MCSCF from A to C

Shirong Wang

Department of Chemistry, Godsson University

August 16, 2021

Plan

Molecular Electronic-Structure Theory, by Trygve Helgaker, Poul Jørgensen, Jeppe Olsen

1. - CASCI §11.1
   - Orbital rotation §3
   - 2nd-order SCF §10.7–10.10
   - CASSCF §5.5, §12
2. MRPT2 §14.7
3. MRCI, DFT/MRCI

Further topics

- Spin projection
- GW/BSE

Overview

Optimization Methods: Basics

$$E(\mathbf{X}_1) = E(\mathbf{X}) + \mathbf{q}^\dagger \mathbf{f}(\mathbf{X}) + \frac{1}{2}\mathbf{q}^\dagger \mathbf{H}(\mathbf{X})\mathbf{q} + \cdots \tag{1.1}$$

$$\mathbf{f}(\mathbf{X}_1) = \mathbf{f}(\mathbf{X}) + \mathbf{H}(\mathbf{X})\mathbf{q} \tag{1.2}$$

where

$$\mathbf{q} = \mathbf{X}_1 - \mathbf{X} \qquad f_i = \frac{\partial E(\mathbf{X})}{\partial X_i} \qquad H_{ij} = \frac{\partial^2 E(\mathbf{X})}{\partial X_i \partial X_j} \tag{1.3}$$

Newton Method, aka Newton-Raphson Method

Let $\mathbf{X}_1 = \mathbf{X}_e$

$$\mathbf{f}(\mathbf{X}) = -\mathbf{H}(\mathbf{X})\mathbf{q} \tag{1.4}$$

i.e.

$$\mathbf{q} = -\mathbf{H}^{-1}(\mathbf{X})\mathbf{f}(\mathbf{X}) \tag{1.5}$$

Quadratic Convergence of Newton's Method

If f is continuously differentiable and its derivative is not 0 at α and it has a second derivative at α then the convergence is quadratic or faster.

$$\Delta x_{i+1} = \frac{f''(\alpha)}{2f'(\alpha)}(\Delta x_i)^2 + O(\Delta x_i)^3 \tag{1.6}$$

Quasi-Newton Optimization

Murtagh-Sargent Method, Szabo §C

$$\mathbf{q}_n = -\alpha_{n-1}\mathbf{G}_{n-1}\mathbf{f}_{n-1} \tag{1.7}$$

1. Set $\alpha_0 = 1$ and $\mathbf{G_0} = \mathbf{I}$. While $(E_1 > E_0)$ set $\alpha_0 \leftarrow \alpha_0/2$
2.
$$\mathbf{U}_k = -\alpha_{k-1}\mathbf{G}_{k-1}\mathbf{f}_{k-1} - \mathbf{G}_{k-1}(\mathbf{f}_k - \mathbf{f}_{k-1}) \tag{1.8}$$
$$a_k^{-1} = \mathbf{U}_k^\dagger \mathbf{d}_k = \mathbf{U}_k^\dagger(\mathbf{f}_k - \mathbf{f}_{k-1}) \tag{1.9}$$
$$T_k = \mathbf{U}_k^\dagger \mathbf{U}_k \tag{1.10}$$
if $a_k^{-1} < 10^{-5} T_k$ or $a_k \mathbf{U}_k^\dagger \mathbf{f}_{k-1} > 10^{-5}$, goto step 1
else
$$\mathbf{G}_k = \mathbf{G}_{k-1} + a_k \mathbf{U}_k \mathbf{U}_k^\dagger \tag{1.11}$$
$$\alpha_k = 1 \tag{1.12}$$
3.
$$\mathbf{q}_k = -\alpha_{k-1}\mathbf{G}_{k-1}\mathbf{f}_{k-1} \tag{1.13}$$

