFromWeight2 g_VMC_PolyExcitToWeight1 g_VM
Identical to **POLYEXCITWEIGHT**, except that the polynomial weighting function applies also to the occupied orbitals. This means that there is another variable, since now the 'ExcitFrom' calculation also needs a value for sigma, and for the exponent. The sigma variables are now both under similar constraints as specified above, which means that they cannot be larger or smaller than the highest and lowest energy orbital respectivly. This prevents the PRECALC block from getting stuck, or from finding local variance minima.

Note: What is sigma?

CHEMPOTWEIGHTING [g_VMC_PolyExcitFromWeight2 g_VMC_PolyExcitToWeight2 G_VMC_EXCITWEIGHT]
Weighting is of the same form as POLYEXCITBOTH, but sigma is now constrained to be at the chemical potential of the molecule. Has only two parameters with which to minimise the expected variance.

CHEMPOT-TWOFROM [g_VMC_ExcitWeights(1) g_VMC_ExcitWeights(2) g_VMC_ExcitWeights(3) G_VMC_EXCITW

When choosing the electron to excite, use a a increasing polynomial up to the chemical potential and a decaying polynomial for spin orbitals above the chemical potential, in order to encourage mixing of the configurations around the HF state. Contains three adjustable parameters and testing needs to be done to see if this is beneficial. Expected to make more of a difference as the vertex level increases.

Note: What is the actual weighting form of **CHEMPOT-TWOFROM**?

UFORM-POWER New power form for the U-matrix element weighting using the appropriate **EXCITWEIGHT** element, which is believed to be better. This uses the form $W = 1 + |U|^{\text{EXCITWEIGHT}}$, rather than the exponential form.

STEPEXCITWEIGHTING [g_VMC_ExcitWeights(1) g_VMC_ExcitWeights(2) G_VMC_EXCITWEIGHT] This excitation weighting consists of a step function between the HF virtual and occupied electon manifold (i.e. step is at the chemical potential) When choosing an electron to move, the weight for selecting the electron is increased by 1 if the electron oribital has energy above the chemical potential and by g_VMC_ExcitWeights(1,1) if below. This occurs for both electrons. When choosing where to excite to, the situation is reversed, and the weight of selecting the unoccupied orbital is increased by 1 if the orbital is a hole in the occupied manifold and g_VMC_ExcitWeights(2,1) if a virtual orbital in the occupied manifold. Bear in mind that the parameters are NOT probabilities. If we are at a higher excitation level w.r.t. HF, then more electrons will be in the virtual manifold, which will alter the normalisation, and mean that when selecting electrons to excite, there will be an increasingly small probability of selecting them from the occupied manifold. The opposite is true when choosing where to put them.

Simply put, if the parameters are both < 1, then the biasing will preferentially generate excitations which reduce the excitation level.

U-weighting is the third parameter as before.

5.5.9 Experimental options

Note: More documentation on these options needed.

EXCITATIONS FORCEROOT Force all excitations in **VERTEX** [**SUM STAR**] **NEW** calculations to come from the root.

EXCITATIONS FORCETREE Disallow any excitations in a **VERTEX SUM NEW** which are connected to another in the graph, forcing a tree to be produced. Not all trees are produced however.

FULLDIAGTRIPS An option when creating a star of triples, to do a full diagonalisation of the triples stars, without any prediagonalisation. Very very slow...

LINEPOINTSSTAR [LinePoints] Set the number of excited stars whose eigenvalues are evaluated when using StarStars, in order to determine linear scaling.

NOTRIPLES Disallow triple-excitations of the root determinant as the 3rd vertex in **HDIAG** calculations at the third vertex level and higher.

5.6 Integrals

Note: INSPECT and ORDER options currently make no sense.

INTEGRAL Starts the integral block.

[Integral options—see below.]

ENDINT End of integral block.

5.6.1 General options

FREEZE [NFROZEN NTFROZEN] Set the number of frozen core states and frozen excited states respectively. Both must be a multiple of two - an error is returned if this is not the case. The Slater determinant space of a calculation does not include determinants which contain excitations from frozen core states or excitations to frozen excited states.

FREEZEINNER [NFROZEN NTFROZEN] Default=.false.[0 0] Allows orbitals to be frozen 'from the inside'. Meaning the NFROZEN occupied spin orbitals with the highest energy are frozen, along with the NTFROZEN lowest energy virtual spin orbitals. I.e. freezing from the fermi energy outwards. The main aim of this was to allow us to select an active space of HOMO and LUMO's, and freeze these to find the energy contained in the orbitals outside the active space.

PARTIALLYFREEZE [NPartFrozen NHolesFrozen] Sets the number of spin orbitals in the partially frozen core, and the maximum number of holes that are allowed within this core. Excitations which remove more than NHolesFrozen from the core are forbidden. This is a parallel FCIMC option.

This option may be changed dynamically using **PARTIALLYFREEZE** [NPartFrozen NHolesFrozen] in the CHANGEVARS file. The partially frozen core may be completely unfrozen in this way by simply setting the NHolesFrozen = NPartFrozen.

INSPECT [SPECDET(I), I=1,NEL-NFROZEN] Investigate the specified determinant.

ORDER [ORBORDER(I), I=1,8)] Set the prelimanary ordering of basis functions for an initial guess at the reference determinant. There are two ways of specifying open orbitals:

1. If orborder2(I,1) is integral, then if it's odd, we have a single.
2. Non-integral. The integral part is the number of closed oribtals, and the fractional*1000 is the number of open orbitals. e.g. 6.002 would mean 6 closed and 2 open which would have orborder(I,1)=6, orborder(I,2)=4 but say 5.002 would be meaningless as the integral part must be a multiple of 2.

