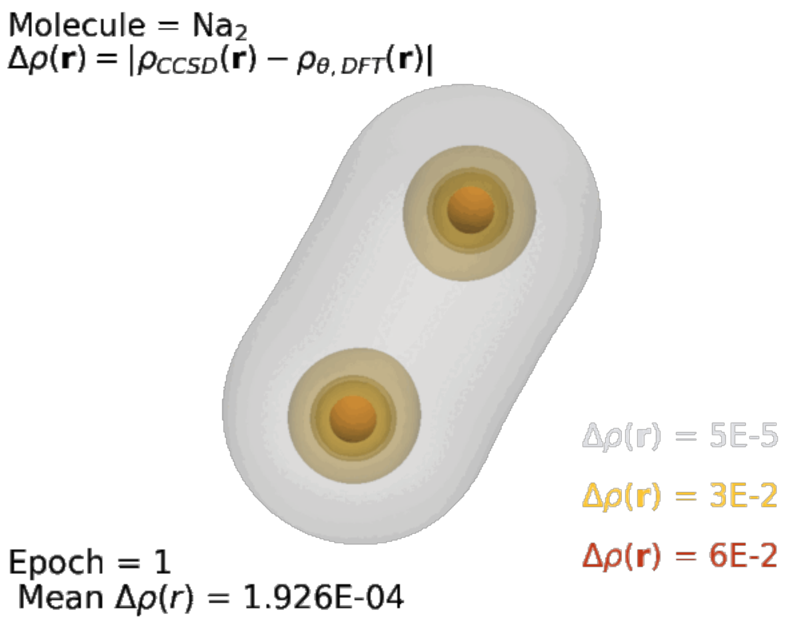
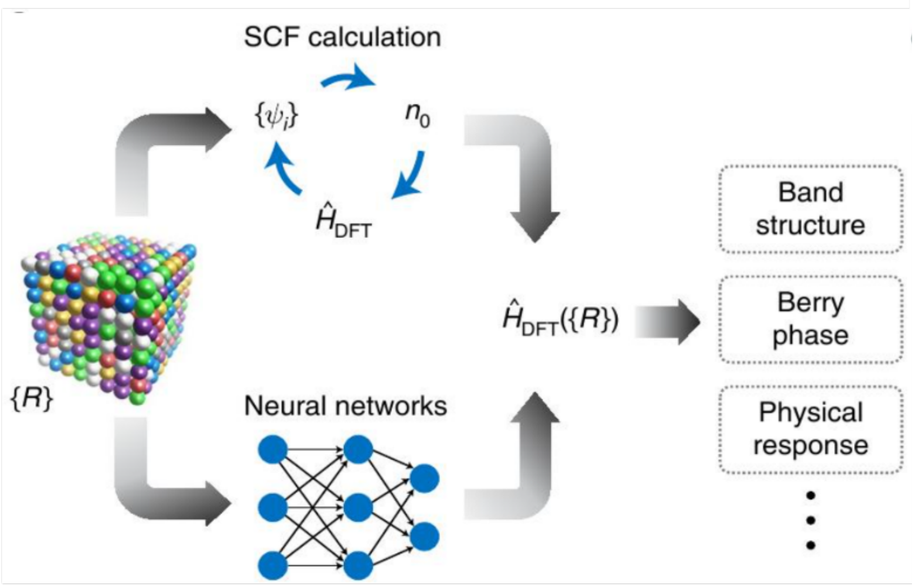
中国科学院大学 University of Chinese Academy of Sciences
前沿交叉科学学院 School of Advanced Interdisciplinary Sciences

Full Differentiable Exchange-Correlation Functional and Equivariant Graph Neural Network for Hamiltonian Prediction

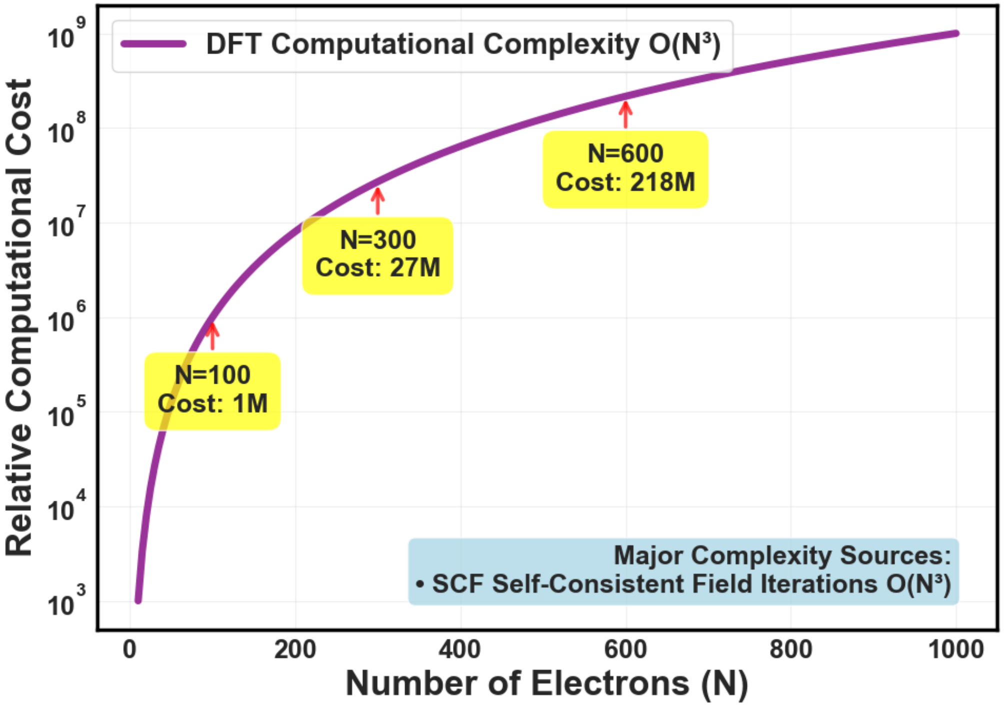
Yuan Jiao
School of Advanced Interdisciplinary Sciences, SAIS
University of Chinese Academy of Sciences

- **Full Differentiable Exchange-Correlation Functional**
- **Equivariant Graph Neural Network Architecture**
- **Graph Neural Network For Excited State**