Quasi-Newton Optimization

Method	$B_{k+1} =$	$H_{k+1} = B_{k+1}^{-1} =$
BFGS	$B_k + \frac{y_k y_k^{\mathrm{T}}}{y_k^{\mathrm{T}} \Delta x_k} - \frac{B_k \Delta x_k (B_k \Delta x_k)^{\mathrm{T}}}{\Delta x_k^{\mathrm{T}} B_k \, \Delta x_k}$	$\left(I - \frac{\Delta x_k y_k^{\mathrm{T}}}{y_k^{\mathrm{T}} \Delta x_k}\right) H_k \left(I - \frac{y_k \Delta x_k^{\mathrm{T}}}{y_k^{\mathrm{T}} \Delta x_k}\right) + \frac{\Delta x_k \Delta x_k^{\mathrm{T}}}{y_k^{\mathrm{T}} \, \Delta x_k}$
Broyden	$B_k + \frac{y_k - B_k \Delta x_k}{\Delta x_k^{\mathrm{T}} \, \Delta x_k} \, \Delta x_k^{\mathrm{T}}$	$H_k + \frac{(\Delta x_k - H_k y_k) \Delta x_k^{\mathrm{T}} H_k}{\Delta x_k^{\mathrm{T}} H_k \, y_k}$
Broyden family	$(1 - \varphi_k) B_{k+1}^{\mathrm{BFGS}} + \varphi_k B_{k+1}^{\mathrm{DFP}}, \quad \varphi \in [0, 1]$	
DFP	$\left(I - \frac{y_k \, \Delta x_k^{\mathrm{T}}}{y_k^{\mathrm{T}} \, \Delta x_k}\right) B_k \left(I - \frac{\Delta x_k y_k^{\mathrm{T}}}{y_k^{\mathrm{T}} \, \Delta x_k}\right) + \frac{y_k y_k^{\mathrm{T}}}{y_k^{\mathrm{T}} \, \Delta x_k}$	$H_k + \frac{\Delta x_k \Delta x_k^{\mathrm{T}}}{\Delta x_k^{\mathrm{T}} \, y_k} - \frac{H_k y_k y_k^{\mathrm{T}} H_k}{y_k^{\mathrm{T}} H_k y_k}$
SR1	$B_k + \frac{(y_k - B_k \, \Delta x_k)(y_k - B_k \, \Delta x_k)^{\mathrm{T}}}{(y_k - B_k \, \Delta x_k)^{\mathrm{T}} \, \Delta x_k}$	$H_k + \frac{(\Delta x_k - H_k y_k)(\Delta x_k - H_k y_k)^{\mathrm{T}}}{(\Delta x_k - H_k y_k)^{\mathrm{T}} y_k}$

2nd-Order Hartree-Fock

```
            approx. NR    full NR
Gaussian:                 scf=qc
ORCA:        SOSCF        NRSCF
PySCF:                    scf.RHF().newton()
```

Orbital Rotation: Basics

matrix exponential

$$e^{\mathbf{A}} = \sum_{n=0}^{\infty} \frac{\mathbf{A}^n}{n!} \tag{1.14}$$

性质

$$\mathbf{B}\, e^{\mathbf{A}}\, \mathbf{B}^{-1} = e^{\mathbf{B}\mathbf{A}\mathbf{B}^{-1}} \tag{1.15}$$

$$e^{\mathbf{A}}\, \mathbf{B}\, e^{-\mathbf{A}} = \mathbf{B} + [\mathbf{A}, \mathbf{B}] + \frac{1}{2!}[\mathbf{A}, [\mathbf{A}, \mathbf{B}]] \tag{1.16}$$

exponential representation of unitary matrices

For any unitary matrix $\mathbf{U}$, we can always find an anti-Hermitian matrix $\mathbf{X}$ such that

$$\mathbf{U} = e^{\mathbf{X}} \quad \mathbf{X}^{\dagger} = -\mathbf{X} \tag{1.17}$$

Proof:

$$\mathbf{U}\mathbf{U}^{\dagger} = \mathbf{I} \Rightarrow \mathbf{X} + \mathbf{X}^{\dagger} = 0 \tag{1.18}$$

