Basic Configuration Interaction Theory (Draft)

Julio Candanedo[1]
Department of Physics, Arizona State University, Tempe, AZ 85287, USA

Contents

1 Many Body Theory **3**
1.1 Quantum Fermionic Calculus 4
1.1.1 Reduction of Many-Body Wavefunctions 4
1.2 Many-Body Hamiltonian 4
1.3 1-body 4
1.4 2-body 4

2 Combinatorics **4**
2.1 FCI 4
2.2 CIS 5
2.3 CISD 5

3 SO-Matrix Elements **5**

4 CI-Matrix Elements **7**
4.1 Slater-Condon Information 7
4.2 Excitation/Deëxcitation Operators 9
4.3 CI Matrix Construction 10

5 Solving the CI Equation **10**

6 Comparison and Examples **10**
6.1 Fully Orthogonal Eigenstates 10

7 Conclusion **10**

8 Acknowledgements **10**

A Python Subindexing **10**

B CI-Matrix from sub-indexing **10**

[1] jcandane@asu.edu

C Raw Code **11**
C.1 useful definitions 11
C.2 `fci.py` 13
C.3 `cis.py` 15
C.4 `cisd.py` 16

Here we plan to go beyond mean-field (1-body) theories by considering all possible combinations of configurations, in an effort to determine the distribution of electrons in molecules. This theory is broadly known as the Configuration Interaction (CI). Like the full Schrödinger equation the theory is linear and is solved via direct diagonalization of the full-CI Hamiltonian. The full-CI is an approximation of the full-SE (SE), for a finite basis set, whereas the SE is the idealistic equation describing the structure on the continuum (an infinite basis set) as well. The full-CI Hamiltonian is constructed matrix-element by matrix-element for non-zero elements described by the well-known *Slater-Condon* rules, described in §4.1. These rules exist because physical Hamiltonians at the microscopic-level contain 1-particle (1-body) and 2-particle/pair (2-body) interactions. These 1-particle interactions are the kinetic energy or effective potential (interactions with particles that are not electrons, e.g. nuclei). While the 2-particle interactions are the inter-electron interactions. The equations implementing the Slater-Condon rules for explicitly restricted-systems are found in both: [Szabo and Ostlund, 2012] and [Helgaker *et al.*, 2014]. Because the FCI space is very large, i.e. scaling exponentially with the number of electrons, it is computationally intractable for most interesting systems (in chemistry and biochemistry). It is often truncated by excitations (mth excitations §**??**) on General-Active-Space (GAS, §**??**) and Multi-Reference (MR, §**??**) descriptions. Our goal is to reduce CI theory into a game, combinatorics followed by a set of rules, and with it to construct a Hamiltonian from the subset of Slater-Determinants $\Lambda_{\text{CI}} \subset \Lambda_{\text{FCI}}$. Additionally, this paper describes the implementation of the CI equations in [PySCF *et al.*, 2018], to hopefully connect and render CI-theory more transparent. The FCI tool of [Psi4NumPy *et al.*, 2018] was initially consulted, but proved to be unhelpful for the author's purposes. Note Einstein-summation-notation, i.e. sum-of-repeated-indices, is used through-out the text.

1 Many Body Theory

Our objective is to understand the Many-Body Schrödinger-equation, romantically put:

$$H\Psi = \mathscr{E}\Psi$$

with H being the Many-body Hamiltonian, Ψ being the many-body *wavefunction* (a complex vector), and $\mathscr{E}$ being the many-body excitation energies.

I.e. in order to compute the Schrödinger-equation, we would need the structure of the Hamiltonian. For this the H Hamiltonian and Ψ are evaluated on a basis $\mathbb{x}$

$$\Psi = \sum \bigwedge \psi$$

$$H(\mathbb{x})\Psi(\mathbb{x}) = \mathscr{E}\Psi(\mathbb{x})$$
$$H(\mathbb{x})\langle \mathbb{x}|\Psi\rangle = \mathscr{E}\langle \mathbb{x}|\Psi\rangle$$

1.1 Quantum Fermionic Calculus

$$\Psi = \psi \wedge \psi \wedge \cdots \wedge \psi$$
$$\mathbb{x} =$$

1.1.1 Reduction of Many-Body Wavefunctions

$$\Psi^{(m)} = \int \Psi^{(n)}$$

1.2 Many-Body Hamiltonian

1.3 1-body

If the electrons are nonrelativistic, then they have a kinetic energy given by the Laplacian-operator[2]:

$$H \sim \langle \Psi | \Delta | \Psi \rangle$$

1.4 2-body

Identical particles, as one electron shifts it shifts

This may actually be accounted by a mean-field *functional* to form an effective 1-body operator, as Density Functional Theory, but in general post-MF methods require a 2-body operation.

$$H \sim \frac{1}{r}$$

2 Combinatorics

2.1 FCI

The Combinatorics of FCI follow the *N-balls-in-M-slots* combinatorics (for identical balls with $N < M$). Except, the *slots* are called orbitals of the MO/SO variety, and *balls* are called electrons. Here we consider SO-orbitals, this considers two independent *N-balls-in-M-slots* problems for $\uparrow$ and $\downarrow$ electrons separately. The number of combinations for the *N-balls-in-M-slots* problem is given by the Choose expression:

$$\#\text{ states} = \binom{M}{N} = \frac{M!}{N!(M-N)!}.$$

Although it might be worthwhile to compute this for low M and N, the computer can do this for larger M and N.