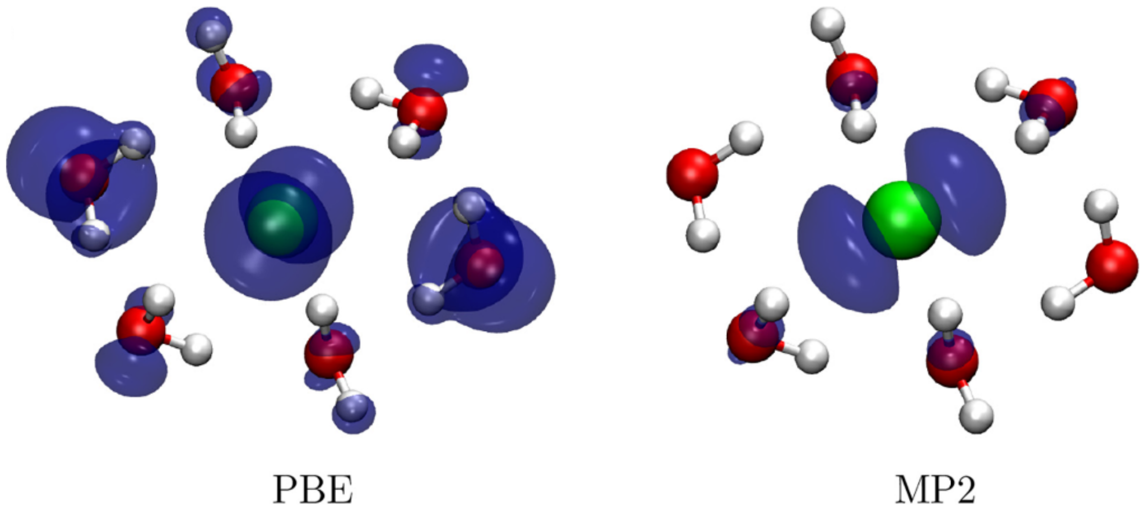




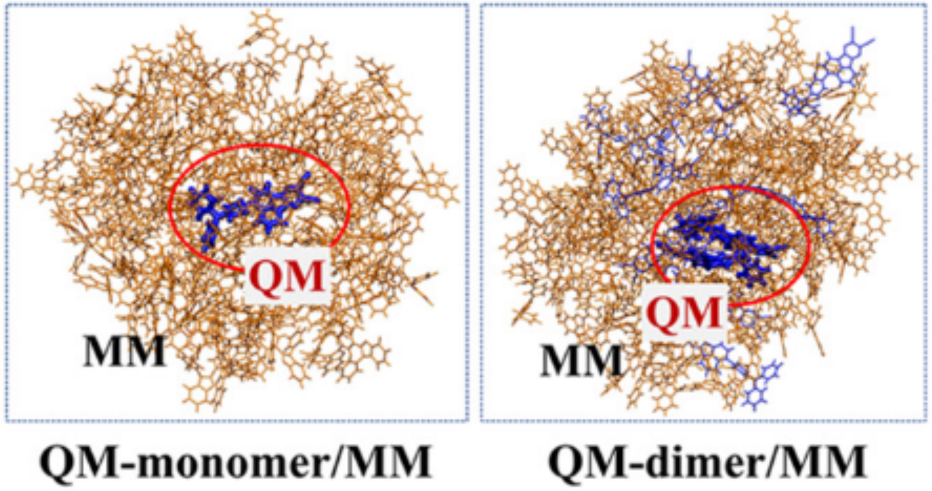
■ Background



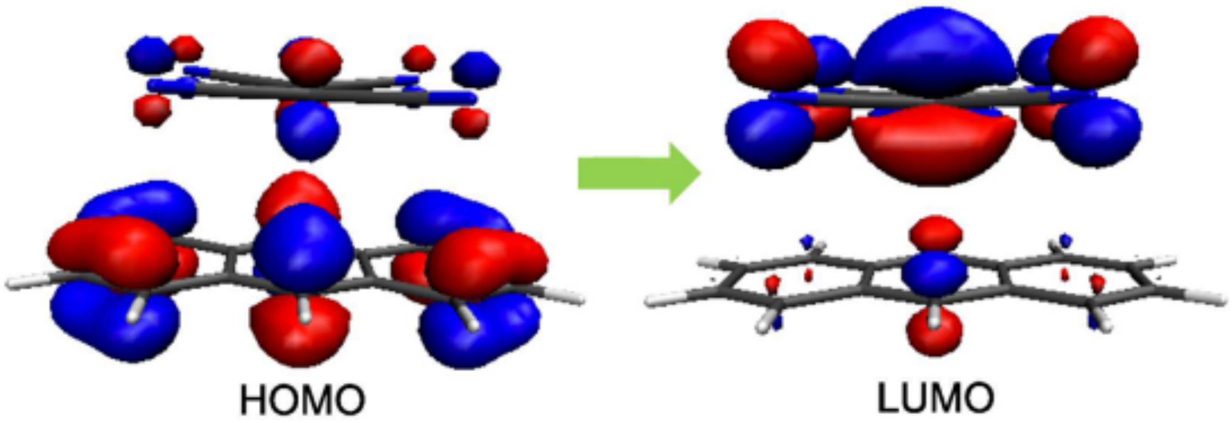
WIREs Comput Mol Sci. 2023;13(2):e1631



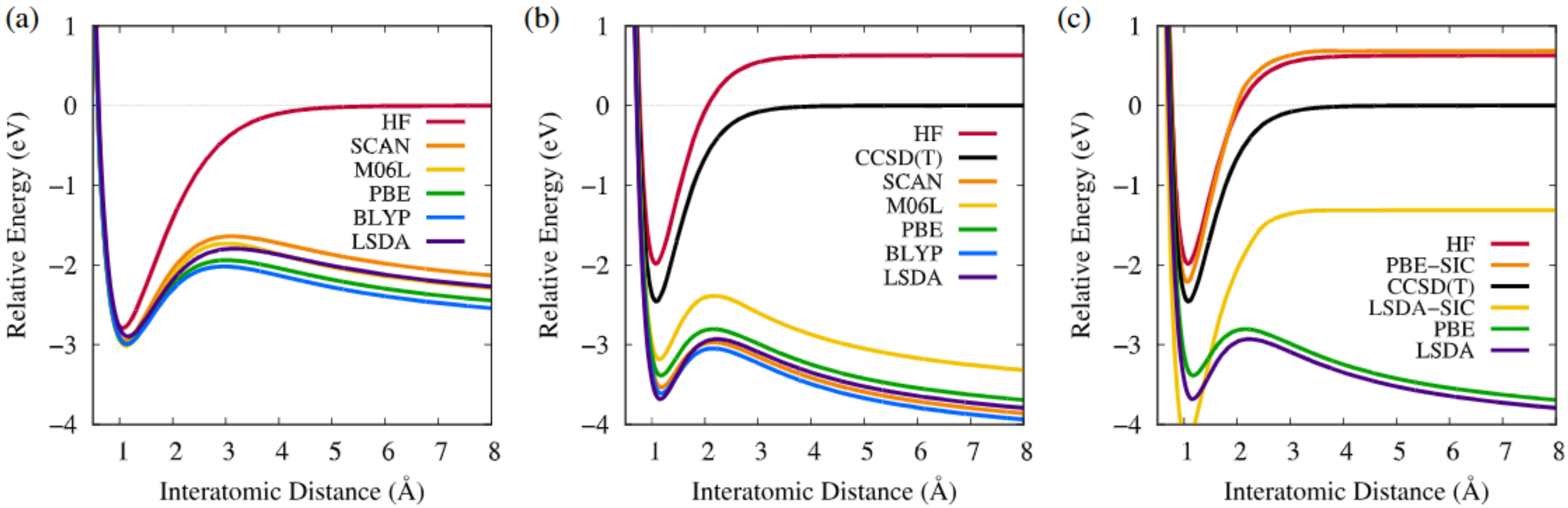
J. Chem. Theory Comput. 2019, 15, 8, 4305–4311



Phys Chem Chem Phys. 2024;26(26):18418-18425



▪ Exchange-Correlation Functional and Self-Interaction Error



N-electron system

$$v_{eff}(\boldsymbol{r},\boldsymbol{R}) = v_{ext}(\boldsymbol{R}) + \int d\boldsymbol{r}' \frac{\rho(\boldsymbol{r}')}{|\boldsymbol{r}-\boldsymbol{r}'|} + v_{xc}(\boldsymbol{r})$$

One-electron system($|\boldsymbol{r}| \to \infty$)

$$v_{eff}(\boldsymbol{r},\boldsymbol{R}) = v_{ext}(\boldsymbol{R}) + \frac{1}{|\boldsymbol{r}|} + v_{xc}(\boldsymbol{r})$$

▪ Full Differentiable Exchange-Correlation Functional – DQC and PySCFAD

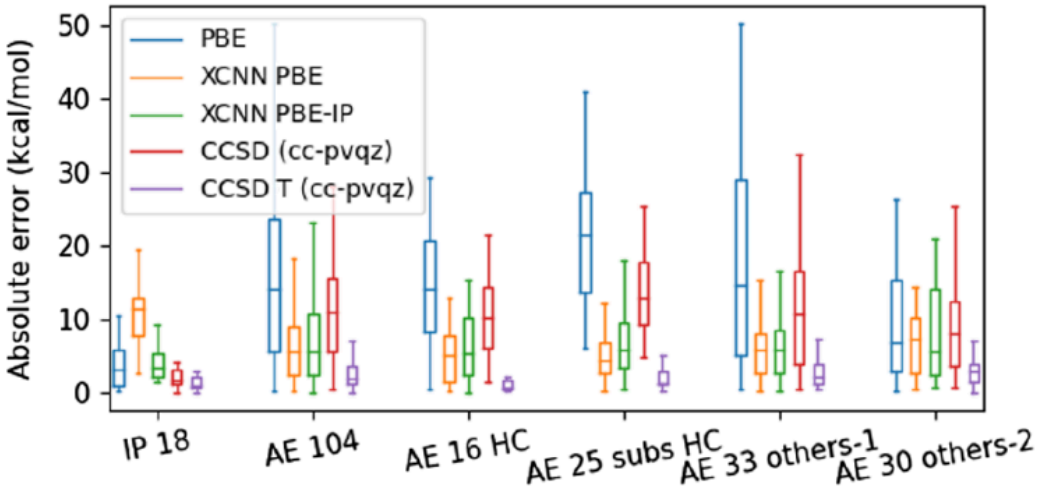
XC Library used in PySCFAD

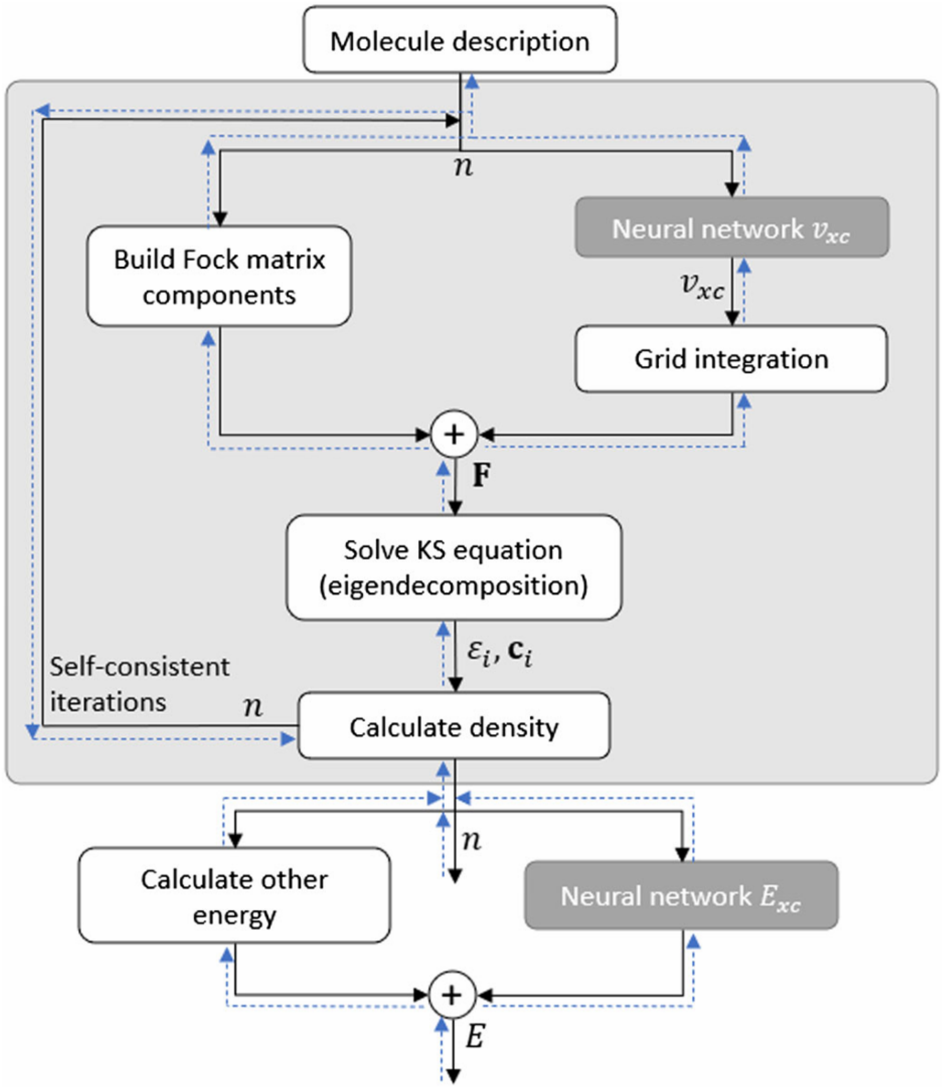
```
from functools import partial
from pyscf.dft import libxc
from pyscf.dft.libxc import parse_xc, is_lda, is_meta_gga
from pyscfad import numpy as np
from pyscfad.ops import jit, custom_jvp
```

