Integral Transformation and MP2 Energy Project

C. David Sherrill
School of Chemistry and Biochemistry
Georgia Institute of Technology

Created 12 July 2007

Goals: Write two computer programs. (1) A program to compute the MP2 energy of a closed-shell molecule, using one- and two-electron MO integrals read from PSI; and (2) a program to transform one- and two-electron integrals from the AO basis (actually symmetry orbital, or SO basis) to the MO basis. (2a) Adapt program (1) to read the integrals from your own integral transformation program.

Note: *This is a new set of notes, written quickly, and may contain errors. Sorry! Please give me any updates/corrections so they become easy to follow in the future.*

General Setup: Because we will interface with the PSI program, you will need to use the basic skeleton of a PSI program, which is reproduced below in Figure 1. It is likely that you will want to use function calls from the libraries `libipv1`, `libpsio`, `libciomr`, `libqt`, and `libchkpt`, so it is a good idea to include all the relevant header files, as shown. We might want to use some PSI keywords for the PSI files, which are given in `psifiles.h`. And it's always a good idea to include both `stdlib.h` and `stdio.h`.

You will probably find it helpful to use a Makefile to help you compile and link to all the correct libraries. We won't really need the lapack and blas for the MP2 code, but you might want it for the transformation, so let's leave in all the complicated stuff that allows these libraries to find the low-level Fortran libraries they need (you may need to hunt around and update these Fortran lib paths for your system). Something like the one in Figure 2 should do the trick.

MP2 Program:

1. Here you will compute the correlation energy, add it to the SCF energy, and get the total energy. We will assume an RHF reference and a closed-shell molecule for simplicity.
2. Derive the equation for the closed-shell, RHF-based MP2 energy expression, starting from the usual spin-orbital MP2 energy equation and integrating out spin. This is a useful refresher for your Szabo and Ostlund type quantum chemistry skills and should be relatively straightforward.
3. You will note that you need two types of quantities for the spin-integrated MP2 energy expression: two-electron integrals, and orbital energies. Make sure your energy expression

Figure 1: Basic skeleton of a PSI program.

```
#include <stdlib.h>
#include <stdio.h>
#include <libipv1/ip_lib.h>
#include <psifiles.h>
#include <libqt/qt.h>
#include <libciomr/libciomr.h>
#include <libchkpt/chkpt.h>
#include <libpsio/psio.h>

FILE *infile, *outfile;
char *psi_file_prefix;
int main(int argc, char *argv[])
{
  extern char *gprgid(void);
  psi_start(argc-1, argv+1, 0);
  ip_cwk_add(gprgid());
  psio_init();
  /* to start timing, tstart(outfile); */
  /* Insert code here */
  /* to end timing, tstop(outfile); */
  psio_done();
  psi_stop();
}
char *gprgid(void)
{
   /* YOU NEED THE COLON IN THE STRING BELOW */
   char *prgid = ":CODE_NAME";
   return(prgid);
}
```

is in terms of spatial orbital integrals in chemists' notation, because that's what PSI most straightforwardly computes.

4. You might want to begin your program by doing some very simple things like reading in the number of MO's and the HF reference energy. You can do that with the `libchkpt` library. `chkpt_init(PSIO_OPEN_OLD)` opens the checkpoint file for reading, `chkpt_rd_escf()` reads the SCF energy, and `chkpt_rd_nmo()` reads the number of molecular orbitals. What could

Figure 2: Basic PSI Makefile.

```
SRC = mp2.c
PSI = /theoryfs/ds/home/sherrill/psi3-bin
PSI_LIB = $(PSI)/lib
PSI_INCLUDE = $(PSI)/include
INCL =
PSILIBS = -lPSI_chkpt -lPSI_iwl -lPSI_psio -lPSI_ciomr -lPSI_ipv1
OBJ = $(SRC:.c=.o)
EXE = mp2
CC = gcc
CFLAGS = -ansi -g -I$(PSI_INCLUDE)
LDFLAGS = -o $(EXE)
LIBS = -L$(PSI_LIB) $(PSILIBS)
   -L/usr/lib/gcc/x86_64-redhat-linux/4.0.2  \
   -L/usr/lib/gcc/x86_64-redhat-linux/4.0.2/../../../../lib64 \
   -L/usr/lib/gcc/x86_64-redhat-linux/4.0.2/../../.. -L/lib/../lib64 \
   -L/usr/lib/../lib64 -lgfortranbegin -lgfortran -lm -lgcc_s \
   -lblas -llapack

%.o: %.c
	$(CC) -c $(CFLAGS) $*.c

$(EXE): $(OBJ)
	$(CC) $(OBJ) $(LDFLAGS) $(LIBS)

$(OBJ): $(INCL)

clean:
	rm -f $(OBJ) $(EXE) core
```

NOTE: As in any makefile, the rule lines (e.g., `$(CC) -c $(CFLAGS) $*.c`*) MUST begin with a tab character, not spaces.*

be easier? Once you're done with the checkpoint file, you can close it with `chkpt_close()`.