[2]If relativistic, this is the Dirac-operator, which may be expanded to first order to the Laplace-operator, this is beyond the scope of this paper.

2.2 CIS

Suppose we are given an initial determinant Λ and we would like to generate all excitations

For CIS theory, we may apply the same formalism, *N-balls-in-M-slots*, but we have to separate the occupied and vacant spaces.

An arbitrary excitation/deëxcitation may be done, by a creation and annihilation event.

A creation event entails a new occupation in the vacant space.

While an annihilation event entails a removal from the occupied space.-

2.3 CISD

3 SO-Matrix Elements

After an SCF calculation, we obtain the MO-coefficients $\{C^{\uparrow}, C^{\downarrow}\}$ which not only yield orthogonal states, but are minimized to a mean-field shielding of the electronic charges. This forms a foundation to build beyond the mean-field approximation, and these theories, including CI are called post-HF or post-SCF. Using these coefficients the 1-body and 2-body interactions may be formed by *dressing* the atomic-centric interactions given by the Atomic Orbital, i.e. AO-arrays. This dressed interaction is called the Spin-Orbit (SO) representation, and serves as the playground where CI games occur. That is to say, CI theory itself keeps the same level of SCF shielding throughout all electronic states[3].

The dressing is implemented by matrix-products between the AO-representation and the Mean-Field shielding Orbital (MO) coefficients. Here we consider 1-body AO interaction represented by a 2-dimensional matrix/array $(\alpha|\beta)$ and a 2-body AO interaction represented by a 4-dimensional matrix/array $(\alpha\beta|\gamma\delta)$:

$$
\begin{aligned}
(\sigma p \,|\, \sigma' q) &= \sum_{\alpha\beta} (\alpha \,|\, \beta)\, C^{\alpha}_{\sigma p} C^{\beta}_{\sigma' q} \\
(\sigma p, \sigma' q \,||\, \tau r, \tau' s) &= \sum_{\alpha\beta\gamma\delta} (\alpha, \beta \,||\, \gamma, \delta)\, C^{\alpha}_{\sigma p} C^{\beta}_{\sigma' q} C^{\gamma}_{\tau r} C^{\delta}_{\tau' s} \\
&= \sum_{\alpha\beta\gamma\delta} (p, q \,||\, r, s)^{\sigma\sigma'\tau\tau'}
\end{aligned}
$$

Lets attempt to understand the 2-body interaction first. Here the complete SO-transformed

[3]Correcting for the excited state shielding and polarizability is done by Multiconfigurational Mean Field Methods, beyond standard CI.

MO inter-electron integrals are[4]:

$$(p,q\,|\,r,s)^{\sigma\sigma'\tau\tau'} = \left(\begin{array}{cc|cc} (p,q\,||\,r,s)^{\uparrow\uparrow\uparrow\uparrow} & (p,q\,|\,r,s)^{\uparrow\uparrow\downarrow\downarrow} & (p,q\,|\,r,s)^{\uparrow\uparrow\uparrow\downarrow} & (p,q\,|\,r,s)^{\uparrow\uparrow\downarrow\uparrow} \\ (p,q\,|\,r,s)^{\downarrow\downarrow\uparrow\uparrow} & (p,q\,||\,r,s)^{\downarrow\downarrow\downarrow\downarrow} & (p,q\,|\,r,s)^{\downarrow\downarrow\uparrow\downarrow} & (p,q\,|\,r,s)^{\downarrow\downarrow\downarrow\uparrow} \\ \hline (p,q\,|\,r,s)^{\uparrow\downarrow\uparrow\uparrow} & (p,q\,|\,r,s)^{\uparrow\downarrow\downarrow\downarrow} & (p,q\,|\,r,s)^{\uparrow\downarrow\uparrow\downarrow} & (p,q\,|\,r,s)^{\uparrow\downarrow\downarrow\uparrow} \\ (p,q\,|\,r,s)^{\downarrow\uparrow\uparrow\uparrow} & (p,q\,|\,r,s)^{\downarrow\uparrow\downarrow\downarrow} & (p,q\,|\,r,s)^{\downarrow\uparrow\uparrow\downarrow} & (p,q\,|\,r,s)^{\downarrow\uparrow\downarrow\uparrow} \end{array}\right)$$