5.6.2 Density fitting options

DFMETHOD [method] control the Density fitting method. Possible methods are:

DFOVERLAP (ij|u|ab)= (ij|u|P)(P|ab)

DFOVERLAP2NDORD (ij|u|ab)= (ij|u|P)(P|ab)+(ij|P)(P|u|ab)-(ij|P)(P|u|Q)(Q|ab)

DFOVERLAP2 (ij|u|ab)= (ij|P)(P|u|Q)(Q|ab)

DFCOULOMB (ij|u|ab)= (ij|u|P)[(P|u|Q)^-1](Q|u|ab)

where the sums over P and Q are implied.

All methods are precontracted to run in order(nBasis) except DFOVERLAP2NDORD.

DMATEPSILON DMatEpsilon (default 0) The threshold for density matrix elements, below which small density matrix elements are ignored, and conequently speeds up calculations.

5.6.3 Hartree–Fock options

The Hartree–Fock options have only been tested for molecular and model systems. They allow the Hartree-Fock orbitals (in the space of the original basis) to be used in a graph calculation instead of the original basis.

Note: James has never used these options. Please can those who have document them in more detail.

HF Use a Hartree–Fock basis.

CALCULATE Calculate the Hartree–Fock basis rather than reading it in. By default, the Hartree–Fock calculation is performed before any freezing of orbitals, i.e. in the full original basis.

HFMETHOD [HFMETTHOD] Default: **SINGLES**.

Specify the method for the Hartree-Fock routine. Options are:

STANDARD Use normal Hartree–Fock process.

DESCENT [SINGLES, OTHER] Use singles or other gradient descent.

MODIFIED Modify virtuals. Experimental.

MAXITERATIONS [NHFIT] Set the maximum number of Hartree–Fock iterations.

MIX [HFMIX] Set the mixing parameter for each Hartree–Fock iteration.

POSTFREEZEHF Do Hartree–Fock after freezing instead of before (still needs **HF** and **CALCULATE**). The Hartree–Fock calculation is performed only in the space of the unfrozen orbitals.

RAND [HFRAND] Default 0.01.

Set the maximum magnitude of the random numbers added to the starting density matrix. Use to perturb away from an initially converged Hartree–Fock solution.

READ [MATRIX BASIS] Read in U matrix and/or Hartree–Fock basis in terms of the original basis.

RHF Use restricted Hartree-Fock theory.

THRESHOLD [ENERGY [HFEDELTA] ORBITAL [HFCDELTA]] Set the convergence threshold for the energy and/or the orbitals.

UHF Use unrestricted Hartree-Fock theory.

5.6.4 Partioning options

If the weight and energy contribution from a graph are evaulated from diagonalising the ρ matrices, then various schemes are available to deal with the $e^{-\beta\hat{H}/P}$ operator.

Note: More detail on these needed.

FOCK-PARTITION For calculation of ρ operator with the Trotter approximation, partition the Hamiltonian according to the N-electron Fock operator and Coulomb perturbation.

FOCK-PARTITION-LOWDIAG For calculation of ρ operator with Trotter approximation, partition the Hamiltonian according to the N-electron Fock operator and coulomb perturbation. Take just the first order approximation (i.e. ignore the β/P term) for the diagonal terms of the ρ matrix.

FOCK-PARTITION-DCCORRECT-LOWDIAG For calculation of ρ operator with Trotter approximation, partition the Hamiltonian according to the N-electron Fock operator and Coulomb perturbation. Remove the Coulomb double counting in the Fock operator.Take just the first order approximation (i.e. ignore the β/P term) for the diagonal terms of the ρ matrix.

DIAG-PARTITION Default partitioning scheme.

For calculation of ρ operator with Trotter approximation, partition the Hamiltonian as the diagonal and non-diagonal matrix elements between the determinants.

RHO-1STORDER Calculate rho elements to only 1st order Taylor expansion (without applying a Trotter approximation).

5.6.5 VASP and CPMD options

There are too many 2-electron integrals to store for periodic systems (**CPMD** or **VASP** based calculations). Instead, as many integrals as possible are cached. Each four-index integral is reduced to two indices, A and B. Each A index has so many slots associated with it in which the integral involving A and B can be stored. The cache stores as many integrals as possible. If the cache is full and a new integral is calculated, then an element in the cache is over-written.

The efficiency of a calculation is heavily dependent on the size of the integral cache.

UMATCACHE [SLOTS] [nSlots] Default nSlots=1024.

Set the number of slots for each A index.

The total amount of memory used by the cache will be in the order of NSLOTS*NSTATES*(NSTATES-1)/2 words.

If nSlots=0, then disable caching of integrals calculated on the fly, but retain precomputation of 2-index 2-electron integrals ($\langle ij|ij\rangle$ and $\langle ij|ji\rangle$.

If nSlots=-1, no 2-electron integrals are stored.

Disabling the cache is very expensive.

The keyword **SLOTS** is optional and is present to contrast with the **MB** keyword.

UMATCACHE MB [MB] Number of megabytes to allocate to the UMAT cache. The number of slots is then set accordingly.

NOUMATCACHE Disable all UMAT caching (idential to **UMATCACHE** -1).

5.6.6 Experimental options

Note: Please document in more detail!

NRCONV [NRCONV] Default 10^{-13}.