XC Functional in PySCFAD

$$E_{xc}^{XCNN} = \alpha E_{LDA}^{xc} + \beta \int dr\ f(n, \xi, s)\rho(\mathbf{r})$$

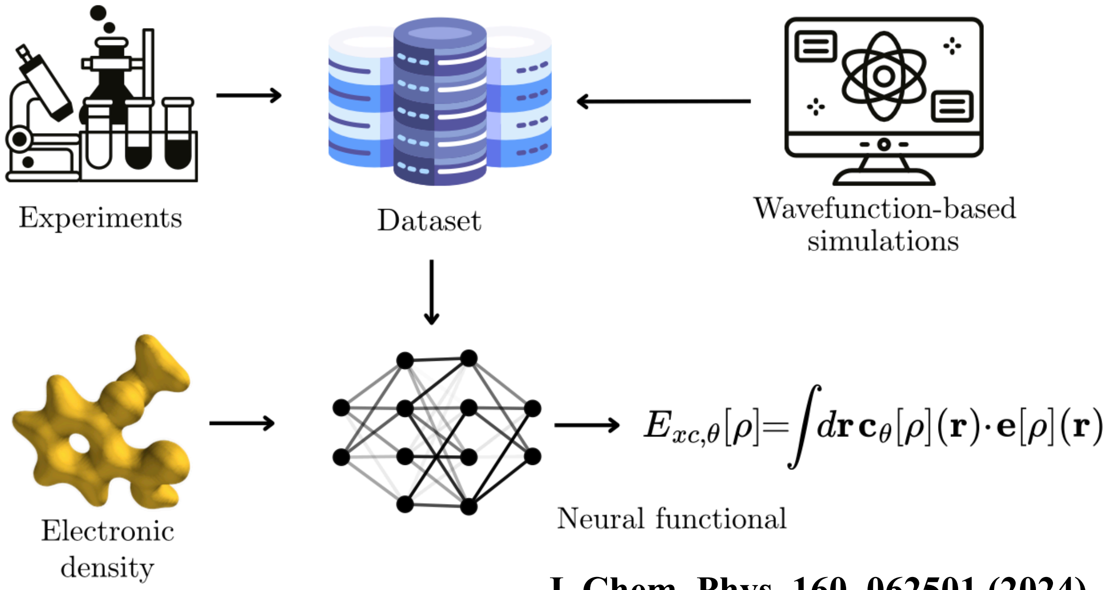
$$E_{xc}^{XCNN} = \alpha E_{GGA}^{xc} + \beta \int dr\ f(n, \xi, s)\rho(\mathbf{r})$$





Phys. Rev. Lett. 127, 126403

■ Full Differentiable Exchange-Correlation Functional – GradDFT Package



J. Chem. Phys. 160, 062501 (2024)

Coeffcient Functions

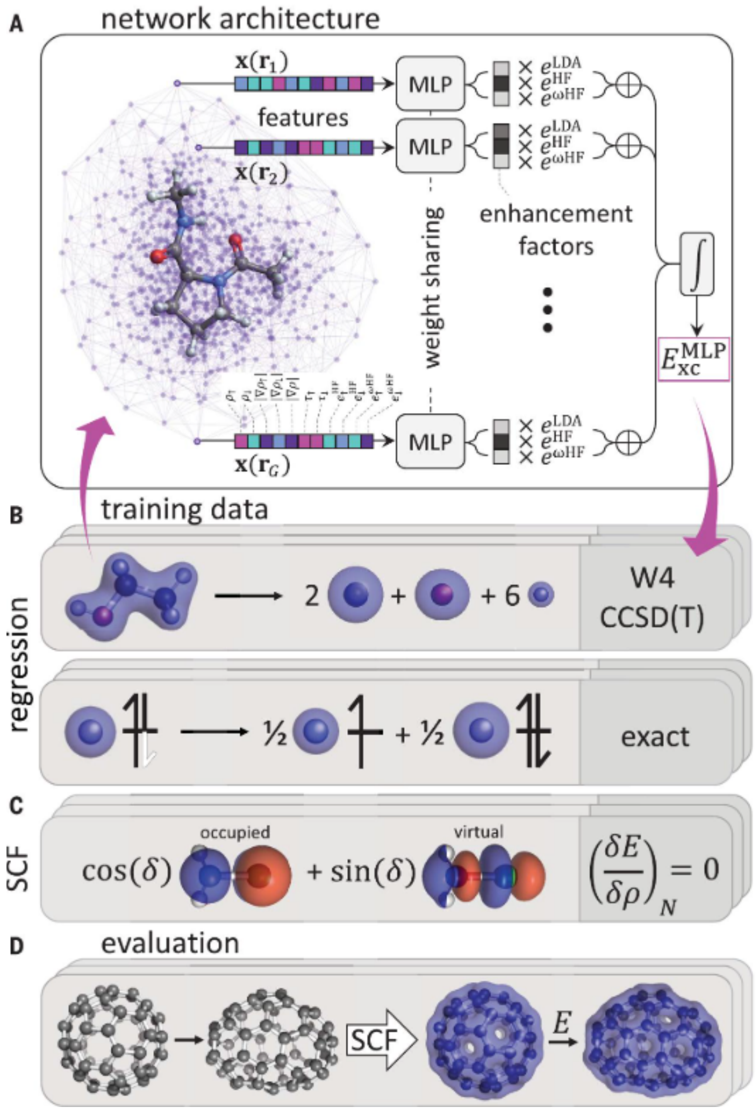
$$c_\theta[\rho](\boldsymbol{r}) = (c_{1,\theta}[\rho](\boldsymbol{r}), c_{2,\theta}[\rho](\boldsymbol{r})\ldots c_{k,\theta}[\rho](\boldsymbol{r}))$$

Energy Densities

$$e[\rho](\boldsymbol{r}) = (e_1[\rho](\boldsymbol{r}), e_2[\rho](\boldsymbol{r})\ldots e_k[\rho](\boldsymbol{r}))$$

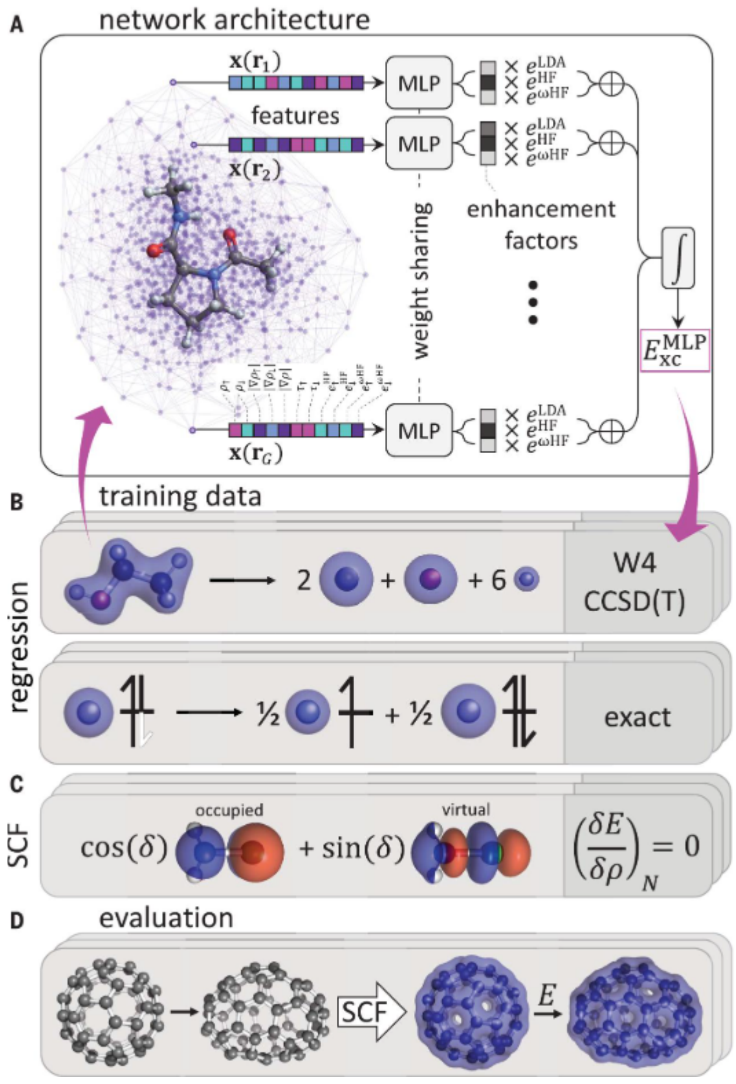
Loss Function

$$L = \frac{1}{N^2}\sum_i \left(E_{pre} - E_{true}\right)^2 + \frac{1}{N^2}\sum_i \left(\rho_{pre} - \rho_{true}\right)^2$$