5. In the first pass, we will get the two-electron ingetrals from PSI using the `iwl_rdtwo()` function. You will need to run the usual PSI programs `input`, `cints`, `cscf`, and `transqt` to generate the requisite files. (Later we will replace `transqt` with a transformation program of your own!) This is the same `iwl_rdtwo()` function used in the Hartree-Fock programming

project, except now we will read from the MO integrals file instead of the SO integrals file. It may be called as

```
iwl_rdtwo(PSIF_MO_TEI, buffer, ioff, nmo, 0, 0, 0, outfile);
```

where `nmo` is the number of molecular orbitals and `ioff` is a precomputed offset array such that

```
ioff[p] = (p(p+1))/2
```

The integrals are packed in the same way as the SO integrals read for the Hartree-Fock project, so refer to those notes to understand how big you need to make `buffer` and how to unpack or read the integrals there. If you want to print out the two-electron integrals for debugging (do not do this except for very tiny test cases!!), you can set the next-to-last argument to 1 instead of 0.

6. Read in or compute the orbital energies. They may be easily read using the function `chkpt_rd_evals()`, which returns a pointer to a double array (which the function will allocate for you) containing all the orbital energies. *Warning! The orbital energies come back grouped by irrep, and are only in energy ordering within each irrep.* This means they will probably be all jumbled up for your purposes, unless you run the computation explicitly in the `C1` subgroup.

7. Once you have all this information, you are ready to set up the loops over orbitals and compute the MP2 correlation energy. Don't forget that PSI (in the `C1` subgroup, anyway) will number the occupied orbitals first, and then the virtuals, and the virtual numbering picks up where the occupieds leave off. So, virtual orbital indices start from the number of occupied orbitals, not from zero.

8. The correlation energy plus the Hartree-Fock energy gives the total MP2 energy.

Orbital Transformation Program:

1. Here we will transform the one- and two-electron integrals from the SO basis to the MO basis. They can be read from the SO basis files produced by `cints` as described in the Hartree-Fock programming project.

2. The one-electron integrals in the MO basis can be produced from the SO basis integrals by two matrix-matrix multiplications:

$$h_{pq} = \sum_{\mu\nu} C_{p\mu} h_{\mu\nu} C_{\nu q}, \tag{1}$$

for every $0 \leq p, q <$ `nmo` and with $0 \leq \mu, \nu <$ `nso`. Because this is true for all p, q, this is a set of matrix operations

$$\mathbf{h}^{MO} = \mathbf{C}^{\dagger}\mathbf{h}^{SO}\mathbf{C} \tag{2}$$

3. Similarly, the two-electron integrals are transformed as

$$(pq|rs) = \sum_{\mu\nu\rho\sigma} C_{p\mu} C_{q\nu} (\mu\nu|\rho\sigma) C_{\rho r} C_{\sigma s}. \tag{3}$$

   Here, the four-index quantities are not quite as straightforward to deal with, and a number of possibilities present themselves. The very obvious solution of simply looping over all p, q, r, s, and then looping over all μ, ν, ρ, σ, is a very poor idea! This would give eight nested loops, making this an $\mathcal{O}(N^8)$ process. If you don't believe this is slow, just give it a try.

4. The point of this exercise is to understand how the $\mathcal{O}(N^8)$ process can be converted into four $\mathcal{O}(N^5)$ steps. The trick is to convert *one index at a time* from the SO to the MO basis. That is, start with the matrix (tensor) $(\mu\nu|\rho\sigma)$, convert one of the indices (a "quarter transform"), then convert the next one, until a total of four quarter-transforms are accomplished. It might help to think of it this way:

$$(pq|rs) = \sum_{\mu} C_{p\mu} \left(\sum_{\nu} C_{q\nu} \left(\sum_{\rho} C_{r} \left(\sum_{\sigma} C_{s} (\mu\nu|\rho\sigma) \right) \right) \right) \tag{4}$$

   This is called *factorizing* the expression, and it helps reduce the cost immensely. (Give it a try!) Extra challenge: see if you can formulate the two-electron integral transformation as a series of matrix multiplications (how it's actually done). Make sure you only run test cases small enough that you don't run out of core memory. Note that in the manipulations we're describing, we haven't been using the permutational symmetry of the integrals (this is possible, but it isn't really compatible with an efficient matrix multiplication algorithm).

5. To finish up, try writing your transformed integrals out to disk in the format of your choosing, and then reading them back in to your MP2 program.

If you code up the integral transformation and MP2 code, and have written a Hartree-Fock code, then you've covered all of the basic components of evaluating correlated energies, except for the integrals themselves!

Additional Information: See A. Szabo and N. S. Ostlund, *Modern Quantum Chemistry* (Dover, New York, 1996).