Basic Theory

N-body Schrödinger equation: $H\Psi(r_1, \cdots, r_N) = E\Psi(r_1, \cdots, r_N)$

- *Ab Initio:* seeking wavefunction Ψ with $3N$ DOFs
- computationally unfeasible for large N
- Hohenberg-Kohn: electron density in ground state

$$n(r) = \int dr_2 \cdots dr_N \|\Psi_{\mathrm{GS}}(r, r_2, \cdots, r_N)\|$$

  determines all system-related observables
- Total energy: $E[n] = T_s[n] + T_{\mathrm{ext}}[n] + T_{\mathrm{ee}}[n] + E_{\mathrm{xc}}[n]$

Kohn-Sham Equation

Ideas:

- Introduce one-particle system with the same ground-state energy
- Find $n(r) = \sum_i |\psi_i(r)|^2$ such that it minimizes $E[n]$
- Normalization condition for each ψ_i
- Using method of Lagrange multiplier obtain Kohn-Sham equation:

$$H_{\mathrm{KS}}\psi_i = \varepsilon_i \psi_i,$$

$$H_{\mathrm{KS}} := -\frac{1}{2}\nabla^2 + V_{\mathrm{ext}}(r) + \int dr' \frac{n(r')}{|r - r'|} + V_{\mathrm{xc}}(r),$$

$$V_{\mathrm{xc}}(r) := \frac{\delta E_{\mathrm{xc}}[n]}{\delta n(r)}.$$

- A nonlinear eigenvalue problem

KS-DFT Self-Consistent Field Iteration

Suppose XC potential (or functional) is given.

1. Start with an initial electron density $\rho = \rho_0$
2. Solve the KS equation with ρ to obtain $\{\psi_i\}$
3. Update electron density ρ with the new orbitals
4. If converges, output ρ and and do post-processing
5. If not, continue Step 2 with the new ρ

Post-processing: compute desired molecular properties

XC approximation via Machine Learning

Idea: Parameterize the XC potential via ML techniques.

Example: parametrize XC potential using CNN[1]

1. Size of H_2/HeH^+ Training Data Set
   - **Input data:** electron densities w/ spatial derivatives (generated by CCSD/aug-cc-pVQZ)
   - **Labels:** effective XC potentials by Wu-Yang (or ZMP) method
2. Each data: $(\rho, \rho_x, \rho_y, \rho_z)$ - $4 \times 9 \times 9 \times 9$ (values at LL quadrature)
3. **Model:** 2 CNN layers, 1 max pooling, flatten, 4 FC layers
4. Train: compute loss (forward) and update parameters (backward)

Once model obtained, perform SCF procedure.

[1] J. Phys. Chem. Lett. 2019, 10, 7264-7269

XC approximation via Machine Learning (Cont.)

Example: parametrize XC functional based on B3LYP[2]

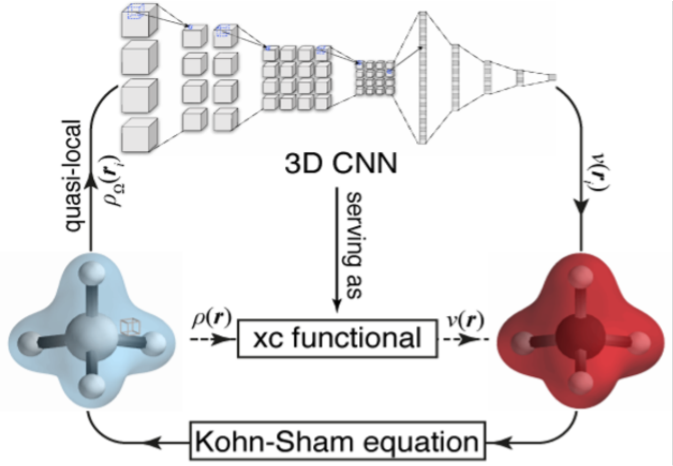
$$\begin{aligned} E_{XC}[\rho] := {} & a_0[\rho]E_X^{\text{Sla}}[\rho] + (1 - a_0[\rho])E_X^{\text{HF}}[\rho] \\ & + a_X[\rho]\Delta E_X^{\text{Bec}} + a_C[\rho]E_C^{\text{LYP}} + (1 - a_C[\rho])E_C^{\text{VWN}} \end{aligned}$$

1. Training Data Set
   - **Input data:** electron densities wo/ spatial derivatives (generated by CCSD/aug-cc-pVQZ)
   - **Labels:** effective XC potentials by Wu-Yang (or ZMP) method
2. Each data: 1×30 data size
3. **Model:** graphical neural network employed
4. Train: compute loss (forward) and update parameters (backward)

Once model obtained, compute XC potential and perform SCF procedure.

