1 Møller-Plesset correlation corrections

1.1 Introduction

One important approximation made in the Hartree-Fock model is that, for each electron, the interaction of this electron with the other electrons in a system is simplified by considering only an average interaction. As a result, the energy of a system as obtained with the Hartree-Fock model is higher than the energy that would be obtained if all electron-electron interactions would individually be taken into account. This difference in energy is commonly referred to as the dynamic electron correlation energy.
One theory for recovering this dynamic correlation energy that is absent from the Hartree-Fock solution is Møller-Plesset perturbation theory. In Firefly, it is possible to perform second-order, third-order, and fourth-order Møller-Plesset energy corrections (commonly abbreviated as MP2, MP3, and MP4), where higher-order methods obtain more of the missing correlation energy than lower-order methods (but also require more computational resources). MP2 is implemented for RHF, UHF, and ROHF wavefunctions with analytic gradients available only with RHF. MP3 and MP4 theories on the other hand are implemented for RHF wavefunctions only and can only be used to obtain energies.
The MP2 code in Firefly has been optimized for performance to a great extent. MP2 performance is discussed in a section of its own, and it is recommended to read through this section before commencing with MP2 calculations.
Second-order Møller-Plesset theory is also implemented for MCSCF wavefunctions in the form of MRMP2 and (X)MCQDPT2 theory. This theory is discussed in the (X)MCQDPT2 chapter.

1.2 MP2 calculations

MP2 calculations can be requested by specifying MPLEVL=2 in $CONTROL, in combination with SCFTYP=RHF, UHF, or ROHF. Keywords that pertain to MP2 calculations are found in the $MP2 group. Additionally, keywords in the $MP2GRD group can be used to control the calculation of gradients.
There is little that has to be said with respect to RHF and UHF MP2 calculations. The most important thing is that the description of the system under investigation should already be good at the zeroth-order level (i.e. at the Hartree-Fock level). If this is not the case, it is probably better to use MCSCF theory and recover the correlation energy using (X)MCQDPT2.
One point which may not be commonly appreciated is that the density matrix for the first-order wavefunction for the RHF case, which is generated during gradient runs or if properties are requested in the $MP2 group (MP2PRP=.TRUE.), is of the type known as "response density", which differs from the more usual "expectation value density". The eigenvalues of the response density matrix (which are the occupation numbers of the MP2 natural orbitals) can therefore be greater than 2 for frozen core orbitals, or even negative values for the highest 'virtual' orbitals. The sum is of course exactly the total number of electrons. We have seen values outside the range 0-2 in several cases where the single configuration RHF wavefunction was not an appropriate description of

the system, and thus these occupancies may serve as a guide to the wisdom of using a RHF reference.

The case of ROHF MP2 deserves more explanation. There are a number of open shell perturbation theories described in the literature. It is important to note that these methods give different results for the second-order energy correction, reflecting ambiguities in the selection of the zeroth-order Hamiltonian and in defining the ROHF Fock matrices. Two of these are available in Firefly. One theory is known as RMP which, it should be pointed out, is entirely equivalent to the ROHF-MBPT2 method. This theory is as UHF-like as possible, and can be chosen by selection of OSPT=RMP in $MP2. The RMP method diagonalizes the alpha and beta Fock matrices separately so that their occupied-occupied and virtual-virtual blocks are canonicalized. This generates two distinct orbital sets whose double excitation contributions are processed by the usual UHF MP2 program, but an additional energy term from single excitations is required.

RMP's use of different orbitals for different spins adds to the CPU time required for integral transformations, of course. RMP is invariant under all of the orbital transformations for which the ROHF itself is invariant.

Unlike UHF MP2, the second-order RMP energy does not suffer from spin contamination, since the reference ROHF wavefunction has no spin contamination. The RMP wavefunction, however, is spin contaminated at 1st and higher order, and therefore the 3^{rd} and higher order RMP energies are spin contaminated.