$$= \left(\begin{array}{cc|cc} (p,q\,||\,r,s)^{\uparrow\uparrow\uparrow\uparrow} & (p,q\,|\,r,s)^{\uparrow\uparrow\downarrow\downarrow} & 0 & 0 \\ (p,q\,|\,r,s)^{\downarrow\downarrow\uparrow\uparrow} & (p,q\,||\,r,s)^{\downarrow\downarrow\downarrow\downarrow} & 0 & 0 \\ \hline 0 & 0 & (p,q\,|\,r,s)^{\uparrow\downarrow\uparrow\downarrow} & (p,q\,|\,r,s)^{\uparrow\downarrow\downarrow\uparrow} \\ 0 & 0 & (p,q\,|\,r,s)^{\downarrow\uparrow\uparrow\downarrow} & (p,q\,|\,r,s)^{\downarrow\uparrow\downarrow\uparrow} \end{array}\right).$$

However, those matrix-elements which have an odd-number of spin electrons are nonphysical and should be 0. Therefore the nonzero matrix-elements are:

$$\begin{aligned} (p\,|\,q)^{\sigma} &= (\sigma p\,|\,\sigma q) = (\alpha\,|\,\beta)\, C^{\alpha}_{\sigma p} C^{\beta}_{\sigma q} \\ (p,q\,|\,r,s)^{\sigma\tau} &= (\sigma p, \sigma q\,|\,\tau r, \tau s) \\ (p,q\,|\,r,s)^{\sigma}_{\tau} &= (\sigma p, \tau q\,|\,\tau r, \sigma s) \\ (p,q\,|\,r,s)_{\sigma\tau} &= (\sigma p, \tau q\,|\,\sigma r, \tau s)\,. \end{aligned}$$

In U-CI, we require only the spin-orbital transformed 1-electron and 2-electron operators, with (spin state given by $\sigma = \{\uparrow, \downarrow\}$):

$$\begin{aligned} (p|q)^{\sigma} &= \begin{pmatrix} h_{\alpha\beta} C^{\uparrow\alpha}_{p} C^{\uparrow\beta}_{q} \\ h_{\alpha\beta} C^{\downarrow\alpha}_{p} C^{\downarrow\beta}_{q} \end{pmatrix} \\ (pq||rs)^{\sigma\tau} &= \begin{pmatrix} (\alpha\beta\,||\,\gamma\delta)\, C^{\uparrow\alpha}_{p} C^{\uparrow\beta}_{q} C^{\uparrow\gamma}_{r} C^{\uparrow\delta}_{s} & (\alpha\beta\,|\,\gamma\delta)\, C^{\downarrow\alpha}_{p} C^{\downarrow\beta}_{q} C^{\uparrow\gamma}_{r} C^{\uparrow\delta}_{s} \\ (\alpha\beta\,|\,\gamma\delta)\, C^{\uparrow\alpha}_{p} C^{\uparrow\beta}_{q} C^{\downarrow\gamma}_{r} C^{\downarrow\delta}_{s} & (\alpha\beta\,||\,\gamma\delta)\, C^{\downarrow\alpha}_{p} C^{\downarrow\beta}_{q} C^{\downarrow\gamma}_{r} C^{\downarrow\delta}_{s} \end{pmatrix}. \end{aligned}$$

[4] from the large blocks, we flip-spin from left-to-right on the 4th MO index, and flip-spin from up-to-down from the 2nd MO index.

4 CI-Matrix Elements

Next, we would like to take the SO-matrix elements, in the previous section, and transform these into the CI-basis. This transformation needs to take into account the structure of the determinants. The structure of the determinants is probes for nontrivial elements as suggested by the Slater-Condon rules, which probe the connectivity of the determinant space. In particular the 1-differences and 2-differences of the determinants.

4.1 Slater-Condon Information

The Slater-Condon rules are dependent on the differences between determinants. In the unrestricted case, we may take differences of the binary representation of the Slater-determinants ($I \to J$):

$$\mathbb{B}_{IJ\sigma p} = B_{J\sigma p} - B_{I\sigma p}. \tag{4.1}$$

Given two CI states I and J, $\mathbb{B}_{\hat{I}\hat{J}\sigma p}$ gives the difference of the binary MO occupations, p, for each spin σ. This matrix consists of 0 (no change), 1 (1 new occupation), xor -1 (1 less occupation). To get the difference-of-occupations for each spin we take the absolute-value for each element and sum (over MOs):

$$S_{IJ\sigma} = \frac{1}{2}\left(\sum_p |\mathbb{B}_{IJ\sigma p}|\right)_{IJ\sigma}.$$

This will be useful to determine the type of Slater-Condon rules required for each CI Hamiltonian matrix element I, J. For each spin-component the possible non-zero differences (a given I, J) in the the determinants are given (e.g. $[\delta\uparrow, \delta\downarrow]$):

$$\begin{array}{ccccc} & & [2,0] & & \\ & [1,0] & & & \\ [0,0] & & [1,1] & & \\ & [0,1] & & & \\ & & [0,2]\,. & & \end{array} \tag{4.2}$$

For differences greater than $\delta\uparrow + \delta\downarrow \geq 3$, have a matrix-element is 0 (rule 3). This yields 7 Slater-Condon rules in U-CI. For each of the 6 terms above apply to some partition of the entire CI matrix. This may be seen by the example on fig. 1, for 4 Slater-Condon rules (each column above, in eq. 4.2).