Unitary Orbital Transformation

$$\tilde{\phi}_p = \sum_q \phi_q U_{qp} \quad \text{or } \tilde{\mathbf{C}} = \mathbf{CU} \tag{1.19}$$

the unitary matrix $\mathbf{U}$ may be written in terms of an anti-Hermitian matrix $\boldsymbol{\kappa}$ as

$$\mathbf{U} = \mathrm{e}^{-\boldsymbol{\kappa}} \tag{1.20}$$

operator form

$$\mathrm{e}^{\hat{\kappa}} = \sum_{n=0}^{\infty} \frac{\hat{\kappa}^n}{n!} \quad \hat{\kappa} = \sum_{pq} \kappa_{pq} a_p^\dagger a_q \tag{1.21}$$

Exponential Parameterization of Density Matrix

MO-based

$$\boldsymbol{\rho}(\boldsymbol{\kappa}) = \mathrm{e}^{-\boldsymbol{\kappa}}\, \boldsymbol{\rho}\, \mathrm{e}^{\boldsymbol{\kappa}} = \mathrm{e}^{-\boldsymbol{\kappa}} \begin{pmatrix} \mathbf{I} & 0 \\ 0 & 0 \end{pmatrix} \mathrm{e}^{\boldsymbol{\kappa}} \tag{1.22}$$

AO-based

$$\begin{aligned} \mathbf{R}(\boldsymbol{\kappa}) &= \mathbf{C}\boldsymbol{\rho}(\boldsymbol{\kappa})\mathbf{C}^T = \mathbf{C}\,\mathrm{e}^{-\boldsymbol{\kappa}}\, \boldsymbol{\rho}\, \mathrm{e}^{\boldsymbol{\kappa}}\, \mathbf{C}^T \\ \mathbf{R} &= \mathbf{C}\boldsymbol{\rho}\mathbf{C}^T \end{aligned} \tag{1.23}$$

$\therefore$

$$\begin{aligned} \mathbf{R}(\boldsymbol{\kappa}) &= \mathbf{C}\,\mathrm{e}^{-\boldsymbol{\kappa}} [\mathbf{C}^T\mathbf{S}\mathbf{C}]\boldsymbol{\rho}[\mathbf{C}\mathbf{S}\mathbf{C}^T]\,\mathrm{e}^{\boldsymbol{\kappa}}\, \mathbf{C}^T \\ &= \mathrm{e}^{-\mathbf{X}\mathbf{S}}\, \mathbf{R}\, \mathrm{e}^{\mathbf{S}\mathbf{X}} \qquad \text{with } \mathbf{X} = \mathbf{C}\boldsymbol{\kappa}\mathbf{C}^T \end{aligned} \tag{1.24}$$

BCH expansion

$$\mathbf{R}(\mathbf{X}) = \mathbf{R} + [\mathbf{R}, \mathbf{X}]_S + \frac{1}{2}[[\mathbf{R}, \mathbf{X}]_S, \mathbf{X}]_S \tag{1.25}$$

where $[\mathbf{R}, \mathbf{X}]_S = \mathbf{RSX} - \mathbf{XSR}$

Electronic Gradient

$$E = 2\,\mathrm{tr}\,\mathbf{hR} + \mathrm{tr}\,\mathbf{RG}(\mathbf{R}) \tag{1.26}$$

HF electronic gradient

$$\begin{aligned} E(\mathbf{X}) &= E^{(0)} + 2\,\mathrm{tr}\,\mathbf{h}[\mathbf{R},\mathbf{X}]_S + 2\,\mathrm{tr}\,\mathbf{h}\frac{1}{2}[[\mathbf{R},\mathbf{X}]_S,\mathbf{X}]_S + \mathrm{tr}[\mathbf{R},\mathbf{X}]_S\mathbf{G}(\mathbf{R}) + \mathrm{tr}\,\mathbf{RG}([\mathbf{R},\mathbf{X}]_S) \\ &\quad + \frac{1}{2}\,\mathrm{tr}[[\mathbf{R},\mathbf{X}]_S,\mathbf{X}]_S\mathbf{G}(\mathbf{R}) + \frac{1}{2}\,\mathrm{tr}\,\mathbf{RG}([[\mathbf{R},\mathbf{X}]_S,\mathbf{X}]_S) + \mathrm{tr}[\mathbf{R},\mathbf{X}]_S\mathbf{G}([\mathbf{R},\mathbf{X}]_S) \\ &= E^{(0)} + 2\,\mathrm{tr}\,\mathbf{F}[\mathbf{R},\mathbf{X}]_S + \mathrm{tr}\,\mathbf{F}[[\mathbf{R},\mathbf{X}]_S,\mathbf{X}]_S + \mathrm{tr}[\mathbf{R},\mathbf{X}]_S\mathbf{G}([\mathbf{R},\mathbf{X}]_S) + \cdots \end{aligned} \tag{1.27}$$