The ZAPT (Z-averaged perturbation theory) formalism is also implemented in Firefly, and can be requested by specifying OSPT=ZAPT in $MP2. Because this theory is not spin contaminated at any order, and has only a single set of orbitals in the MO transformation, it is the default. It should be noted that, at present, the new MP2 code can only execute ROHF MP2 calculations of the RMP kind. ZAPT2 calculations should be run with old MP2 code (see the section on MP2 performance for more information on the new MP2 code).

There are a number of other open shell theories with names such as HC, OPT1, OPT2, and IOPT. These are not implemented in Firefly but equivalent results can be obtained by using GUGA-style MCSCF followed by (X)MCQDPT2. The same is true for GVB-based MP2. For example, one could use a $DRT input such as

```
NMCC=N/2-1 NDOC=0 NAOS=1 NBOS=1 NVAL=0
```

which generates a single CSF if the two open shells have different symmetry. For a one pair GVB function, one can specify

```
NMCC=N/2-1 NDOC=1 NVAL=1
```

which generates a 3 CSF function entirely equivalent to the two configuration TCSCF, also known as GVB-PP(1). And if one would attempt a triplet state with the GUGA MCSCF program

```
NMCC=N/2-1 NDOC=0 NALP=2 NVAL=0
```

one would get a result equivalent to the OPT1 open shell method instead of the RMP result. For details on $DRT input, please see the chapter on MCSCF.
A feature new to Firefly 8.0.0 is the possibility to scale the two MP2 spin components by certain factors. This functionality is controlled through the SCS keyword and is programmed for RHF, UHF, and ROHF (though only RMP, not ZAPT2). Currently, only energies are available. Input should be of the form:

```
$MP2 SCS(1)=singlet_pairs_multiplier,triplet_pairs_multiplier $END
```

As an example, Grimme's SCS-MP2 scheme can be specified as:

```
$MP2 SCS(1)=1.2,0.333333333333333 $END
```

while the SOS-MP2 scheme from Head-Gordon and co-workers can be specified as:

```
$MP2 SCS(1)=1.3,0 $END
```

Note that use of spin scaling forces the use of MP2 METHOD=1 code (see next section), which is faster than Laplace-transform based code even for SOS-MP2.

1.3 Performance of the MP2 code

Over the years, the MP2 code in Firefly has been great optimized greatly. Therefore, some information with respect to performance should be given.
One of the most important things to note is that, in comparison to the GAMESS(US) code on which Firefly is originally based, Firefly has a new RHF/ROHF/UHF MP2 energy program which is designed to handle large systems (e.g., 500 AOs or more). It is direct, very fast, and requires much less memory compared to other MP2 methods. Use of this new program can be requested by specifying METHOD=1 in $MP2 and is recommended for all medium and large jobs. For small jobs, the old method is still faster due to less overhead (the new program always requires the 2-electron AO integrals to be reevaluated four times, regardless of the size of the job). It is for this reason that the old method has remained the default one.
The memory requirements of the new program scale as approximately N2. This as opposed to the other MP2 programs currently implemented, which scale as at least N^3 for the segmented transformation and as $A \cdot N^2$ for the alternative integral transformation. Here, A and N are the number of active orbitals and the total number of MOs, respectively. Disk requirements of the new code scale as A^2N^2, whereas they scale as $A^2 \cdot (N-A)^2$ for the alternative integral transformation. The segmented transformation does not use temporary disk storage. Disk I/O is used, but to a very limited degree. Therefore, the CPU utilization is usually 90% or even better. The CPU utilization is usually less than 50% for other MP2 transformation methods working in the conventional mode, while, in the direct mode, there is a very serious overhead because