This sets the convergence criteria for the Newton-Raphson algorithm in findroot. This takes place after initial bounds for the root are calculated using regular falsi (see above). Values smaller than 10^{-15} tend to create a fault since the Newton-Raphson algorithm cannot converge given the number of iterations allowed.

NRSTEPSMAX [NRSTEPSMAX] This sets the maximum number of Newton Raphson steps allowed to try and converge upon a root to the accuracy given in **NRCONV**. This is only applicable for the star graph, when trying to find the roots of the polynomial using **POLY NEW**, **POLY OLD** or **POLYCONVERGE**. Default value is 50.

RFCONV [RFCONV] Default 10^{-8}.

Set the convergence criteria for the Regular falsi algorithm in findroot. Only used with a star calculation which involves calculating the roots of a polynomial to find the eigenvalues. A Newton-Raphson convergence takes place after.

INCLUDEQUADRHO This changes the rho matrix for stars so that it includes the square of the eigenvalues - rho -> rho + rho^2/2. This is in an attempt to improve size consistency for the star graph. No change for large beta, and only very small changes for smaller betas.

EXPRHO The rho matrix is exponentiated, 1 is subtracted, and this is used as the matrix to be diagonalised. This is the full expansion for which **INCLUDEQUADRHO** is a truncation. Again, this is used to achieve size consistency with the star, although seems to have little effect, and no effect at high beta.

DISCONNECTNODES If using a nodal approximation, the connections between determinants in the same nodes are ignored - should then be equivalent to the original star calculation.

CALCEXCITSTAR Used with **STARSTARS**, it explicitly calculates each excited star and diagonalises them seperatly. This removes the approximation of cancelling ficticious excitations if the original star is used as a template for higher excitations. Scaling is bad, and all matrix elements have to be calculated exactly.

STARNODOUBS Only to be used with **CALCEXCITSTAR** when explicitly calculating excited stars, it forbids the excited stars to have excitations which are double excitations of the Hartree–Fock determinant.

STARQUADEXCITS Only to be used with **CALCEXCITSTAR**, when calculating the excited stars, it only allow the excited stars to have excitations which are quadruple excitations of the Hartree–Fock determinant.

QUADVECMAX Used with STARSTARS, it uses only the largest first element of the eigenvectors as the connection to each excited star. This means that for each excited star, only one connection is made back to the original star, meaning that the scaling is reduced. This seems to be a good approximation.

QUADVALMAX Same as QUADVECMAX, only the largest eigenvalue for each excited star is used. Seems to be little difference in results.

DIAGSTARSTARS Used with **STARSTARS**, it performs a full diagonalisation on each excited star, using the original star as a template, i.e. same excitations, and same offdiagonal elements. All that occurs is that the diagonal elements are multiplied by rho_jj. Large Scaling.

EXCITSTARSROOTCHANGE Used with **DIAGSTARSTARS** only at the moment, when this is set, only the root element of the excited star matrices changes when constructing excited stars with roots given by rho_jj. The remainder of the excited star matrix is identical to the original star matrix.

RMROOTEXCITSTARSROOTCHANGE Another option for use with **DIAGSTARSTARS**, when this is set, the same occurs as for **EXCITSTARSROOTCHANGE**, apart from the fact that the root is removed as an excited determinant in each excited star.

5.7 Logging

Note: All the logging options should say to which file they print output. Please correct this!

LOGGING Start the logging input block. The logging options allow additional (potentially expensive, potentially verbose) information to be printed out during a calculation. By default, all logging options are turned off.

[Logging options—see below.]

ENDLOG End the logging input block.

5.7.1 General options

FMCPR [LABEL, RHO, 1000, EXCITATION] More than one of the options can be specified.

Log the following to the PATHS file:

LABEL Logs the determinants contained by each graph as each determinant is generated in the format: $[(D_0),(D_1),...,D_v),]$ where each determinant given as a comma-separated list of the indices of the occupied orbitals: e.g. $D_0 =$ (1, 2, 9, 10,).

If CSFs are being used, then the CSF is printed. There is no newline after this.

For **MULTI MC** or **SINGLE MC**, only the non-composite graphs are printed.

EXCITATION

Log each graph in excitation format instead of full format above. The format is [A(i, j)->(a, b),B(k, l)->(c, d),...,C(m, n)->(e, f)]

where A, B,..., C are determinants in the graph from which the excitation is made. i, j,... are the orbitals within that determinant which are excited from, where (i<j, k<l,...). a, b,... are the orbitals they are excited to, where (a<b, c<d, ...).

This format does not in general provide a unique way of specifying multiply connected graphs, but the first possible determinant to which the next det in the graph is connected is chosen, so what is output should be unique. Single excitations are written as e.g. A(i, 0)->(a, 0).

RHO

Log the ρ matrix for each graph in the form: $(\rho_{11},\rho_{12},\cdots,\rho_{1v},|\rho_{21},\rho_{22},\cdots,\rho_{2v},|\cdots|\rho_{v1},\rho_{v2},\cdots,\rho_{vv},|)$,

where the graph consists of v vertices. A newline is appended.

XIJ Log the xij matrix, which contains the generation probabilities of one determinant in the graph from all the others. For MC this is already generated, but for full sums this must be generated, so will be slower. Generation probabilities are set with the **EXCITWEIGHTING** option.

The format is: $\{x_{11},x_{12},...,x_{1v},|...|x_{v1},x_{v2},...,x_{vv},|\}$

In general $x_{ij} \neq x_{ji}$. The x_{kk} element lists the number of possible excitations from k determinant. The matrix is followed by a newline.

