TCCM homework 1: Computation of the MP2 energy

A. Ammar, Y. Damour, P. Reinhardt, A. Scemama

November 22, 2024

You can work in groups of up to three people. We expect that you will use Git to work collaboratively on your project. We expect the following files and structure in your submission:

- A `LICENSE` file specifying the license for your code.
- An `AUTHORS` file listing the names of all contributors.
- A `README.md` file providing a brief description of the directory structure and the project.
- An `INSTALL.md` file with clear instructions on how to compile and run the program.
- A `tests` directory containing tests to ensure that the program behaves as expected.
- (Optional) A `doc` directory for additional documentation, if the project requires more detail than the `README.md` file can provide.
- A `src` directory containing all the source files of your program.

In this homework, you will read data (integrals, orbital energies) from a file and use it to compute the Hartree-Fock energy and the MP2 correlation energy for a closed-shell system.

1 Theory

1.1 Hartree-Fock Energy Calculation

Using the one- and two-electron integrals in the molecular orbital (MO) basis, the (closed-shell) Hartree-Fock energy can be computed as:

$$E = E_{\mathrm{NN}} + 2\sum_{i=1}^{N_{\mathrm{occ}}} \langle i|h|i\rangle + \sum_{i=1}^{N_{\mathrm{occ}}}\sum_{j=1}^{N_{\mathrm{occ}}} \left[2\langle ij|ij\rangle - \langle ij|ji\rangle\right] \tag{1}$$

where:

- E_{NN} is the nuclear repulsion energy,
- $2\sum_i \langle i|h|i\rangle$ represents the kinetic energy and electron-nucleus potential term,
- $\langle pq|rs\rangle$ are the two-electron integrals,
- Indices i, j run over the *occupied* molecular orbitals.

1.2 MP2 Energy Correction Calculation

If the MOs are eigenfunctions of the Fock matrix, the MP2 energy is given by:

$$E_{\mathrm{MP2}} = \sum_{(i,j)\in \mathrm{occupied}} \sum_{(a,b)\in \mathrm{virtuals}} \frac{\langle ij|ab\rangle\big(2\langle ij|ab\rangle - \langle ij|ba\rangle\big)}{e_i + e_j - e_a - e_b} \tag{2}$$

where:

- e_p is the energy of the p-th orbital (the p-th eigenvalue of the Fock matrix),
- i, j label *occupied* orbitals,
- a, b label *virtual* orbitals.

2 Programming

2.1 Installing TREXIO

You will need to read data from a TREXIO file, a format and library designed for exchanging wave function data between quantum chemistry programs. First, install the TREXIO library on your machine.

TREXIO depends on the HDF5 library, which can be installed using your package manager. On Ubuntu:

```
sudo apt install libhdf5-dev
```

On macOS:

```
brew install hdf5
```

Download TREXIO here: `https://github.com/TREX-CoE/trexio/releases/download/v2.5.0/trexio-2.5.0.tar.gz` and install it on your system:

```
tar -zxvf trexio-2.5.0.tar.gz
cd trexio-2.5.0
./configure
make
sudo make install
```

This installs the TREXIO library to `/usr/local/lib`.

Documentation for TREXIO is available here: `https://trex-coe.github.io/trexio/`.

2.2 Accessing Data from TREXIO Files

Include the TREXIO header in all source files that use the library:

```
#include <trexio.h>
```

A few sample files are provided in the `data` directory.

2.2.1 Opening and Closing TREXIO Files

To open a TREXIO file:

```
trexio_exit_code rc;
trexio_t* trexio_file = trexio_open(filename, 'r', TREXIO_AUTO, &rc);
if (rc != TREXIO_SUCCESS) {
  printf("TREXIO Error: %s\n", trexio_string_of_error(rc));
  exit(1);
}
```

To close the TREXIO file:

```
rc = trexio_close(trexio_file);
if (rc != TREXIO_SUCCESS) {
  printf("TREXIO Error: %s\n", trexio_string_of_error(rc));
  exit(1);
}
trexio_file = NULL;
```

Each TREXIO function returns an exit code (`trexio_exit_code`), where `TREXIO_SUCCESS` indicates success.