The CI Hamiltonian matrix-elements may be partitioned into seven 1-dimensional arrays. The value of these matrix-elements are determined by a formula on the next page. As the 0-difference rule is always located on the matrix-diagonal It may be used to initialize the CI Hamiltonian with 0s in the off diagonal (covering rule 3). Next, for the remaining 5 rules, the indices of the CI Hamiltonian for their applicability are determined by (determined by the

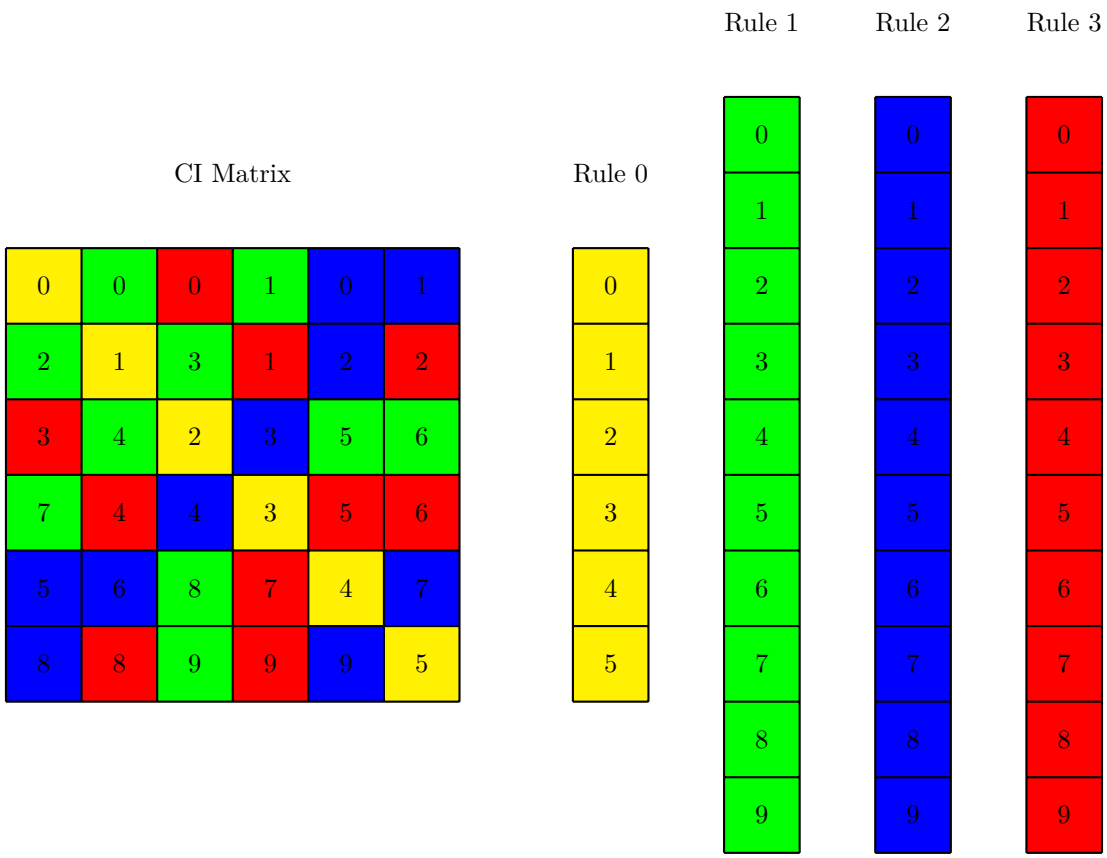

Figure 1: Above is an example of decomposing a 6×6 CI Matrix into 4 1-dimensional arrays, according to which Slater-Condon Rule is satisfied: rule 0 (yellow), rule 1 (green), rule 2 (blue), and rule 3 (red).

Spin-Difference Matrix):

$$\begin{aligned} I^{\uparrow}, J^{\uparrow} \text{ such that } S\left[I^{\uparrow}, J^{\uparrow}\right] &= [1,0] \\ I^{\downarrow}, J^{\downarrow} \text{ such that } S\left[I^{\downarrow}, J^{\downarrow}\right] &= [0,1] \\ I^{\uparrow\uparrow}, J^{\uparrow\uparrow} \text{ such that } S\left[I^{\uparrow\uparrow}, J^{\uparrow\uparrow}\right] &= [2,0] \\ I^{\uparrow\downarrow}, J^{\uparrow\downarrow} \text{ such that } S\left[I^{\uparrow\downarrow}, J^{\uparrow\downarrow}\right] &= [1,1] \\ I^{\downarrow\downarrow}, J^{\downarrow\downarrow} \text{ such that } S\left[I^{\downarrow\downarrow}, J^{\downarrow\downarrow}\right] &= [0,2]\,, \end{aligned}$$

note for a symmetric (in $I \leftrightarrow J$) spin-difference matrix S_{IJ_σ} this is double counting[5].