***Science*. 2021;374(6573):1385-1389**

Full Differentiable Exchange-Correlation Functional – GradDFT Package



Hatree-Fock Energy Density

$e^{\omega}_{HF}[\rho](\boldsymbol{r})$ **with** $\omega = 0, 0.5$

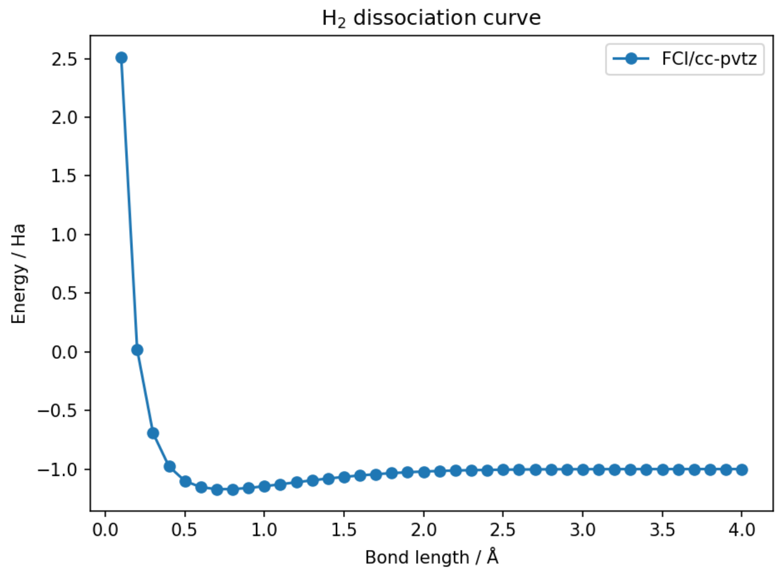
Electronic Density

$\rho_\alpha,\ \rho_\beta,\ |\nabla\rho|^2,\ |\nabla\rho_\alpha|^2,\ |\nabla\rho_\beta|^2,$

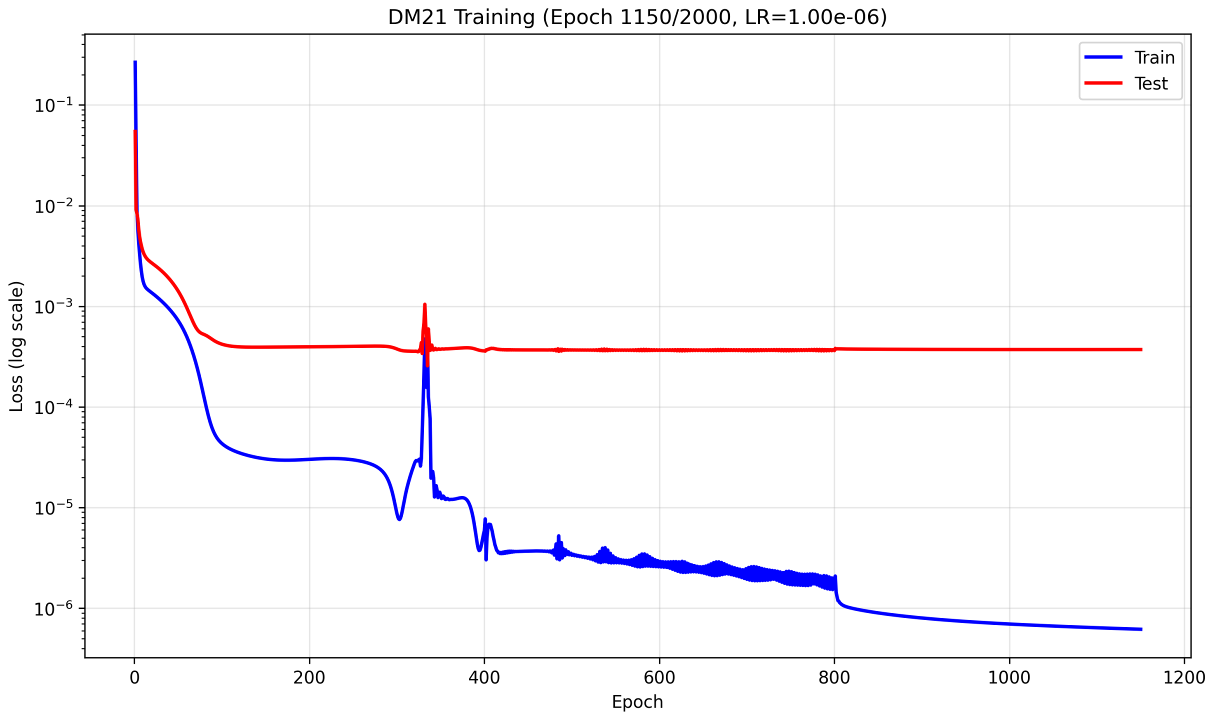
DFT Initial → Features on the Grid → MLP → **Coeffcient Functions**

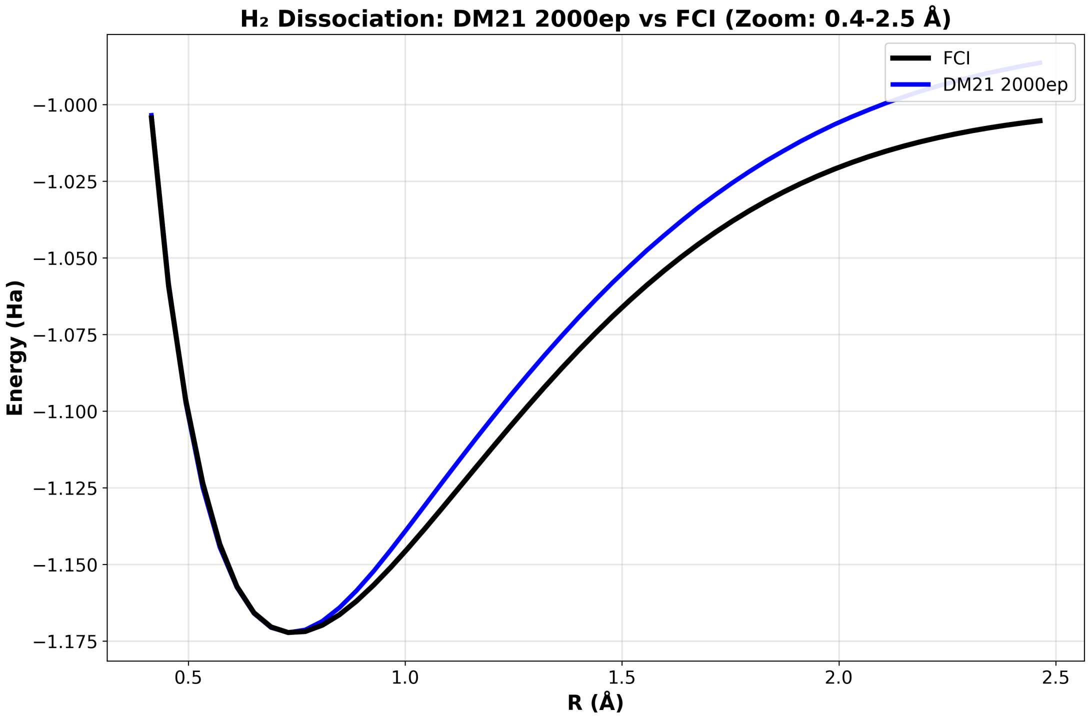
$$E^{DM21}_{xc} = \int d\boldsymbol{r} c_\theta[\rho](\boldsymbol{r}) \cdot \begin{pmatrix} e^{xc}_{LDA}[\rho](\boldsymbol{r}) \\ e_{HF}[\rho](\boldsymbol{r}) \\ e^{\omega}_{HF}[\rho](\boldsymbol{r}) \end{pmatrix}$$

$$E^{RSH}_{xc} = \int d\boldsymbol{r} c_\theta[\rho](\boldsymbol{r}) \cdot \begin{pmatrix} e^{x}_{LDA}[\rho](\boldsymbol{r}) \\ e^{x}_{B88}[\rho](\boldsymbol{r}) \\ e^{c}_{LYP}[\rho](\boldsymbol{r}) \\ e^{c}_{VMN}[\rho](\boldsymbol{r}) \\ e^{x}_{HF}[\rho](\boldsymbol{r}) \\ e^{\omega}_{HF}[\rho](\boldsymbol{r}) \end{pmatrix}$$