2.2.2 Linking with TREXIO

Compile your program with the `-ltrexio` flag and specify the include and library paths:

```
gcc -I/usr/local/include -L/usr/local/lib -ltrexio my_code.c -o my_code
```

2.2.3 Reading the Nuclear Repulsion Energy

Use this function to read the nuclear repulsion energy:

```
trexio_exit_code trexio_read_nucleus_repulsion(trexio_t* const trexio_file,
                                               double* const energy);
```

Be careful, this function takes as second argument a a pointer to a `double` variable. Similarly to `scanf`, you need to pass the address to modify a variable. You will need to use this function as

```
trexio_exit_code rc; // TREXIO return code
double energy; // Variable where the energy is read
rc = trexio_read_nucleus_repulsion(trexio_file, &energy);
// Check the return code to be sure reading was OK
if (rc != TREXIO_SUCCESS) {
  printf("TREXIO Error reading nuclear repulsion energy:\n%s\n",
         trexio_string_of_error(rc));
  exit(1);
}
```

A correct implementation should return a nuclear repulsion energy of 9.19497 atomic units.

2.2.4 Reading One-Electron Integrals

Read the number of molecular orbitals using:

```
trexio_exit_code trexio_read_mo_num(trexio_t* const trexio_file,
                                    int32_t* const mo_num);
```

Read the one-electron integrals (core Hamiltonian) using:

```
trexio_exit_code trexio_read_mo_1e_int_core_hamiltonian(trexio_t* const trexio_file,
                                                        double* const data);
```

The `data` array should be large enough to hold $(\mathrm{mo}_{\mathrm{num}})^2$ elements.

2.2.5 Reading Two-Electron Integrals

Two-electron integrals are stored in a sparse format. First, get the number of non-zero integrals:

```
trexio_exit_code trexio_read_mo_2e_int_eri_size(trexio_t* const trexio_file,
                                                int64_t* const n_integrals);
```

Allocate memory for indices and values:

```
int32_t* const index = malloc(4 * n_integrals * sizeof(int32_t));
if (index == NULL) {
  fprintf(stderr, "Malloc failed for index");
  exit(1);
}

double* const value = malloc(n_integrals * sizeof(double));
if (value == NULL) {
  fprintf(stderr, "Malloc failed for value");
  exit(1);
}
```

and read the integrals from the file using

```
trexio_exit_code trexio_read_mo_2e_int_eri(trexio_t* const file,
                                           const int64_t offset_file,
                                           int64_t* const buffer_size,
                                           int32_t* const index,
                                           double* const value);
```

where `offset_file=0` and `buffer_size = n_integrals`. Warning: you should pass the address of `buffer_size`, because on output of this function `buffer_size` contains the number of read integrals.

To access the n-th integral, you can get the 4 indices (i, j, k, l) and the corresponding value $\langle ij|kl\rangle$ using

```
int i = index[4*n+0];
int j = index[4*n+1];
int k = index[4*n+2];
int l = index[4*n+3];
double integral = value[n];
```

Warning: Two-electron integrals obey 8-fold permutational symmetry:

$$\langle ij|kl\rangle = \iint \phi_i(\mathbf{r}_1)\phi_j(\mathbf{r}_2)\frac{1}{|\mathbf{r}_1 - \mathbf{r}_2|}\phi_k(\mathbf{r}_1)\phi_l(\mathbf{r}_2)d\mathbf{r}_1 d\mathbf{r}_2$$

- i and k can be swapped
- j and l can be swapped
- $\mathbf{r}_1$ and $\mathbf{r}_2$ can be swapped

So $\langle ij|kl\rangle = \langle il|kj\rangle = \langle kl|ij\rangle = \langle kj|il\rangle = \langle ji|lk\rangle = \langle li|jk\rangle = \langle lk|ji\rangle = \langle jk|li\rangle$. To minimize the size of the file, for each given quartet of indices only one permutation is stored in the TREXIO file. In what follows, you must handle this symmetry during computations.

2.3 Computing the Hartree-Fock Energy

Use the data you have read to implement Eq.(1) and compute the Hartree-Fock energy. The expected result is -76.0267987 atomic units for the water molecule.

2.4 Computing the MP2 Energy Correction

To compute the MP2 energy correction:

- Read orbital energies using:

```
trexio_exit_code trexio_read_mo_energy(trexio_t* const file,
                                       double* const mo_energy);
```

- Implement Eq.(2).

The expected MP2 correction is -0.20395997 atomic units for the water molecule.