This algorithm scales as $\mathcal{O} \sim N^2$, with N being the number of CI states (which in-turn scales exponentially). In principle, the sparsity of the CI algorithm scales suggests a more optimum

For an example, let's consider we have a system of 3 spin-orbitals (SO) per spin-type, and 2 $\uparrow$-electrons and 1 $\downarrow$-electron system. This system has 9 possible FCI states (let's enumerate them 1 thru 9). This system has the following connectivity given in figures 2 and 3.

[5]E.g. for every pair $i \in I^{\uparrow}$, $j \in J^{\uparrow}$, there exists a pair $j \in I^{\uparrow}$, $i \in J^{\uparrow}$

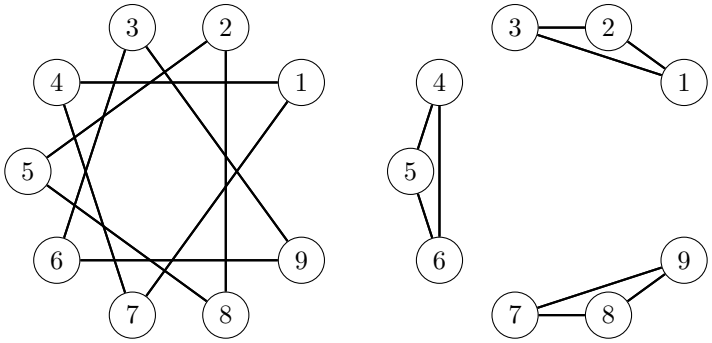

Figure 2: Slater-Condon Rules for 1-difference, in ↑ and ↓ spaces respectively.

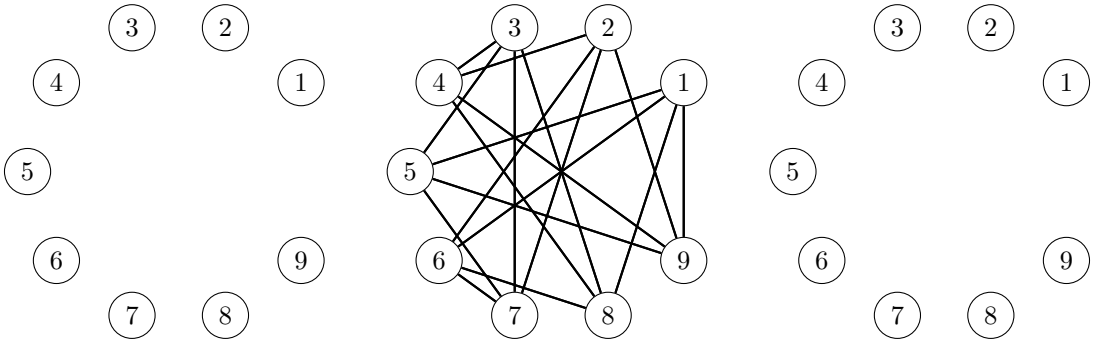

Figure 3: Slater-Condon Rules for 2-difference, in ↑↑, ↑↓, and ↓↓ spaces respectively.

The connectivity of these CI states is purely combinatorics, i.e. independent of the molecular structure. The sparsity of the CI matrix is purely due to the neglecting of determinant-differences greater than 2. If differences of 3 or higher do not exist then the Slater-Condon CI connectivity is just the complete matrix.

4.2 Excitation/Deëxcitation Operators

The creation/annihilation operators are obtained directly from the difference matrix. Suppose $\mathcal{I}$ and $\mathcal{J}$ are already determined to be a nontrivial Slater-Condon matrix element. Then the pair-difference, δ, is just an array with -1, 0, and 1:

$$\delta^{\mathcal{K}\sigma p} = B^{\mathcal{I}\sigma p} - B^{\mathcal{J}\sigma p}$$

where $\delta^{\mathcal{K}\sigma p} = 1$ creation operator, rest matrix p elements set to 0,

where $\delta^{\mathcal{K}\sigma p} = -1$ annihilation operator, rest matrix p elements set to 0,

$$\delta^{\dagger}_{\mathcal{K}\sigma p} \longrightarrow a_{\mathcal{K}\sigma p}, a^{\dagger}_{\mathcal{K}\sigma p}$$