After all these possible options, the following are printed: Weight [pGen] ETilde*Weight Class [Accepted]

pGen is only printed for: Monte Carlo, or if doing a full sum and the XIJ logging option is set. Accepted is only printed for Monte Carlo calculations. A newline is placed at the end of this data. For Monte Calo calculations, the values printed depend on the options. If **LABEL** is set, then all generated graphs and their values are printed, otherwise only values of accepted graphs are printed.

CALCPATH [LABEL RHO] Log CALCPATH_R to PATHS. Either just label logging or also log the ρ matrix, with the same format as above.

HAMILTONIAN Log HAMIL, PSI and PSI_LONG.

HFBASIS Log HFBASIS.

HFLOGLEVEL [LEVEL] Default 0.

If LEVEL is set to be positive, the density matrices, fock matrices and eigenvectors during a Hartree–Fock calculation are printed out to SDOUT.

MCPATHS Log MCPATHS data to the MCPATHS file for full vertex sum and MCSUMMARY file when using a METHODS section. Also log to the RHOPII file.

PSI Log PSI_COMP.

TIMING [iGlobalTimerLevel | LEVEL iGlobalTimerLevel | PRINT nPrintTimer]

LEVEL iGlobalTimerLevel Default 40. Timing information is only recorded for routines with level less than or equal to iGlobalTimerLevel. Less than 10 means general high level subroutines. Greater than 50 means very low level. Routines without a level are always timed (which is most of them). The greater the value of iGlobalTimerLevel, the more routines are timed. This can affect performance in some cases.

PRINT nPrintTimer Default 10. Print out timing information for the nPrintTimer routines which took the longest time.

XIJ Synonym for **FMCPR XIJ**.

5.7.2 FCIMC options

HISTSPAWN [iWriteHistEvery]

This option will histogram the spawned wavevector, averaged over all previous iterations. It scales horrifically and can only be done for small systems which can be diagonalized. It requires a enumeration of all determinants initially to work. It can write out the average wavevector every iWriteHistEvery. If a diagonalization option is set, SymDets will also be written out, containing the exact wavevector in the same format from the diagonalization.

HISTPARTENERGIES [BinRange] [iNoBins] [OffDiagBinRange] [OffDiagMax]

This is a histogramming option. It is slow, so not for use unless the diagnostic is needed. It will histogram the diagonal hamiltonian matrix element for three types of particle. Two input values are needed. The first argument is a real value to give the width of the histogram bin. The second is the number of bins needed (integer). Three histograms are produced: EVERYENERGYHIST - this is the histogram over all iterations of every particle in the system. ATTEMPTENERGYHIST - this is the histogram of the energy of all attempted spawned particles (including the ones which are successfully spawned). For this one, the contibution to the energy is actually 1/Prob of generating. SPAWNENERGYHIST - this is the histogram of all successfully spawned particles. All these histograms are normalized to one before printing out. Also now, the off-diagonal matrix elements are histogrammed. OffDiagBinRange is a real input parameter which indicates the range of the bins and OffDiagMax is the maximum matrix element to histogram. The doubles and singles will be done seperately, as

are the accepted spawns and total spawns. Therefore, four files are produced - SINGLESHIST, ATTEMPTSINGLESHIST, DOUBLESHIST, ATTEMPTDOUBLESHIST. Again, these are normalized and the ATTEMPT files histogram proportionally to 1/probability of generating the excitation.

AUTOCORR [NoACDets(2)] [NoACDets(3)] [NoACDets(4)] This is a parallel FCIMC option. It will output the histogrammed occupation number for certain determinants every iteration. This is so that a seperate standalone ACF program can be used on it. Currently the histogramming is evaluated for the HF determinants by default, but can also histogram determinants from other excitation levels. Firstly, it will calculate the 'NoACDets(2)' largest-weighted MP1 components (double excitations). It will then take the largest weighted double and do a new MP1 calculation with it as the root. It will then histogram the 'NoACDets(3)' largest weighted triple excitations, and the 'NoACDets(4)' largest quadruple excitations from this calculation to also histogram.

REDUCEDPOPSFILE [iWritePopsEvery] [iPopsPartEvery] This works in the same way as the normal popsfile, but only every iPopsPartEvery particle is printed out.

POPSFILE [iWritePopsEvery] Default: on. Default iWritePopsEvery (optional argument) 100000. Print out the determinants every iWritePopsEvery Monte-Carlo cycles. iWritePopsEvery should idealy be a multiple of **STEPSSHIFT**, the number of cycles between updates to the diagonal shift performed in the **FCIMC** calculation, to make sure the start of the next simulation follows smoothly

A calculation can then be restarted at a later date by reading the determinants back in using **READPOPS** in the **CALC** section. Walker number can also be scaled up/down by using **SCALEWALKERS**. If the iWritePopsEvery argument is negative, then the POPSFILE is never written out, even at the end of a simulation. This is useful for very large calculations where the POPSFILE will take a long time to write out and use a lot of disk space.

BINARYPOPS This means that the popsfile (full or reduced) will now be written out in binary format. This should now take up less disk space, and be written quicker. It can be read in as normal without specifying any extra criteria. Two files will be produced, a formatted file with the header info and a POPSFILEBIN with the walker information.

ZEROPROJE This is for FCIMC when reading in from a POPSFILE. If this is on, then the energy estimator will be restarted.

WAVEVECTORPRINT This is for Star FCIMC only - if on, it will calculate the exact eigenvector and values initially, and then print out the running wavevector every WavevectorPrint MC steps. However, this is slower.