▪ Full Differentiable Exchange-Correlation Functional





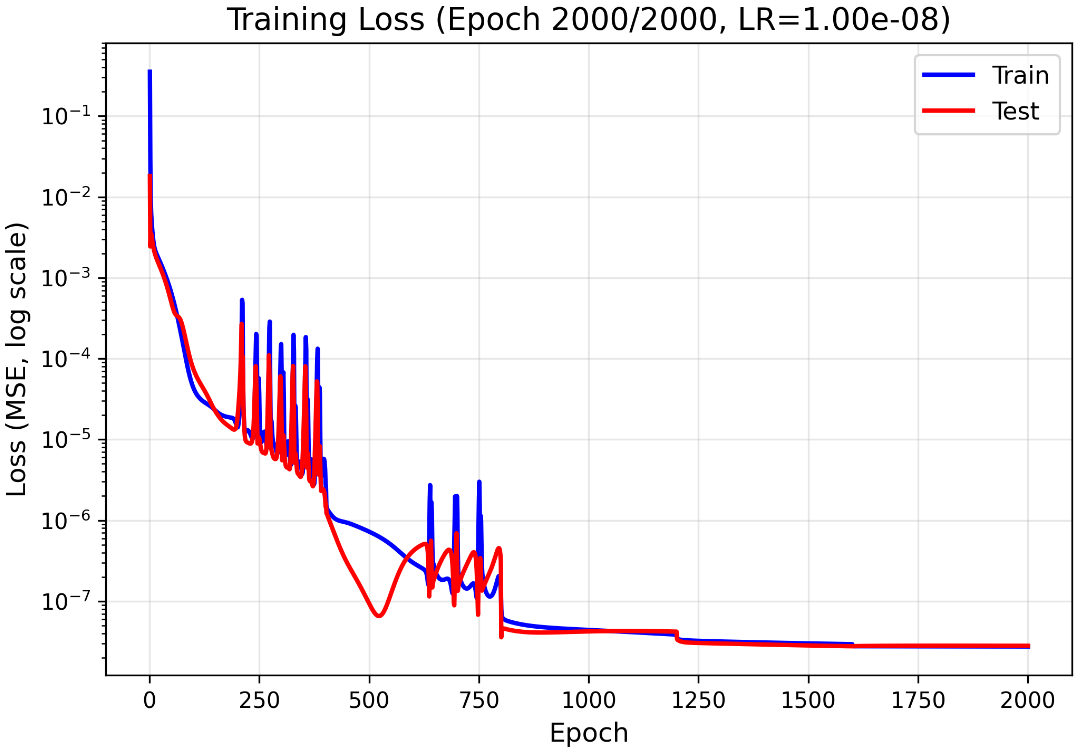
LDA XC Functional with HF Hybrid - DM21

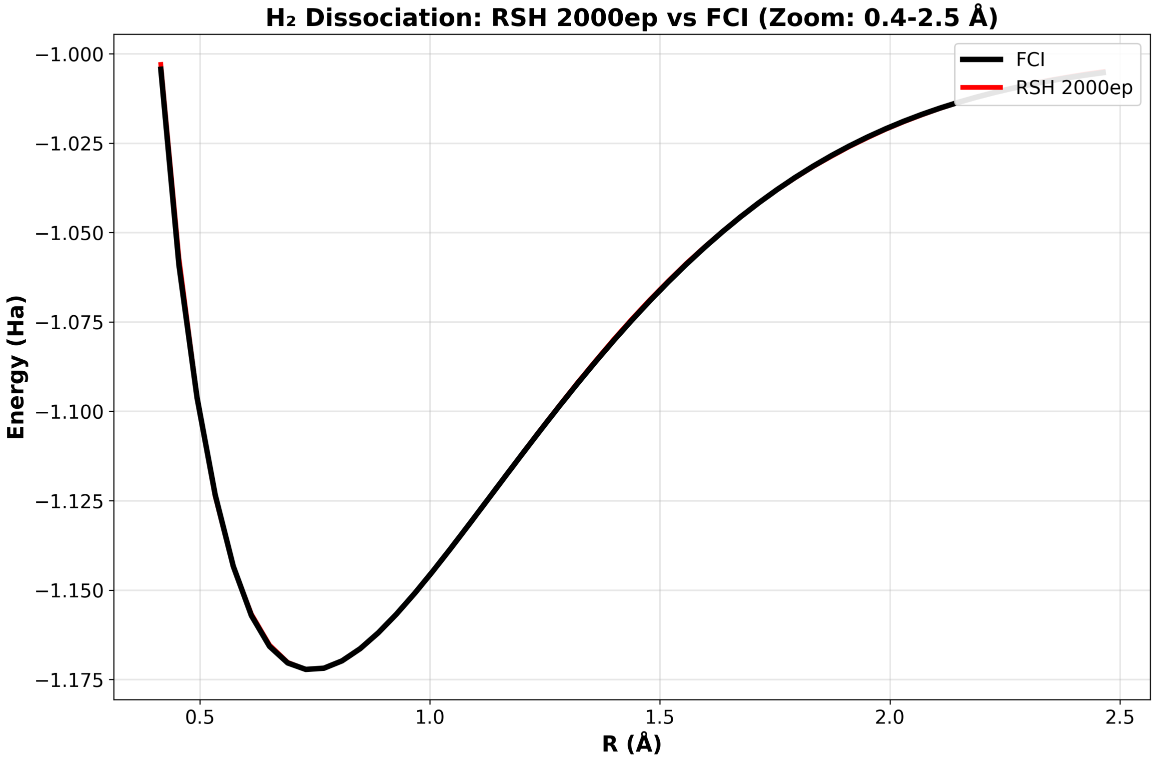
$$E_{xc}^{LDA} = a_x E_x^{LDA} + a_c E_c^{VMN} + \mathrm{a}_0(\mathrm{E}_\mathrm{x}^\mathrm{HF} - \mathrm{E}_\mathrm{LDA}^\mathrm{x})$$

$$E_{xc}^{DM21} = \int d\boldsymbol{r} \boldsymbol{c}_\theta[\rho](\boldsymbol{r}) \cdot \begin{pmatrix} e_{LDA}^{xc}[\rho](\boldsymbol{r}) \\ e_{HF}[\rho](\boldsymbol{r}) \\ e_{HF}^{\omega}[\rho](\boldsymbol{r}) \end{pmatrix}$$

$$e_{HF}^{\omega}\big[\rho\big](\boldsymbol{r}) = \sum_{ij} \boldsymbol{\phi}_i^*(\boldsymbol{r}) \frac{\mathrm{erf}\Big(\omega[\rho]\big|r - r'\big|\Big)}{\big|r - r'\big|} \boldsymbol{\phi}_j(\boldsymbol{r})$$

