

MENG 25510 Final Report: Hartree-Fock on HeH^+ with 6-311G and STO-3G

N. R. Dohrmann and S. W. Fitz

(Dated: 30 May 2022)

Please see the corresponding code at https://github.com/FitzSW/HF_HeH for this project.

I. INTRODUCTION

This is a reference document to go with our submission for the MENG 25510 final coding project, for which we chose to implement the Hartree-Fock (HF) Self-Consistent Field (SCF) method on a HeH^+ molecular system using the 6-311G basis set on both atoms for one round of calculations, and STO-3G for another. The geometry at which we performed the calculation was optimized at the CCSD/aug-cc-pTZ level of theory using Gaussian 16¹. The result of our code is then compared to an energy found with PySCF²⁻⁴ using the same geometry and basis set.

II. HARTREE-FOCK ALGORITHM AND CODE DETAILS

We closely follow the suggested implementation scheme given in Szabo and Ostlund⁵, in which explicit formulas and matrix algorithms are given. Any of the following formulas can be found in this reference text if not explicitly stated otherwise.

A. Program Control Flow

First, the input geometry and basis set files are read into the main program (`main.f90`), which are then used to construct a 1D array of the “orbitals” derived type (written in `orbitals.f90`), which each include the name of its host atom, the orbital angular momentum type (e.g. “s”), the coordinates of the host atom, the length of the contraction, as well as arrays of coefficients and exponential factors for the contraction.

Then, the program reads in the desired basis set (files included in the “Basis_Set” directory) as well as geometry (under “Opt”) with subroutines written in the `reader.f90` module in order to construct the set of appropriate orbitals. Currently, these are changed by editing the main file, though geometries and basis sets could be adapted to be read in via the command line.

From here, we then calculate the stored integrals to populate the matrices \mathbf{S} , \mathbf{T} , $\{\mathbf{V}^{\text{nuc}}\}$, and \mathbf{TE} , with the last representing the two-electron integrals. The eigenvalue problem that we are trying to solve is recommended to be handled via a transformation to a set of orthogonalized orbitals via a matrix \mathbf{X} obtained from the *symmetric orthogonalization* scheme

$$\mathbf{X} = \mathbf{S}^{-1/2} \quad (1)$$

The necessary matrix computations are done with assistance from the C++ numerics library Eigen⁶, which is also later used to find the eigenvalues and eigenvectors of matrices at each cycle of the SCF procedure when solving

$$\mathbf{F}'\mathbf{C}' = \mathbf{C}'\boldsymbol{\varepsilon} \quad (2)$$

with \mathbf{F}' as the transformed Fock matrix, \mathbf{C}' as the transformed coefficient matrix, and $\boldsymbol{\varepsilon}$ as a diagonal matrix of orbital energies.

After the transformed coefficient matrix is found, it is reverted back to the original basis via $\mathbf{C} = \mathbf{X}\mathbf{C}'$. A real attempt at the density matrix \mathbf{P} can now be found, as well as \mathbf{G} for the next cycle of the SCF procedure using a new Fock matrix. This cycle is repeated iteratively until the largest change in any element of the density matrix is less than 10^{-4} . Finally, once we have obtained converged matrices, a numerical value for the ground state electronic energy of the occupied molecular orbital is

$$E_0^{\text{elec.}} = \frac{1}{2} \sum_{\mu\nu} \mathbf{P}_{\mu\nu} (\mathbf{H}_{\mu\nu}^{\text{core}} + \mathbf{F}_{\mu\nu}) \quad (3)$$

which can then be added to the nuclear repulsion energy $E_0 = E_0^{\text{elec.}} + E_0^{\text{nuc.}}$.

III. USE AND INSTALLATION

In order to run our code, please clone the git repository that is linked in the abstract of this document. These instructions will assume that you are on a *nix system (i.e. you have access to `wc` and `rm` as system commands) and have `gfortran` and a C++ compiler such as `clang` that has access to the Eigen header files. If you don't have the header files, then run either of the below commands depending if you are on a Mac or are on Linux (Midway preferred)

```
./mac_eigen.sh    ./midway_eigen.sh
```

These will install the necessary files via git. **Please make sure to do this *before* running the build script.** Then, inside the cloned directory, run one of the following commands:

```
./mac_build.sh    ./midway_build.sh
```

to build and execute the project with the 6-311G basis set. The results of the SCF procedure will be written to file in `hf_out.out`. Should you run into any trouble at all running this, please do not hesitate to reach out to us. We guarantee that the program has been tested on both Mac OS and Midway, but if for an unknown reason it doesn't work then let us know.