PRINTFCIMCPSI This works for parallel FCIMC. This will enumerate all excitations (up to the truncation level specified, or the full space if not specified), and then histogram the spawning run, writing out the final averaged wavefunction at the end.

HISTEQUILSTEPS [NHistEquilSteps] Default=.false. [0] This works when the evolving wavefunction is to be histogrammed (for example using the above **PRINTFCIMCPSI** option, or the **USECINATORBS** orbital rotation option). This sets the histogramming to only begin after NHistEquilSteps iterations. This is so that the fluctuation populations at the beginning of a calculation may be left out.

PRINTORBOCCS Default=.false. This turns on the histogramming of the determinant populations, and at the end of the spawning, calls a routine to add up the contribution of each orbital to the final wavefunction. A ORBOCCUPATIONS file is then printed containing the orbitals and their normalised absolute occupations.

WRITEDETE [NoHistBins] [MaxHistE] This is an FCIMC option and will write out a histogram of the energies of determinants which have had particles spawned at them and their excitation level. The histogram logs the total amount of time spent at a determinant and its energy for each energy range. This is diagnostic information. The first variable to input is the number of histogram bins which will be calculated, and the second is the maximum determinant energy of the histogram.

PRINTTRICONNECTIONS [TriConMax] [NoTriConBins] This is a parallel FCIMC option. It looks at sets of connected determinants i,j and k. A sign coherent triangular connection is one where walkers spawned all around the triangle return to the original determinant with the same sign. Sign incoherent connections are those where the sign is reversed. If this option is on, two files are printed. TriConnTotal monitors the number of sign coherent and sign incoherent triangles over the course of the simulation, as well as the sum of the Hij x Hik x Hjk values, and the ratios for each. (The ratios are coherent / incoherent). TriConnHist

prints out a histogram of the Hij x Hik x Hjk values for coherent (col 1 and 2) and incoherent (col 3 and 4) triangles. The histogram goes from 0 -> +/- TriConMax with NoTriConBins for each.

HISTTRICONNELEMENTS [TriConHElSingMax] [TriConHElDoubMax] [NoTriConHElBins] This option histograms all the H elements involved in the triangular connections of determinants mentioned above. These are separated into doubles and singles, and an extra file, containing only the Hjk elements is also included. The histogram range is between +/-TriConHElSingMax for the singles and +/-TriConHElDoubMax for the doubles, with NoTriConHElBins bins for each. With this option, some stats are also printed in the output regarding the average magnitudes for each type of H elements.

PRINTHELACCEPTSTATS This option prints out a file (HElsAcceptance) containing information about the nature of the H elements resulting in accepted and not accepted spawns. This includes the number of not accepted spawns vs accepted, and the average size of the H element involved in accepted and not accepted spawns.

PRINTSPINCOUPHELS Default=.false. When attempting to spawn on a determinant i, this option finds the determinant j which is spin coupled to i, and prints out a set of stats relating to the sign and magnitude of the H element connecting i and j, Hij. These stats are printed in a file named SpinCoupHEl.

CCMCDEBUG iCCMCDebug Specify the CCMC debug level. Default 0 (no debugging information printed). Higher numbers will generate more information.

CCMCLOGTRANSITIONS [NONUNIQUE UNIQUE]

Do we log all transitions in CCMC. Very slow and memory intensive - only possible for extremely small systems. Default is **UNIQUE**. If **NONUNIQUE** is specified, then clusters with different orders are distinguished.

5.7.3 GraphMorph options

DISTRIBS Write out the distribution of the excitations in each graph as it morphs over the iterations. The first column is the iteration number, and then subsequent columns denote the number of n-fold excitations in the graph.

5.7.4 PRECALC options

PREVAR Print the vertex level, Iteration number, parameter, and expected variance, for each parameter which was searched for in the **PRECALC** block, showing the convergence on the optimum value, to the PRECALC file.

SAVEPRECALCLOGGING Allows different logging levels to be used in the **PRECALC** block than for the main calculation.

All logging options specified before **SAVEPRECALCLOGGING** are only used in the the **PRECALC** part of the calculation. All logging options specified after **SAVEPRECALCLOGGING** are only used in the the main part of the calculation.

5.7.5 Monte Carlo options

BLOCKING Perform a blocking analysis on the MC run. An MCBLOCKS file will be produced, which lists log(2)[blocksize], the average of the blocks, the error in the blocks(where the blocks are the energy ratio), and the full error, treating the energy estimator as a correlated ratio of two quantities.

ERRORBLOCKING [OFF] Default= ErrorBlocking.true. This can be used to turn off the error blocking analysis that is peformed by default on parallel FCIMC calculations. The default error blocking begins when the sum of the HF population over an update cycle reaches 1000. At the end of the simulation a BLOCKINGANALYSIS file is printed containing a list of block sizes with the resulting average of the projected energies calculated over an update cycle, the error in this energy and the error on the calculated error due to the block size.

BLOCKINGSTARTHFPOP [HFPopStartBlocking] Default=1000 This can be used to change the HF population that triggers the start of the error blocking analysis. Using this keyword over rides the default, and the blocking starts when the sum of the HF pop over an update cycle reaches HFPopStartBlocking.

BLOCKINGSTARTITER [IterStartBlocking] Default=.false. This can be used to set the error blocking to begin at iteration number IterStartBlocking, rather than a particular HF population.