4.3 CI Matrix Construction

The explicit CI matrix construction may be done as:

$$
\begin{aligned}
H &\stackrel{!}{=} 0_{IJ} \\
H[I,I] &= (p|p)^{\sigma}\, B_{I\sigma p} + \frac{1}{2}\,(pp|qq)^{\sigma\tau}\, B^{I\sigma p} B^{I\tau q} \\
H[I^{\uparrow},J^{\uparrow}] &= (p|q)^{\uparrow}\, a^{\mathcal{K}}_{\uparrow p} a^{\dagger\mathcal{K}}_{\uparrow q} + (pq|rr)^{\uparrow\uparrow}\, a^{\mathcal{K}}_{\uparrow p} a^{\dagger\mathcal{K}}_{\uparrow q} c^{\mathcal{K}}_{\uparrow r} + (pq|rr)^{\uparrow\downarrow}\, a^{\mathcal{K}}_{\uparrow p} a^{\dagger\mathcal{K}}_{\uparrow q} c^{\mathcal{K}}_{\downarrow r} \\
H[I^{\downarrow},J^{\downarrow}] &= (p|q)^{\downarrow}\, a^{\mathcal{K}}_{\downarrow p} a^{\dagger\mathcal{K}}_{\downarrow q} + (pq|rr)^{\downarrow\downarrow}\, a^{\mathcal{K}}_{\downarrow p} a^{\dagger\mathcal{K}}_{\downarrow q} c^{\mathcal{K}}_{\downarrow r} + (pq|rr)^{\downarrow\uparrow}\, a^{\mathcal{K}}_{\downarrow p} a^{\dagger\mathcal{K}}_{\downarrow q} c^{\mathcal{K}}_{\uparrow r} \\
H[I^{\uparrow\uparrow},J^{\uparrow\uparrow}] &= (pq|rs)^{\uparrow\uparrow}\, a^{\mathcal{K}}_{\uparrow p} a^{\dagger\mathcal{K}}_{\uparrow q} a^{\mathcal{K}}_{\uparrow r} a^{\dagger\mathcal{K}}_{\uparrow s} \\
H[I^{\uparrow\downarrow},J^{\uparrow\downarrow}] &= (pq|rs)^{\uparrow\downarrow}\, a^{\mathcal{K}}_{\uparrow p} a^{\dagger\mathcal{K}}_{\uparrow q} a^{\mathcal{K}}_{\downarrow r} a^{\dagger\mathcal{K}}_{\downarrow s} \\
H[I^{\downarrow\downarrow},J^{\downarrow\downarrow}] &= (pq|rs)^{\downarrow\downarrow}\, a^{\mathcal{K}}_{\downarrow p} a^{\dagger\mathcal{K}}_{\downarrow q} a^{\mathcal{K}}_{\downarrow r} a^{\dagger\mathcal{K}}_{\downarrow s} \qquad .
\end{aligned}
$$

5 Solving the CI Equation

After construction of the CI Hamiltonian Matrix (in the CI basis), a simple diagonalization is performed to get the eigen-spectrum:

$$H_{IJ} X_{JK} = \mathscr{E}_K X_{IK}$$

where X_{JK} is the eigenvalue matrix and $\mathscr{E}_K$ are the eigenvectors. If the list of determinants include the Hartree-Fock determinant, then this is the solution to the fully correlated ground state.

6 Comparison and Examples

6.1 Fully Orthogonal Eigenstates

7 Conclusion

Ultimately, we are left with two major questions:

How to generate all nontrivial CI matrix elements without $\mathcal{O} \sim N^2$ scaling?

Is it possible to determine the important CI states (and their connectivity), from the molecular geometry alone?