■ Full Differentiable Exchange-Correlation Functional – H_2 Dissociation Curve





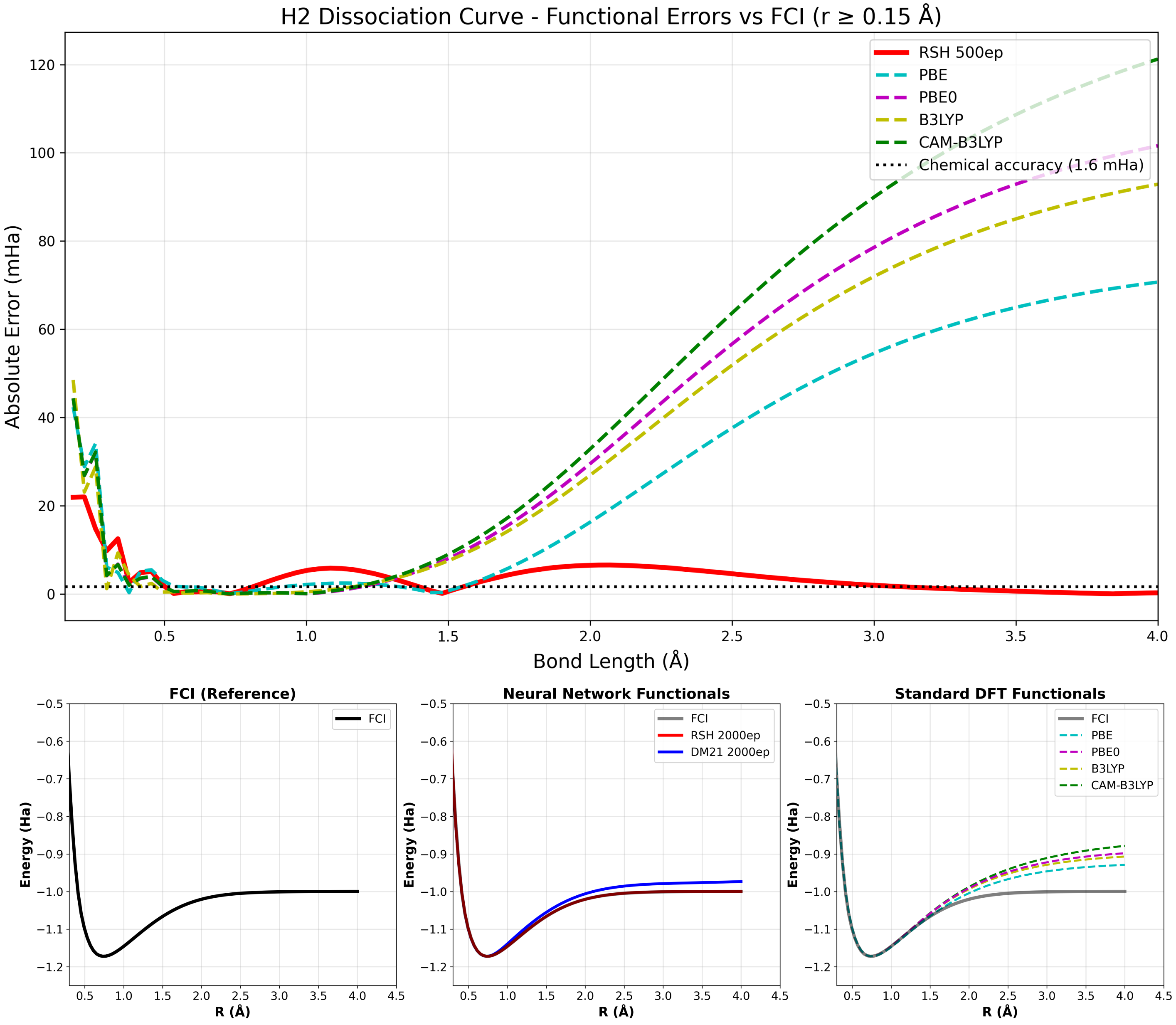
B3LYP XC Functional - Ours

$$E_{xc}^{B3LYP} = E_x^{LDA} + a_0\left(E_{HF}^x - E_x^{LDA}\right) + a_x\left(E_{B88}^x - E_{LDA}^x\right) + a_c(E_{LYP}^c - E_{VMN}^c)$$

$\longrightarrow$

$$E_{xc}^{RSH} = \int d\boldsymbol{r} c_\theta[\rho](\boldsymbol{r}) \cdot \begin{pmatrix} e_{LDA}^x[\rho](\boldsymbol{r}) \\ e_{B88}^x[\rho](\boldsymbol{r}) \\ e_{LYP}^c[\rho](\boldsymbol{r}) \\ e_{VMN}^c[\rho](\boldsymbol{r}) \\ e_{HF}^x[\rho](\boldsymbol{r}) \\ e_{HF}^\omega[\rho](\boldsymbol{r}) \end{pmatrix}$$

▪ Full Differentiable Exchange-Correlation Functional – H_2 Dissociation Curve



Neural Network XC Functional

	MAE(mHa)	RMSE(mHa²)	MAX(mHa)
RSH	1.365	6.222	46.709
DM21	15.519	17.833	49.705

Traditional XC Functional

	MAE(mHa)	RMSE(mHa²)	MAX(mHa)
PBE	29.185	39.142	70.685
PBE0	43.314	56.551	101.575
B3LYP	42.756	51.816	92.863
CAM-B3lYP	49.079	65.499	121.240

- # Full Differentiable Exchange-Correlation Functional – Response Kernal Calculation

XC Functional Code in PySCF and PySCFAD

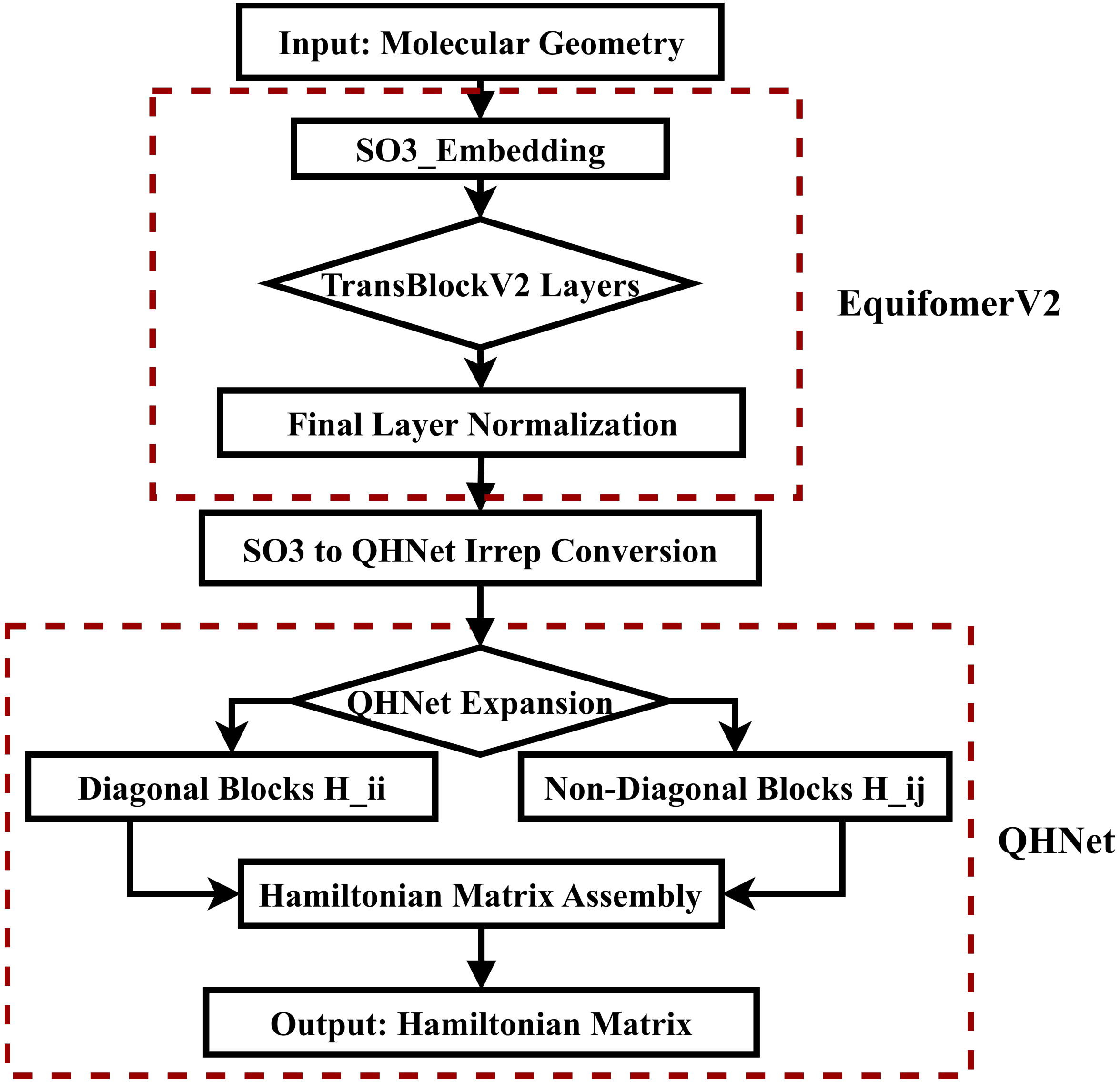
```python
def eval_gga_xc(xc_code, rho, spin=0, relativity=0, deriv=1, omega=None, verbose=None):
    # A fictitious XC functional to demonstrate the usage
    rho0, dx, dy, dz = rho
    gamma = (dx**2 + dy**2 + dz**2)
    exc = .01 * rho0**2 + .02 * (gamma+.001)**.5
    vrho = .01 * 3 * rho0**2 + .02 * (gamma+.001)**.5
    vgamma = .02 * .5 * (gamma+.001)**(-.5)
    vxc = (vrho, vgamma)
    v2rho2 = 0.01 * 6 * rho0
    v2rhosigma = np.zeros(gamma.shape)
    v2sigma2 = 0.02 * .5 * -.5 * (gamma+.001)**(-1.5)
    # 2nd order functional derivative
    fxc = (v2rho2, v2rhosigma, v2sigma2)
    kxc = None  # 3rd order functional derivative

    # Mix with existing functionals
    pbe_xc = dft.libxc.eval_xc('pbe,pbe', rho, spin, relativity, deriv, verbose)
    exc += pbe_xc[0] * 0.5
    vrho += pbe_xc[1][0] * 0.5
    vgamma += pbe_xc[1][1] * 0.5
    # The output follows the libxc.eval_xc API convention
    return exc, vxc, fxc, kxc
```