of the multiple reevaluation of 2-electron integrals.
Asymptotically, the FLOPs count with the new program is about a half or even better as compared to other MP2 energy transformation methods.
Then, it is important to discuss some specifics about the MP2 gradient code. Since version 7.1, Firefly has new and efficient semidirect MP2 gradient (and properties) program that is based on fastints code and runs in parallel using the P2P interface. Contrary to the new MP2 energy program, the new MP2 gradient program is used by default, so no additional keywords have to be specified in order to enable its use. Gradients are available only for RHF wavefunctions. Note that use of the gradient program automatically forces the use of the new MP2 energy program.
The memory demands of the new gradient program are quite modest: asymptotically, they are proportional to

$$N_{\mathrm{occ}} \cdot (N_{\mathrm{occ}} + N_{\mathrm{core}}) \cdot N_{\mathrm{vir}}$$

where N_{occ} is the number of active occupied orbitals (i.e., excluding frozen core orbitals), N_{core} is the number of frozen core occupied orbitals, and N_{vir} is the number of virtual orbitals. Parallel scalability is good, with most of the memory demands reducing linearly with the number of used computing nodes.
As for disk I/O, the new MP2 gradient program uses (large) temporary files to store half-transformed 2-electron integrals (in the DASORT file) and the non-separable part of the MP2 2-particle density matrix (DM2, stored in DAFL30 file during gradient runs only). Therefore, disk I/O can be quite intensive. When running in parallel, the contents of these files is evenly distributed across all nodes. If one is running this code in parallel on an SMP or multicore system, it is recommended to assign a separate physical disk to each Firefly process.
Unfortunately, the exact amount of disk space required to store files DASORT and DAFL30 cannot be predicted exactly because of two reasons. First, the Firefly uses sparsity of half-transformed integrals and DM2 to save disk space by storing only non-zero values. Second, half-transformed integrals, DM2 elements as well as their labels are further packed, and it is impossible to predict the exact packing ratio. However, it is possible to provide upper bounds on the overall maximum sizes of DASORT and DAFL30. Note, these bounds usually seriously overestimate the real size of these files!
Namely, the overall size of all DASORT files is less or equal than:

$$6 * N_{\mathrm{occ}} \cdot N_{\mathrm{virt}} \cdot N_{\mathrm{ao}} \cdot (N_{\mathrm{ao}} + 1) bytes$$

Similarly, the overall size of all DAFL30 files is less or equal than:

$$8 * N_{\mathrm{occ}} * N_{\mathrm{virt}} * N_{\mathrm{ao}} * (N_{\mathrm{ao}} + 1) bytes$$

Here, N_{occ} is the number of active occupied orbitals, N_{virt} is the number of virtual orbitals, and N_{ao} is the number of Cartesian atomic orbitals.
When using the new gradient program, it is recommended to use the following input:

```
$CONTRL INTTYP=HONDO $END
$P2P P2P=.T. DLB=.T. XDLB=.T. $END
$SMP CSMTX=.T. $END
$MP2 METHOD=1 $END
```

Depending on your operating system (e.g., under some Windows systems), the use of the following addition options:

```
$SYSTEM SPLITF=.T./.F. $END (large file splitting enabled/disabled)
```

and/or

```
$SYSTEM IOFLGS(30)=1 $END (activates file cache write through mode for unit # 30 which contains DM2 elements)
```

may have serious positive or negative impact on the overall performance as well.
The following set of parameters of the new MP2 gradient code seems to be optimal in the case you are running large problems in parallel on Linux-based SMP or multicore system which do not have a separate physical disks for each instance of the parallel Firefly processes.

```
$SYSTEM SPLIT=.F. $END
$SYSTEM IOFLGS(30)=0 $END
$MP2GRD DBLBF=.F. FUSED=.F. ASYNC=.T. $END
```

The following set of additional I/O parameters seem to be optimal for large-scale jobs running under Windows Vista and Windows Server 2008 R1:

```
$MP2 IOFLGS(1)=65536,65536 IOFLGS(3)=65536,65536 $END
```