8 Acknowledgements

A Python Subindexing

B CI-Matrix from sub-indexing

C Raw Code

C.1 useful definitions

```python
import numpy as np
from pyscf import fci, ao2mo, scf, gto

def givenAgetB(AA, AB, N_mo):
  "Given A(i occupied orbitals for each determinant) get B (binary rep.)"

  Binary = np.zeros((AA.shape[0], 2, N_mo), dtype=np.int8)
  for I in range(len(Binary)):
      Binary[I, 0, AA[I,:]] = 1
      Binary[I, 1, AB[I,:]] = 1

  return Binary

def SpinOuterProduct(A, B, stack=False):
  AA = np.einsum("Ii, J -> IJi", A, np.ones(B.shape[0], dtype=np.int8)).reshape(
      (A.shape[0]*B.shape[0], A.shape[1]) )
  AB = np.einsum("Ii, J -> JIi", B, np.ones(A.shape[0], dtype=np.int8)).reshape(
      (A.shape[0]*B.shape[0], B.shape[1]) )

  if stack:
    return np.array([AA,AB])
  else:
    return AA, AB

def get_SO_matrix(uhf_pyscf, SF=False, H1=None, H2=None):
    """ Given a PySCF uhf object get SO Matrices """

    Ca, Cb = (uhf_pyscf).mo_coeff
    S = (uhf_pyscf.mol).intor("int1e_ovlp")
    eig, v = np.linalg.eigh(S)
    A = (v) @ np.diag(eig**(-0.5)) @ np.linalg.inv(v)
    H = uhf_pyscf.get_hcore()

    n = Ca.shape[1]
    eri_aa = (ao2mo.general( (uhf_pyscf)._eri , (Ca, Ca, Ca, Ca),
        compact=False)).reshape((n,n,n,n), order="C")
    eri_aa -= eri_aa.swapaxes(1,3)
    eri_bb = (ao2mo.general( (uhf_pyscf)._eri , (Cb, Cb, Cb, Cb),
        compact=False)).reshape((n,n,n,n), order="C")
    eri_bb -= eri_bb.swapaxes(1,3)
    eri_ab = (ao2mo.general( (uhf_pyscf)._eri , (Ca, Ca, Cb, Cb),
        compact=False)).reshape((n,n,n,n), order="C")
    #eri_ba = (1.*eri_ab).swapaxes(0,3).swapaxes(1,2) ## !! caution depends on symmetry
    eri_ba = (ao2mo.general( (uhf_pyscf)._eri , (Cb, Cb, Ca, Ca),
        compact=False)).reshape((n,n,n,n), order="C")
    H2 = np.stack(( np.stack((eri_aa, eri_ab)), np.stack((eri_ba, eri_bb)) ))

    H1 = np.asarray([np.einsum("AB, Ap, Bq -> pq", H, Ca, Ca), np.einsum("AB, Ap, Bq -> pq", H, Cb,
        Cb)])

    if SF:
      eri_abab = (ao2mo.general( (uhf_pyscf)._eri , (Ca, Cb, Ca, Cb),
          compact=False)).reshape((n,n,n,n), order="C")
```

```
      eri_abba = (ao2mo.general( (uhf_pyscf)._eri , (Ca, Cb, Cb, Ca),
          compact=False)).reshape((n,n,n,n), order="C")
      eri_baab = (ao2mo.general( (uhf_pyscf)._eri , (Cb, Ca, Ca, Cb),
          compact=False)).reshape((n,n,n,n), order="C")
      eri_baba = (ao2mo.general( (uhf_pyscf)._eri , (Cb, Ca, Cb, Ca),
          compact=False)).reshape((n,n,n,n), order="C")
      H2_SF = np.stack(( np.stack((eri_abab, eri_abba)), np.stack((eri_baab, eri_baba)) ))
      return H1, H2, H2_SF

    else:
      return H1, H2

def get_excitation_op(i, j, binary, sign, spin=0):
    Difference = binary[i,spin] - binary[j, spin]
    a_t =   (Difference + 0.5).astype(np.int8)
    a   = -1*(Difference - 0.5).astype(np.int8)
    if np.sum(a[0]) > 1: ### this is a double excitation
        å_t = 1*a_t ## make copy
        å_t[ np.arange(len(å_t)),(å_t!=0).argmax(axis=1) ] = 0 ## zero first 1
        a_t = np.abs(å_t - a_t) ## absolute difference from original
        a_t = np.asarray([sign[j, spin]*å_t,sign[j, spin]*a_t]) ## stack

        å = 1*a ## make copy
        å[ np.arange(len(å)),(å!=0).argmax(axis=1) ] = 0 ## zero first 1
        a = np.abs(å - a) ## absolute difference from original
        a = np.asarray([sign[i, spin]*å,sign[i, spin]*a]) ## stack

        return a_t, a

    return sign[j, spin]*a_t, sign[i, spin]*a
```