XC Functional Code in GradDFT

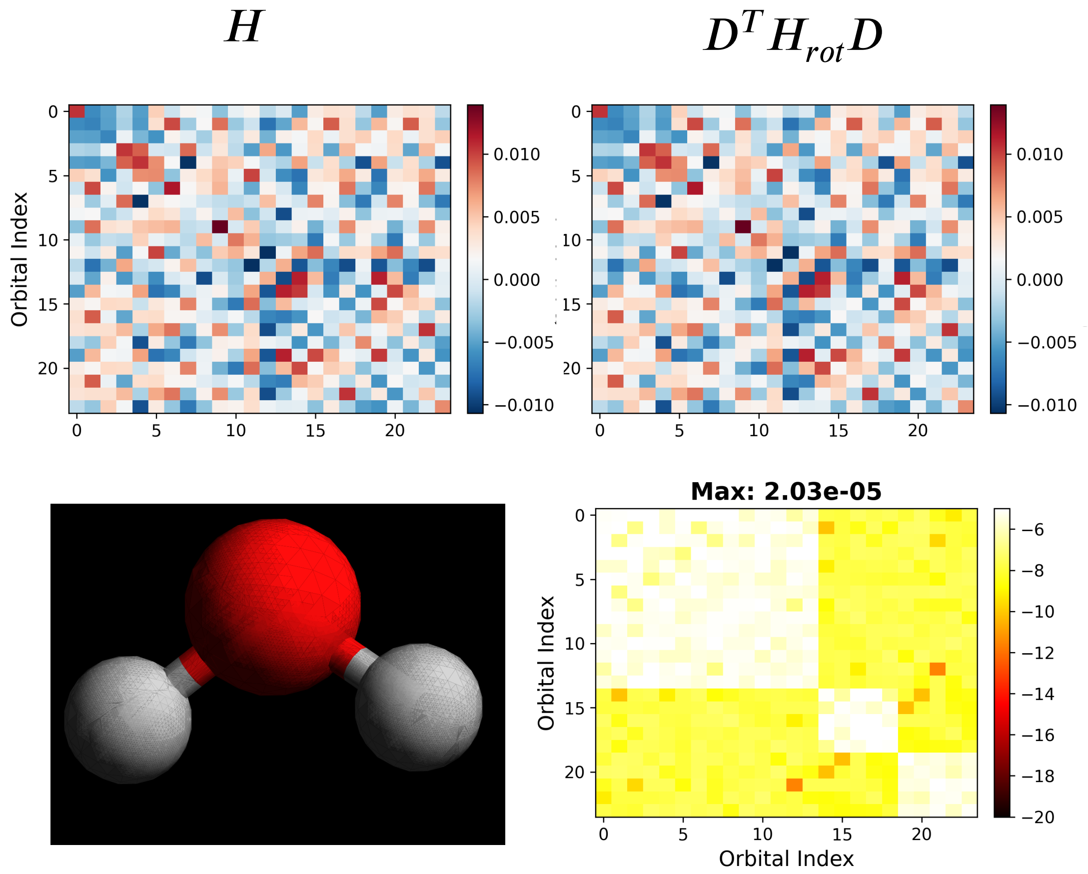
```python
def vwn_c_e(rho, clip_cte=1e-30):
    A = jnp.array([[0.0621814, 0.0621814 / 2]])
    b = jnp.array([[3.72744, 7.06042]])
    c = jnp.array([[12.9352, 18.0578]])
    x0 = jnp.array([[-0.10498, -0.325]])
    rho = jnp.where(rho > clip_cte, rho, 0.0)
    log_rho = jnp.log2(jnp.clip(rho.sum(axis=1, keepdims=True), a_min=clip_cte))
    rs = 2 ** (jnp.log2((3 / (4 * jnp.pi)) ** (1 / 3)) - log_rho / 3.0)
    x = rs**0.5
    X = x**2 + b * x + c
    X0 = x0**2 + b * x0 + c
    Q = jnp.sqrt(4 * c - b**2)
    e_PF = A / 2 * (2 * jnp.log(x) - jnp.log(X) + 2 * b / Q *
                    jnp.arctan(Q / (2 * x + b)) - b * x0 / X0 * (
                    jnp.log((x - x0) ** 2 / X) + 2 * (2 * x0 + b) / Q *
                    jnp.arctan(Q / (2 * x + b))))
    return correlation_polarization_correction(e_PF, rho, clip_cte) * rho.sum(axis=1)
```

Exchange correlation functionals translated from libxc to jax

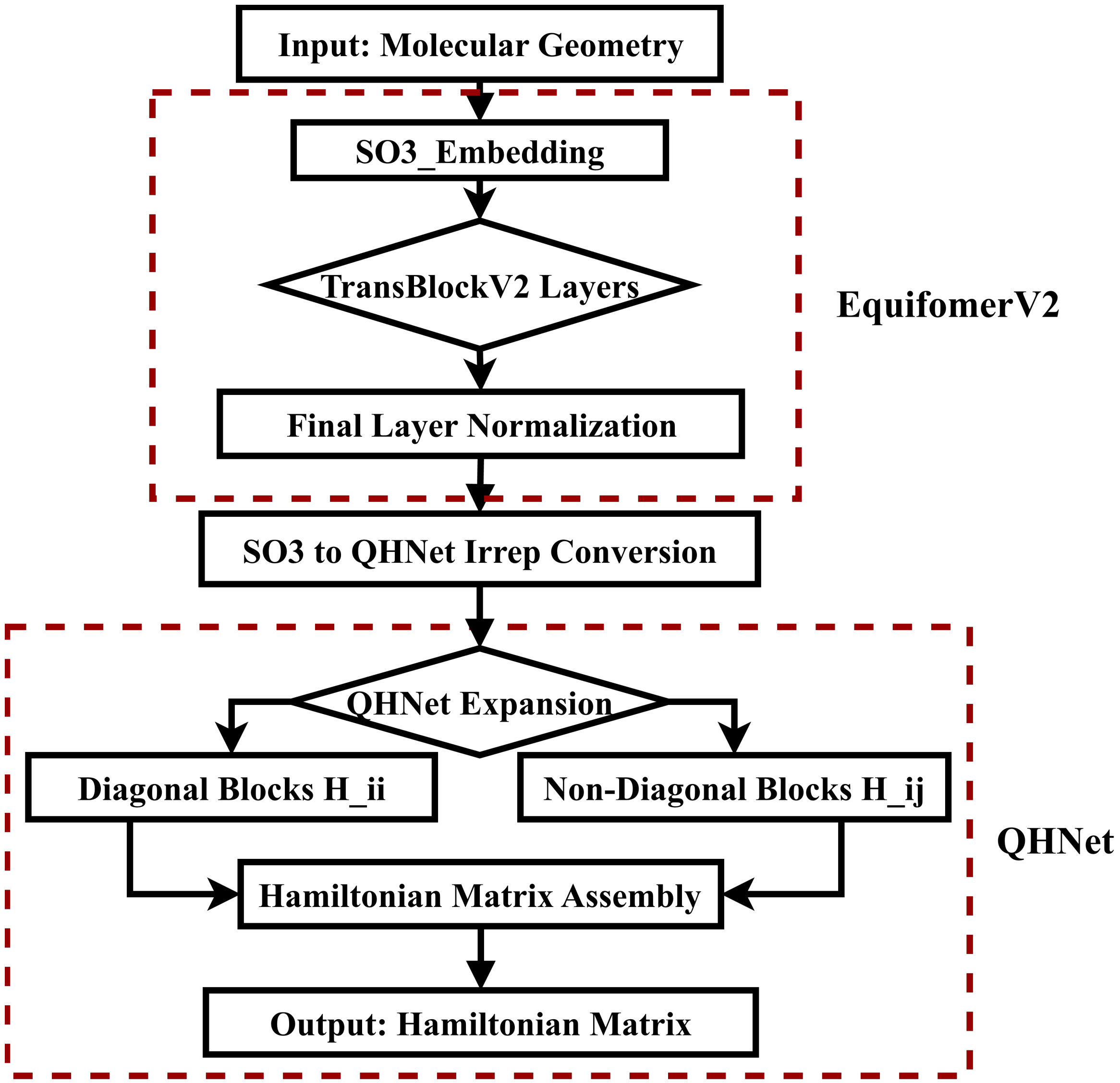
■ Equivariant Graph Neural Network Architecture