TABLE I. HF Results Comparison on HeH⁺ by Method

Method / Data Source	Bond Length ^a [Å]	Total E_0 , [Ha.]
CCSD/aug-cc-pVTZ ^b	.776140	-2.9753932
HF/6-311G ^c	.776140	-2.9164905
HF/6-311G ^d	.776140	-2.9164650
HF/STO-3G ^c	.776140	-2.8421947
HF/STO-3G ^d	.776140	-2.8600130

^a Optimized with CCSD/aug-cc-pVTZ, this value used with all subsequent calculations.

^b Gaussian 16

^c PySCF

^d This Report

IV. RESULTS AND DISCUSSION

Encouragingly, we have found results in quite good agreement with PySCF for both basis sets. Within the version of the code that uses 6-311G, we found a final electronic energy of -4.28007793 with a nuclear repulsion energy of 1.36361289 for a total molecular energy of -2.91646504 [Ha.]. Likewise, the energies obtained with STO-36 (listed in the same order as above) are -4.22688007, 1.36686718, and -2.86001301 [Ha.]. Please see a comparison of our resulting energy to the expected PySCF value and a higher-quality CCSD calculation in Table I.

V. CONCLUSION

If this project were to be extended for more general use, the basis sets and geometries of additional systems could easily be read into the main program in the current state of the code. The main area for improvement that would have to be more thoroughly adapted from the current state of the project are the

evaluation of molecular integrals involving atomic orbitals of angular momentum $l > 0$, which are much more complex in their evaluation than the s type orbitals that are considered in this report. Additionally, there is further room for optimization in the linear algebra functionality of this project. Currently, matrices must be written to file, processed, and read back into the main executable at the start of the program and at each stage of the SCF cycle. Clearly, this is not the most efficient way to handle matrix computations, and an improved code would ideally include LAPACK functions and subroutines to be used within Fortran itself.

¹M. J. Frisch, G. W. Trucks, H. B. Schlegel, G. E. Scuseria, M. A. Robb, J. R. Cheeseman, G. Scalmani, V. Barone, G. A. Petersson, H. Nakatsuji, X. Li, M. Caricato, A. V. Marenich, J. Bloino, B. G. Janesko, R. Gomperts, B. Mennucci, H. P. Hratchian, J. V. Ortiz, A. F. Izmaylov, J. L. Sonnenberg, D. Williams-Young, F. Ding, F. Lipparini, F. Egidi, J. Goings, B. Peng, A. Petrone, T. Henderson, D. Ranasinghe, V. G. Zakrzewski, J. Gao, N. Rega, G. Zheng, W. Liang, M. Hada, M. Ehara, K. Toyota, R. Fukuda, J. Hasegawa, M. Ishida, T. Nakajima, Y. Honda, O. Kitao, H. Nakai, T. Vreven, K. Throssell, J. A. Montgomery, Jr., J. E. Peralta, F. Ogliaro, M. J. Bearpark, J. J. Heyd, E. N. Brothers, K. N. Kudin, V. N. Staroverov, T. A. Keith, R. Kobayashi, J. Normand, K. Raghavachari, A. P. Rendell, J. C. Burant, S. S. Iyengar, J. Tomasi, M. Cossi, J. M. Millam, M. Klene, C. Adamo, R. Cammi, J. W. Ochterski, R. L. Martin, K. Morokuma, O. Farkas, J. B. Foresman, and D. J. Fox, "Gaussian16 Revision C.01," (2016), gaussian Inc. Wallingford CT.

²Q. Sun, X. Zhang, S. Banerjee, P. Bao, M. Barbry, N. S. Blunt, N. A. Bogdanov, G. H. Booth, J. Chen, Z.-H. Cui, *et al.*, "Recent developments in the pyscf program package," J. Chem. Phys. **153**, 024109 (2020).

³Q. Sun, T. C. Berkelbach, N. S. Blunt, G. H. Booth, S. Guo, Z. Li, J. Liu, J. D. McClain, E. R. Sayfutyarova, S. Sharma, S. Wouters, and G. K.-L. Chan, "Pyscf: the python-based simulations of chemistry framework," WIREs Comput. Mol. Sci. **8**, e1340 (2018), <https://wires.onlinelibrary.wiley.com/doi/pdf/10.1002/wcms.1340>.

⁴Q. Sun, "Libcint: An efficient general integral library for gaussian basis functions," J. Comp. Chem. **36**, 1664–1671 (2015), <https://onlinelibrary.wiley.com/doi/pdf/10.1002/jcc.23981>.

⁵A. Szabo and N. S. Ostlund, *Modern quantum chemistry: introduction to advanced electronic structure theory* (Courier Corporation, 2012).

⁶G. Guennebaud, B. Jacob, *et al.*, "Eigen v3," <http://eigen.tuxfamily.org> (2010).