$$E^{(1)}_{\mu\nu} = \frac{\partial}{\partial X_{\mu\nu}} 2\,\mathrm{tr}\,\mathbf{F}[\mathbf{R},\mathbf{X}]_S = 4(\mathbf{SRF} - \mathbf{FRS})_{\mu\nu} \tag{1.28}$$

```python
def get_grad(mo_coeff, mo_occ, fock_ao):
    '''RHF orbital gradients
    Returns:
        Gradients in MO representation.  It's a num_occ*num_vir vector.
    '''
    occidx = mo_occ > 0
    viridx = ~occidx
    g = reduce(numpy.dot, (mo_coeff[:,viridx].conj().T, fock_ao,
        mo_coeff[:,occidx])) * 2
    return g.ravel()
```

orbital-based SCF

Diag $\mathbf{F}$ – $\mathbf{R} = \mathbf{C}^T\mathbf{C}$ – ...

dm-based SCF

$$E = 2\,\mathrm{tr}\,\mathbf{RF} \tag{1.29}$$

$$\mathbf{R}_{n+1} = \mathrm{e}^{-\mathbf{X}_n\mathbf{S}}\,\mathbf{R}_n\,\mathrm{e}^{\mathbf{S}\mathbf{X}_n} \tag{1.30}$$

SuperCI

Newton-Raphson AugHess

$$\begin{pmatrix} 0 & \mathbf{g} \\ \mathbf{g} & \mathbf{H} \end{pmatrix} \begin{pmatrix} 1 \\ \mathbf{x} \end{pmatrix} = \varepsilon \begin{pmatrix} 1 \\ \mathbf{x} \end{pmatrix} \tag{1.31}$$

$$\mathbf{R} = \begin{pmatrix} 0 & \mathbf{x} \\ -\mathbf{x} & 0 \end{pmatrix} \tag{1.32}$$

```python
eris = casscf.ao2mo(mo)
e_tot, e_cas, fcivec = casscf.casci(mo, ci0, eris, log, locals())
casdm1, casdm2 = casscf.fcisolver.make_rdm12(fcivec, ncas, casscf.nelecas)

while not conv and imacro < casscf.max_cycle_macro:
    imacro += 1;    imicro = 0
    rota = casscf.rotate_orb_cc(mo, lambda:fcivec, lambda:casdm1, lambda:casdm2,
                   eris, r0, conv_tol_grad*.3, max_stepsize, log)
    for u, g_orb, njk, r0 in rota:
        imicro += 1
        norm_gorb = numpy.linalg.norm(g_orb)
        if imicro == 1:   norm_gorb0 = norm_gorb
        norm_t = numpy.linalg.norm(u-numpy.eye(nmo))

        casdm1, casdm2, gci, fcivec = casscf.update_casdm(mo, u, fcivec, e_cas,
            eris, locals())

        norm_gci = numpy.linalg.norm(gci)
        log.debug('micro %d  |u-1|=%5.3g  |g[o]|=%5.3g  |g[c]|=%5.3g  |ddm|=%5.3g
            ', imicro, norm_t, norm_gorb, norm_gci, norm_ddm)
```

```
rota = casscf.rotate_orb_cc(mo, lambda:fcivec, lambda:casdm1, lambda:casdm2,
eris, r0, conv_tol_grad*.3, max_stepsize, log)
```

Thank You

Acknowledgment

– Prof. A
– Mr. B
– Dr. C
– Mr. D