The error blocking may also be initiated instantly by using **STARTERRORBLOCKING** in the CHANGEVARS file. Additionally, **PRINTERRORBLOCKING** will print the BLOCKINGANALYSIS file at that point, yet the calculation (and blocking) will continue (note - this file will be overwritten when the calculation ends and the final blocking stats are printed, so it must be renamed if it is to be kept). **RESTARTERRORBLOCKING** in the CHANGEVARS file zeroes all the blocking arrays and starts again from that point in the calculation.

SHIFTERRORBLOCKING [OFF] Default= ShiftErrorblocking.true. This can be used to turn off the default error blocking of the shift values. This only starts when the shift begins to vary, and may be restarted or the current SHIFTBLOCKINGANALYSIS file printed at that point using CHANGEVARS.

SHIFTBLOCKINGSTARTITER [IterShiftBlock] This can be used to specify the number of iterations after the shift is allowed to change that the shift error blocking begins.

VERTEX [EVERY n] Log the vertex MC with $\tilde{E}$ every n (real) cycles and/or log the vertex MC contribution every cycle. Setting Delta $= \tilde{E} - \tilde{E}_{\text{ref}}$, where $\tilde{E}_{\text{ref}}$ is usually the 1-vertex graph:

EVERY write a VMC file with the following info, with a new line each time the current graph changes:

tot # virt steps, # steps in this graph, #verts, Class, Weight, Delta, <sign(W)>, <Delta sign(W)>, ~standard deviation <Delta sign>/<sign>,pgen

n: write a VERTEXMC file with the following info:

0, #graphs, <sign(W)>, stdev(sign(W)), <Delta>, <sign Delta>/<sign>, <Delta^2>, acc ratio, trees ratio, nontree+ ratio, non-tree- ratio, <Delta sign(W)>, E~ reference, #sequences,w reference

Note: George, what are most of these values?

WAVEVECTORPRINT [nWavevectorPrint] Relevant only for Monte Carlo star calculations.

Calculate the exact eigen-vectors and -values initially, and print out the running wavevector every nWavevectorPrint Monte Carlo steps. This is slows the calculation down substantially.

5.7.6 Rotate Orbs Options

ROFCIDUMP [OFF] At the end of an orbital rotation (or in the case of a softexit), by default a ROFCIDUMP file will be printed using the transformation coefficients. This may then be read in to a spawning calculation. In the case of ROFCIDUMP OFF, no FCIDUMP will be printed. Note: When reading in the ROFCIDUMP, the number of electrons must be reduced by the number frozen in the previous rotation, and the number frozen set to 0.

ROHISTOGRAMALL If this keyword is present, two files are printed for all possible histograms. One labelled HistHF*, and one HistRot* containing the histogram before and after rotation. With this, certain histograms may be turned off by using the below keywords. Alternatively combinations of the keywords below may be used to just print a selection of the possible histograms.

ROHISTOFFDIAG [OFF] Histograms <ij|kl> terms before and after rotation where i<k and j<l.

ROHISTDOUBEXC [OFF] Histograms the 2<ij|kl>-<ij|lk> terms, the off diagonal hamiltonian elements for double excitations.

ROHISTSINGEXC [OFF] Histograms the single excitation hamiltonian elements.

ROHISTER [OFF] Histograms the <ii|ii> values before and after rotation.

ROHISTONEEIINTS [OFF] Histograms the one electron integral terms, <i|h|i>.

ROHISTONEPARTORBEN [OFF] Histograms the one particle orbital energies, epsilon_i = <i|h|i> + sum_j [<ij||ij>], where j is over the occupied orbitals only.

ROHISTVIRTCOULOMB [OFF] Histograms the two electron coulomb integrals <ij|ij> where i and j are both virtual spatial orbitals and i<j.

TRUNCROFCIDUMP [NoFrozenOrbs] This option goes along with the **USEMP2VDM** rotation option. Having diagonalised the MP2VDM matrix to get the transformation matrix. This option truncates the virtual orbital space by removing the NoFrozenOrbs SPIN orbitals with the lowest occupation numbers (MP2VDM eigenvalues). Only the remaining orbitals are transformed and included in the ROFCIDUMP that is printed. This kind of transformation requires different ordering of the orbitals to that which is standard for spawning calculation, so it is not possible to go straight from this rotation into a spawning calc. The ROFCIDUMP must be printed out then read back in.

WRITETRANSFORMMAT Default false. This keyword must be included if we are doing a natural orbital rotation, and we want to print out an MOTRANSFORM file. This file contains the transformation matrix in binary which can be used with Qchem to get the cube files for the new orbitals. NOTE: This file is only printed correctly if NECI is compiled using PGI when the file is printed.

CHAPTER

SIX

OUTPUT FILES

6.1 BLOCKS

BLOCKS contains the list of blocks used if the **BLOCK** option is used to calculate the Hamiltonian in the form:

```
BlockIndex   K(1)   K(2)   K(3)   MS SYM   nDets nTotDets
```

where:

BlockIndex starts from 1.

K(1:3) are momentum values for the **HUBBARD** and **UEG** symmetries and are irrelevant for other systems.

MS is $2S_z$.

Sym is the spatial symmetry index of the determinant. It can be a single number or a set of numbers enclosed in parentheses. Relevant for systems other than **HUBBARD** and **UEG**.

nDets is the number of symmetry unique determinants in the block.

nTotDets is the total number of determinants in the block.

6.2 CLASSPATHS

CLASSPATHS is calculated when a vertex sum is performed and lists properties of the graphs by their class:

```
Class nGs TotWeight   TotwEt TotWeightPos   TotWeightNeg
```

where:

Class is a binary string indicating the connectivity of the graph.