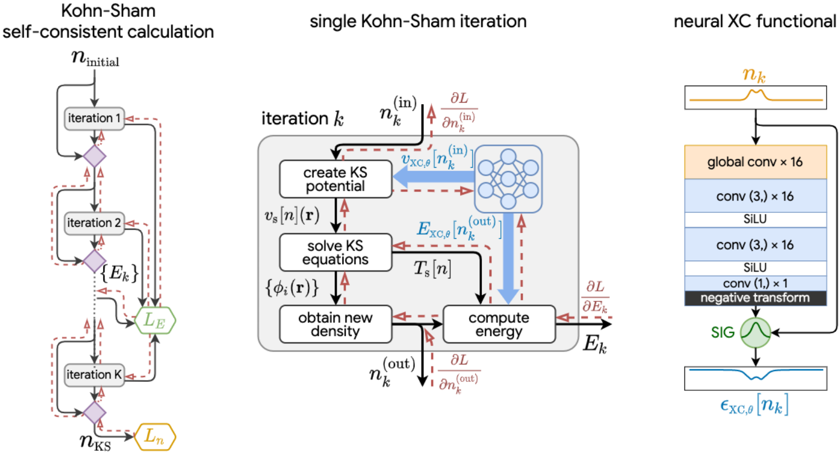
[2] Chem. Phys. Lett. 2004, 390, 186-192

Model Training and SCF



SCF loss: computed via AD w.r.t. spatial coordinates.

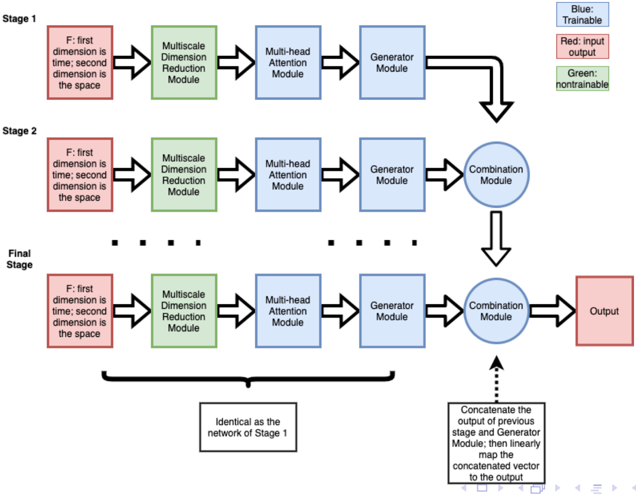
Combining training with SCF iteration



Multi-stage strategy

Model Problem:

$$u_t = \mathcal{F}(x, u, \nabla u; a, \xi),\ \mathcal{B}(x, u, \nabla u) = 0,\ a(x, u) := \kappa(x) \exp(\gamma u)$$



Stage 1

F: first dimension is time; second dimension is the space

Multiscale Dimension Reduction Module

Multi-head Attention Module

Generator Module

Blue: Trainable

Red: input output

Green: nontrainable

Stage 2

F: first dimension is time; second dimension is the space

Multiscale Dimension Reduction Module

Multi-head Attention Module

Generator Module

Combination Module

Final Stage

F: first dimension is time; second dimension is the space

Multiscale Dimension Reduction Module

Multi-head Attention Module

Generator Module

Combination Module

Output

Identical as the network of Stage 1

Concatenate the output of previous stage and Generator Module; then linearly map the concatenated vector to the output

Precise Demonstration

(m_1, r_1): hyper-parameters to be tuned; D_i: dimension reduction modules; M_i, C_i, and G_i: trainable modules; C_i: fully connected layers

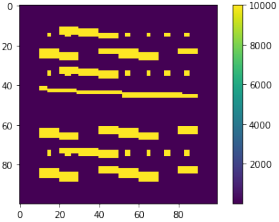
$$\begin{tikzcd}
F_0 \in \mathbb{R}^{m_0,n} \arrow[r, "D_1"] & F_{1,s_1} \in \mathbb{R}^{m_0,r_0} \arrow[r, "M_1"] & F_{2,s_1} \in \mathbb{R}^{m_1,r_1} \arrow[r, "G_1"] & u_{H,s_1} \in \mathbb{R}^l, \arrow[d] & \\
F_0 \in \mathbb{R}^{m_0,n} \arrow[r, "D_2"] & F_{1,s_2} \in \mathbb{R}^{m_0,r_0} \arrow[r, "M_2"] & F_{2,s_2} \in \mathbb{R}^{m_1,r_1} \arrow[r, "G_2"] & u_{H,s_2} \in \mathbb{R}^l \arrow[r, "C_2"] & u^c_{H,s_2} \in \mathbb{R}^l, \arrow[dl] \\
F_0 \in \mathbb{R}^{m_0,n} \arrow[r, "D_k"] & F_{1,s_k} \in \mathbb{R}^{m_0,r_0} \arrow[r, "M_k"] & F_{2,s_k} \in \mathbb{R}^{m_1,r_1} \arrow[r, "G_k"] & u_{H,s_k} \in \mathbb{R}^l \arrow[r, "C_k"] & u^c_{H,s_k} \in \mathbb{R}^l,
\end{tikzcd}$$

Numerical Demonstration: nonlinear parametrized PDE

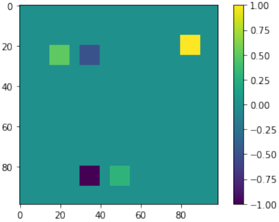
$$u_t = \nabla \cdot (a(x,u)\nabla u) + f_\xi \text{ with BC + IC, } a(x,u) := \kappa(x)\exp(\gamma u)$$

(m_1, r_1)	1^{st}	2^{nd}	3^{rd}
(30, 10)	.13322	.08701	.08439
(25, 12)	.11515	.08538	.08162
(20, 15)	.09814	.08788	.08210
(15, 20)	.12591	.08538	.08054
(10, 30)	.29105	.09888	.07815

Table: MRE with Decoupled Information



0
20
40
60
80
0
20
40
60
80
10000
8000
6000
4000
2000



0
20
40
60
80
0
20
40
60
80
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00