C.2 fci.py

```
import numpy as np
from itertools import combinations, permutations
from pyscf import fci, ao2mo, scf, gto

mol = gto.M(atom="He 0.0 0.0 0.0; H 1.5 0.0 0.0; He 0.0 1.5 0.0; N 0.0, 0.0, 1.5", spin=0,
    basis="sto-3g")
uhf = scf.UHF(mol)
uhf.kernel()

N   = O_sp.shape[1]
Λ_α = np.asarray( list(combinations( np.arange(0, N, 1, dtype=np.int8) , N_s[0] ) ) )
Λ_β = np.asarray( list(combinations( np.arange(0, N, 1, dtype=np.int8) , N_s[1] ) ) )
ΛA, ΛB = SpinOuterProduct(Λ_α, Λ_β)
Binary = givenΛgetB(ΛA, ΛB, N)

sign  = np.cumsum( Binary, axis=2)
for I in range(len(Binary)):
    iia = np.where( Binary[I,0] == 1)[0]
    iib = np.where( Binary[I,1] == 1)[0]
    sign[I, 0, iia] = np.arange(0, len(iia), 1)
    sign[I, 1, iib] = np.arange(0, len(iib), 1)

Γ_Isp = ( (-1)**(sign) ).astype(np.int8)

H1, H2 = get_SO_matrix(uhf)

## Rule 0 & 3
H_CI  = np.einsum("Spp, ISp -> I", H1, Binary, optimize=True)
H_CI += np.einsum("STppqq, ISp, ITq -> I", H2, Binary, Binary, optimize=True)/2
H_CI  = np.diag(H_CI)

SpinDifference = np.sum( np.abs(Binary[:, None, :, :] - Binary[None, :, :, :]), axis=3)//2

## indices for 1-difference
I_A, J_A = np.where( np.all(SpinDifference==np.array([1,0], dtype=np.int8), axis=2) )
I_B, J_B = np.where( np.all(SpinDifference==np.array([0,1], dtype=np.int8), axis=2) )

### get excitation operators
a_t , a = get_excitation_op(I_A , J_A , Binary, Γ_Isp, spin=0)
b_t , b = get_excitation_op(I_B , J_B , Binary, Γ_Isp, spin=1)
ca      = ((Binary[I_A,0,:] + Binary[J_A,0,:])/2).astype(np.int8)
cb      = ((Binary[I_B,1,:] + Binary[J_B,1,:])/2).astype(np.int8)

## Rule 1
H_CI[I_A , J_A ] -= np.einsum("pq, Kp, Kq -> K", H1[0], a_t, a, optimize=True)
H_CI[I_A , J_A ] -= np.einsum("pqrr, Kp, Kq, Kr -> K", H2[0,0], a_t, a, ca, optimize=True)
H_CI[I_A , J_A ] -= np.einsum("pqrr, Kp, Kq, Kr -> K", H2[0,1], a_t, a, Binary[I_A,1],
    optimize=True)

H_CI[I_B , J_B ] -= np.einsum("pq, Kp, Kq -> K", H1[1], b_t, b, optimize=True)
H_CI[I_B , J_B ] -= np.einsum("pqrr, Kp, Kq, Kr -> K", H2[1,1], b_t, b, cb, optimize=True)
H_CI[I_B , J_B ] -= np.einsum("pqrr, Kp, Kq, Kr -> K", H2[1,0], b_t, b, Binary[I_B,0],
    optimize=True)

## indices for 2-differences
```

```
I_AA, J_AA = np.where( np.all(SpinDifference==np.array([2,0], dtype=np.int8), axis=2) )
I_BB, J_BB = np.where( np.all(SpinDifference==np.array([0,2], dtype=np.int8), axis=2) )
I_AB, J_AB = np.where( np.all(SpinDifference==np.array([1,1], dtype=np.int8), axis=2) )

aa_t, aa = get_excitation_op(I_AA, J_AA, Binary, Γ_Isp, spin=0)
bb_t, bb = get_excitation_op(I_BB, J_BB, Binary, Γ_Isp, spin=1)
ab_t, ab = get_excitation_op(I_AB, J_AB, Binary, Γ_Isp, spin=0)
ba_t, ba = get_excitation_op(I_AB, J_AB, Binary, Γ_Isp, spin=1)

## Rule 2
H_CI[I_AA, J_AA] = np.einsum("pqrs, Kp, Kq, Kr, Ks -> K", H2[0,0], aa_t[0], aa[0], aa_t[1], aa[1],
    optimize=True)
H_CI[I_BB, J_BB] = np.einsum("pqrs, Kp, Kq, Kr, Ks -> K", H2[1,1], bb_t[0], bb[0], bb_t[1], bb[1],
    optimize=True)
H_CI[I_AB, J_AB] = np.einsum("pqrs, Kp, Kq, Kr, Ks -> K", H2[0,1], ab_t, ab, ba_t, ba,
    optimize=True)

nuclear_rep = uhf.energy_nuc()
e_fci_my, X_IJ = np.linalg.eigh(H_CI)

print("CI Energy : " + str( e_fci_my[0] + nuclear_rep ))
```

C.3 cis.py

```
1 a[0]              ##
```

C.4 cisd.py

```
1 a[0]              ##
```

References

[Helgaker *et al.*, 2014] Helgaker, T., Jørgensen, P., and Olsen, J. (2014). *Molecular Electronic-Structure Theory*. John Wiley & Sons.

[Psi4NumPy *et al.*, 2018] Psi4NumPy, Smith, D. G., Burns, L. A., Sirianni, D. A., Nascimento, D. R., Kumar, A., James, A. M., Schriber, J. B., Zhang, T., Zhang, B., Abbott, A. S., *et al.* (2018). *J. Chem. Theory Comput.*, 14(7):3504–3511.

[PySCF *et al.*, 2018] PySCF, Sun, Q., Berkelbach, T. C., Blunt, N. S., Booth, G. H., Guo, S., Li, Z., Liu, J., McClain, J. D., Sayfutyarova, E. R., Sharma, S., *et al.* (2018). *Wiley Interdiscip. Rev. Comput. Mol. Sci.*, 8(1):e1340.

[Szabo and Ostlund, 2012] Szabo, A. and Ostlund, N. S. (2012). *Modern Quantum Chemistry*. Dover Publications.