The connectivity matrix has 1 if there's a line in the graph e.g. a 3-vertex graph. The bits are labelled from the bottom right starting from 0:

```
(. 2 1 )  (bits)                  (. 1 1 )
(  . 0 )  -> 210     connections: (  . 0 ) -> 110 (binary) -> 6 (decimal)
(    . )                         (    . )
```

nGs is the number of graphs that fell in this class. TotWeight is the total weight, w_i, of those graphs. TotEt is the total value of $w_i\tilde{E}_i$ for those graphs. TotWeightPos is the total weight of graphs(?) with positive weights. TotWeightNeg is the total weight of graphs(?) with negative weights.

6.3 CLASSPATHS2

CLASSPATHS2 is calculated when a vertex sum is performed and gives a histogram of the weights for each graph type.

For each graph type, a header is printed out followed by the histogram data.

Header:

```
Class nGs   TotWeightPos   TotWeightNeg
```

Body:

```
log_10(weight)   nPos   nNeg
```

where:

Class, nGs, TotWeightPos, TotWeightNeg are the same as in CLASSPATHS.

The body consists of lines from -1 to -15 listing number of +ve and -ve graphs with a weight in that band. The top and bottom bands catch any overspills.

6.4 DETS

DETS contains a list of determinants when they are enumerated. This consists of two columns:

```
Determinant   Symmetry
```

where:

Determinant contains an ordered list of NEL numbers, separated by commas, surrounded by parentheses: e.g. (1, 2, 13, 14,).

Symmetry varies with the system type and is the internal symmetry label of the determinant. It is either an integer or a propogation vector which corresponds to an irreducible representation of an Abelian group.

6.5 ENERGIES

ENERGIES contains the list of eigenvalues in the order they are generated. If the Hamiltonian is **BLOCK** ed, then these will not all be in ascending order, but rather ascending order within each block.

6.6 HAMIL

HAMIL contains the non-zero elements of the Hamiltonian in three columns:

```
i  j  Hij
```

where:

Hij $= \langle D_\mathbf{i}|H|D_\mathbf{j}\rangle$.

i,j are indices for the $\mathbf{i}, \mathbf{j}$ determinants, with i:math:*le* j, and increasing i.

D_i corresponds to the i-th determinant as given in DETS.

6.7 MCPATHS and MCSUMMARY

The MCPATHS logging file (or MCSUMMARY if a **METHODS** block is used) has the following layout:

```
<Header Line>
<Vertex Sum Section for Det 1>
<Vertex Sum Section for Det 2>
...
<MC Summary information>
```

The Header Line is:

```
\"Calculating  XXX W_Is...\"
```

where XXX is the number of determints calculated, and the number of Vertex Sum sections.

Each Vertex Sum Section consists of:

```
(Determinant Calculated)
<method level 1 line>
<method level 2 line>
...
```

The Method level line is:

```
<level>  <weight>  <cumlweight>  <timing> <GraphsSummed>  [<PartGraphs>] <w E~> [<MP2 contrib> [<
```

where:

level The vertex level as specified by the METHODS section.

weight The contribution of this level to s_i (see RHOPII or RHOPIIex file section). This can be further analysed in CLASSPATHS{,2}.

cumlweight The sum of weights up to and including this level.

timing (double) The number of seconds calculating this level took.

GraphsSummed The total number of graphs summed together at this level.

PartGraphs Recursively, how many times FMCPR? is called. It is called once as each node is added to a graph.

$w\tilde{E}$ The contribution of this level to $w\tilde{E}$. This can be further analysed in CLASSPATHS{,2}

MPn contrib The contribution of graphs at this level to MPn theory.

There are is one method level line for each method level specified in the **METHODS** section, plus one for the 1-vertex graph.

The MC Summary information is split into 4 parts:

TotalStats:

```
GRAPHS(V)...WGHT-(V)
```

GenStats:

```
GEN-> *
```

AccStats:

```
ACC-> *
```

Sequences:

```
Sequences, Seq Len
```

TotalStats: The output is split into columns depending on the levels sampled in the Monte Carlo:

DataType Total 1-vertex 2-vertex 3-vertex ...

The DataTypes are:

GRAPHS(V) The number of graphs sampled with this number of vertices.

TREES(V) The number of trees sampled with this number of vertices. Trees contain no cycles.

NON-TR+(V) The number of non-trees sampled with this number of vertices whose weight is positive.

NON-TR-(V) The number of non-trees sampled with this number of vertices whose weight is negative.

WGHTT(V) The total weight of trees with this number of vertices.

WGHT+(V) The total weight of positive non-trees with this number of vertices.

WGHT-(V) The total weight of negative non-trees with this number of vertices.

GenStats: Statistics on the number of graph-graph transitions generated.

The columns correspond to the graph FROM which the transition was generated: DataType 1-vertex 2-vertex 3-vertex ...

The rows correspond to the graph TO which the transition was generated: GEN-> 1 GEN-> 2 ...

AccStats: Format as GenStats, but has the number of transitions accepted.

Sequences: Records the number of sequences of consecutive graphs accepted with the same weight.

6.8 RHOPII

RHOPII is produced during the calculation of a vertex sum. It contains:

```
Index   Determinant=Di   w_i   P ln rho_ii   ln s_i   E~_i   Degeneracy
```

where:

Index begins at 0.

Note: This is not true for the test jobs which are not model systems. What does Index mean?

Determinant is formed by the list of spin-obitals enclosed in parentheses.

w_i is the calcuated value of math:*w_{veci}=bra D_{veci} | e^{-beta H} | D_{veci} ket*.