These settings turn on direct unbuffered disk I/O for the files DASORT and DICTNRY and disable standard buffered disk reads and writes.
If the I/O remains a significant bottleneck, or if the amount of available disk space is not sufficient for carrying out an MP2 calculation, one can enable the direct evaluation of AO integrals. This is done by specifying DIRECT=.T. in $MP2, which will somewhat reduce the amount of disk space used at the cost of additional CPU time.
Finally, it should be noted that the new gradient code requires higher accuracy when there is a partial linear dependence in the AO basis set. We recommend the use of the following settings (tighter values can be used if desired):

```
$CONTRL ICUT=11 INTTYP=HONDO $END
$SCF NCONV=7 $END
$MP2GRD TOL1=1D-12 TOL2=1D-12 $END
```

1.4 MP3 and MP4 calculations

MP3 and MP4 type calculations can be requested by specifying MPLEVL=3 and MPLEVL=4 in $CONTROL in combination with SCFTYP=RHF. Related keywords can be found in the groups $MP3 and $MP4. As noted earlier, these calculations are only implemented for RHF wavefunctions - ROHF and UHF MP3/MP4 calculations are currently not possible. For MP4, it is possible to perform three types of calculations, namely MP4(SDQ), MP4(SDTQ), and MP4(T). The default MP4 type is MP4(SDQ).
MP3 and MP4(SDQ) calculations are multithreaded and cannot be executed in parallel mode as the corresponding code was written several years ago and was not allowed to run in parallel at the time. Despite this, the code is still quite efficient and scales well with increasing numbers of threads. The MP4(T) code on the other hand was written more recently, is more advanced, and is written to run in parallel, though it also benefits from multithreading. This makes performing an MP4(SDTQ) calculation in the most optimal way a bit complicated.
To explain this a bit more: it is perfectly possible to perform an MP4 calculation in a single job which can done by specifying

```
$MP4 SDTQ=.T. $END
```

However, in terms of computational time this is not the most optimal way. When run in parallel, the MP4(SDQ) part of the calculation cannot run in parallel and will therefore just be duplicated on all instances of Firefly. At the same time, multithreaded execution is also not optimal as the MP4(T) part of the calculation will not benefit as much from multithreading as it does from parallel execution. Also, the Hartree-Fock part of the calculation does not benefit from multithreading.
Because of this, it is advised to split an MP4(SDTQ) calculation up in three separate steps (i.e., jobs). The first step is to perform a parallel Hartree-Fock calculation. The option

```
$CONTRL WIDE=.T. $END
```

can hereby be used to punch the HF orbitals with double accuracy. The second step is then to perform a multithreaded MP4(SDQ) calculation, using the orbitals from the previous step as the initial guess. The third and final step is to perform an MP4(T) calculation (which can be requested using

```
$MP4 TONLY=.T. $END
```

again using the HF orbitals as the initial guess. As noted, the MP4(T) code can run parallel and multithreaded at the same time and it is advisable to do so. The optimal number of threads per process depends here on the particular computer architecture that is used. In most cases, it is equal to the number of physical cores sharing the same memory domain on the cc-NUMA system. E.g., assuming that a 2-way four-core Xeon 5500 system is used, the best way is to use 2 processes per box with four working threads each. However, this can

be adjusted to minimize I/O or scratch storage, and the code is flexible here.
The usage of Abelian symmetry can greatly decrease the required CPU time for any MP3 or MP4 calculations. Unlike MP2 case, where the speed up due to the use of symmetry is roughly proportional to the first power of the order of the symmetry group (Ng) used, in the case of MP3 and MP4 jobs, the speed up is proportional to Ng^2 on the average.
It finally should be noted that, in practice, there might not be much of a reason for performing MP3 calculations, as the computational cost of a large MP4(SDQ) calculation is typically only 2-3 times greater than that of a similar MP3 job. Also, the memory demands of both types of calculations are very similar. MP4(SDTQ) jobs are always much more demanding.