P ln rho_ii is formed by P, the path length, and rho_ii is $\rho_{\mathbf{ii}} = \langle D_\mathbf{i}|e^{-(\beta/P)H}|D_\mathbf{i}\rangle$ calculaed to the approximation specified by the input parameters.

s_i is defined from $\ln w_i = P \ln \rho_{ii} + \ln s_i$.

E~_i is the value of $\tilde{E}_\mathbf{i} = \frac{\langle D_\mathbf{i}|He^{-\beta H}|D_\mathbf{i}\rangle}{w_i}$.

6.9 RHOPIIex

RHOPIIex is produced if a diagonalization is performed. The data is calculated by CalcRhoPII. It contains:

```
Determinant=Di   w_i   P ln rho_ii   ln s_i   E~_i   Degeneracy
```

where:

Determinant is formed by the list of spin-obitals enclosed in parentheses.

w_i is the calcuated value of $w_{\mathbf{i}} = \langle D_{\mathbf{i}}|e^{-\beta H}|D_{\mathbf{i}}\rangle$.

P ln rho_ii is formed by P, the path length, and rho_ii is $\rho_{\mathbf{ii}} = \langle D_{\mathbf{i}}|e^{-(\beta/P)H}|D_{\mathbf{i}}\rangle$ calculaed to the approximation specified by the input parameters.

s_i is defined from $\ln w_i = P \ln \rho_{ii} + \ln s_i$.

E~_i is the value of $\tilde{E}_{\mathbf{i}} = \frac{\langle D_{\mathbf{i}}|He^{-\beta H}|D_{\mathbf{i}}\rangle}{w_i}$.

Degeneracy is the number of symmetry related determinants which are not explicitly calculated.

CHAPTER

SEVEN

EXAMPLE INPUT FILES

7.1 Standalone MP2 calculations

The **CALC** block is simply:

```
Calc
    MPTheory only
EndCalc
```

7.1.1 CPMD

For a straight-forward **CPMD**-based calculation using, say, 200 spin-virtuals, the input file is:

```
System CPMD
EndSys

Calc
    MPTheory only
EndCalc

Integrals
    UMatCache MB 750
    Freeze 0,-200
EndInt
```

where we have also allocatabed a maximum of 750MB to be used in caching the integrals.

If we wish to ignore the single excitations of the reference (Kohn–Sham) determinant, then we can use:

```
System CPMD
EndSys

Calc
    MPTheory only
    Excitations doubles
EndCalc

Integrals
    UMatCache MB 750
    Freeze 0,-200
EndInt
```

7.1.2 Molecular systems

Similarly, for molecular systems, a valid input file is of the form:

```
System READ
    Electrons 36
EndSys

Calc
    MPTheory only
EndCalc
```

More to come.

BIBLIOGRAPHY

[AttenEx] Efficient calculation of the exact exchange energy in periodic systems using a truncated Coulomb potential, James Spencer and Ali Alavi, PRB, 77 193110 (2008).

[CamCasp] Cambridge package for Calculation of Anisotropic Site Properties, Alston Misquitta and Anthony Stone. http://www-stone.ch.cam.ac.uk/programs.html#Camcasp

[CPMD] CPMD, http://www.cpmd.org/, Copyright IBM Corp 1990-2008, Copyright MPI für Festkörperforschung Stuttgart 1997-2001.

[DALTON] DALTON, a molecular electronic structure program, Release 2.0 (2005), see http://daltonprogram.org/

[FCIQMC] Fermion Monte Carlo without fixed nodes: a Game of Life, death and annihilation in Slater Determinant space, G.H. Booth, A.J.W. Thom and Ali Alavi, J. Chem. Phys., 131, 054106, (2009).

[GHBCPGS] CPGS report, George Booth.

[Initiator] Survival of the Fittest: Accelerating convergence in full configuration-interaction Quantum Monte Carlo , Deidre Cleland, G.H. Booth, and Ali Alavi, J. Chem. Phys., 132, 041103, (2010).

[MolPro] MOLPRO is a package of ab initio programs written by H.-J. Werner, P. J. Knowles, R. Lindh, F. R. Manby, M. Schütz, P. Celani, T. Korona, G. Rauhut, R. D. Amos, A. Bernhardsson, A. Berning, D. L. Cooper, M. J. O. Deegan, A. J. Dobbyn, F. Eckert, C. Hampel, G. Hetzer, A. W. Lloyd, S. J. McNicholas, W. Meyer, M. E. Mura, A. Nicklaß, P. Palmieri, R. Pitzer, U. Schumann, H. Stoll, A. J. Stone, R. Tarroni, and T. Thorsteinsson, see http://www.molpro.net

[RGPtIII] Part III report, Ramin Ghorashi.

[StarPaper] Electron correlation from path resummations: the double-excitation star, Alex J. W. Thom, George H. Booth, and Ali Alavi, Phys. Chem. Chem. Phys., 10, 652-657 (2008).

[SumPaper] A combinatorial approach to the electron correlation problem, Alex J. W. Thom and Ali Alavi, J. Chem. Phys. 123, 204106, (2005).

[ThomPhDThesis] Towards a quantum Monte Carlo approach based on path resummations, Alex J.W. Thom, PhD Thesis (2006).

[TwoElBox] Two interacting electrons in a box: An exact diagonalization study, Ali Alavi, JCP 113 7735 (2000).

[VASP] VASP, http://cms.mpi.univie.ac.at/vasp/.