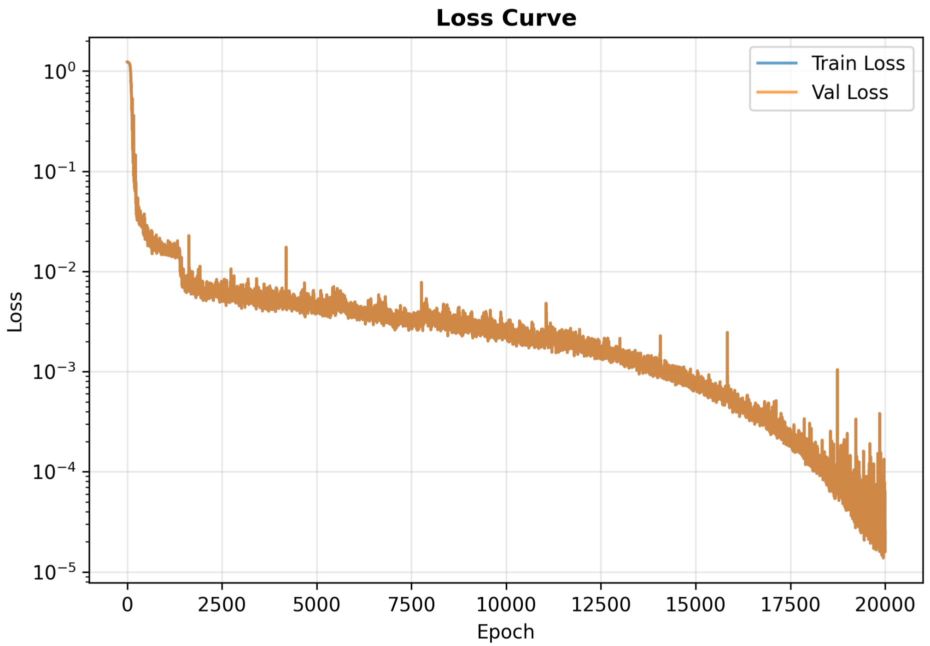


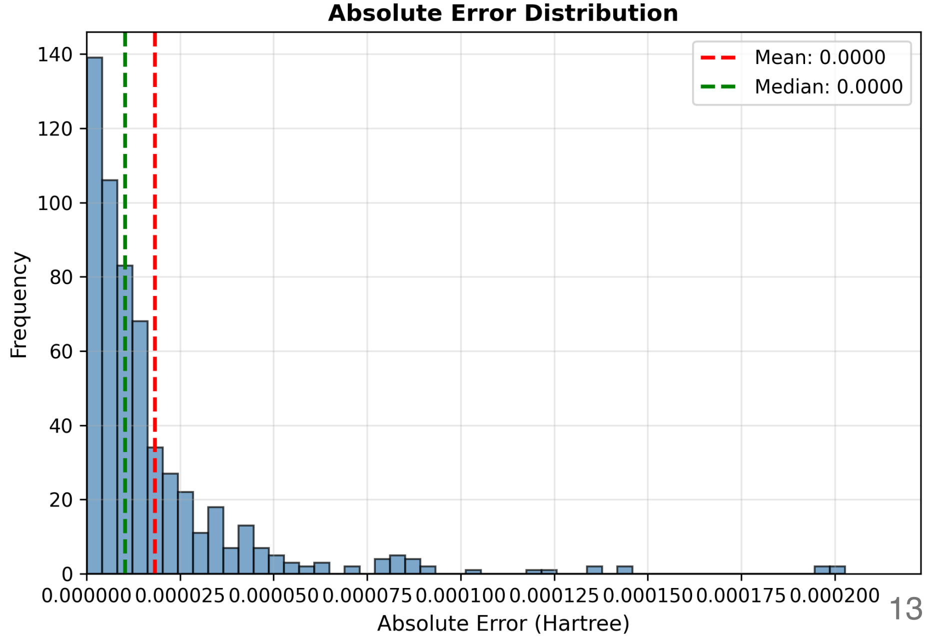
Equivariance Testing



▪ Equivariant Graph Neural Network Architecture

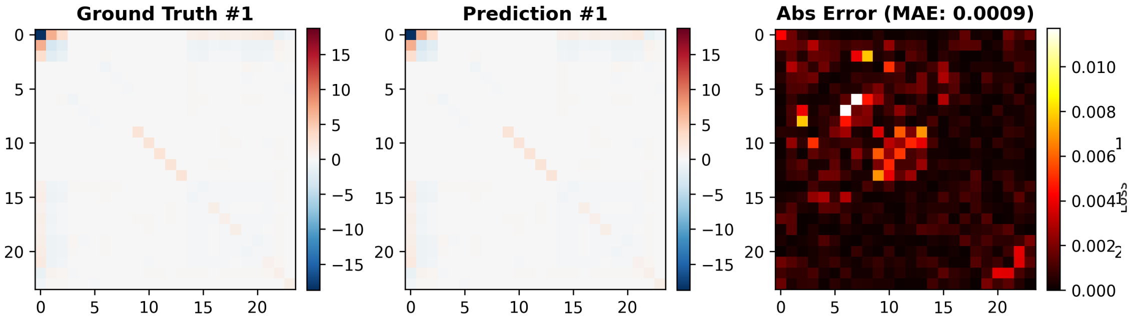




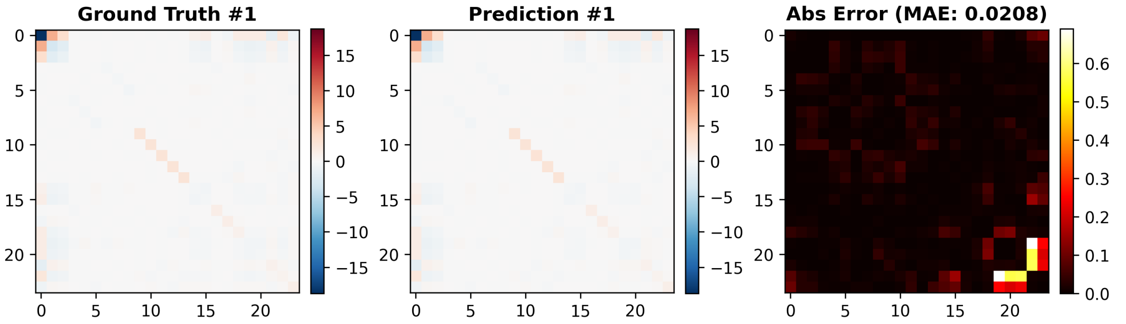


Equivariant Graph Neural Network Architecture

Training Set

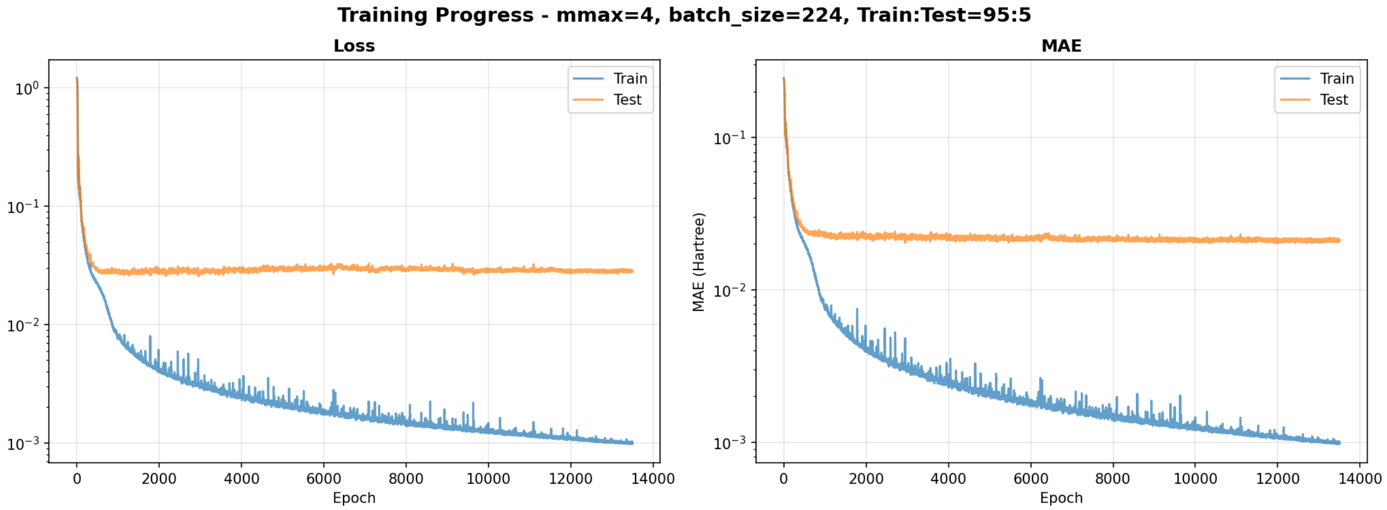


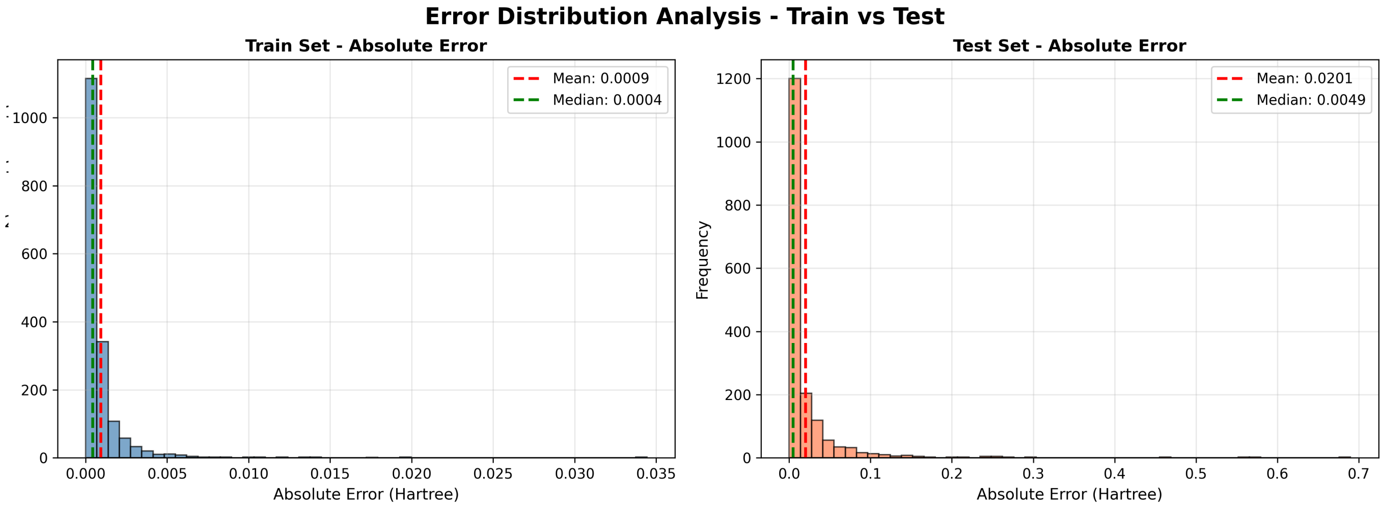
Validation Set



Loss Function

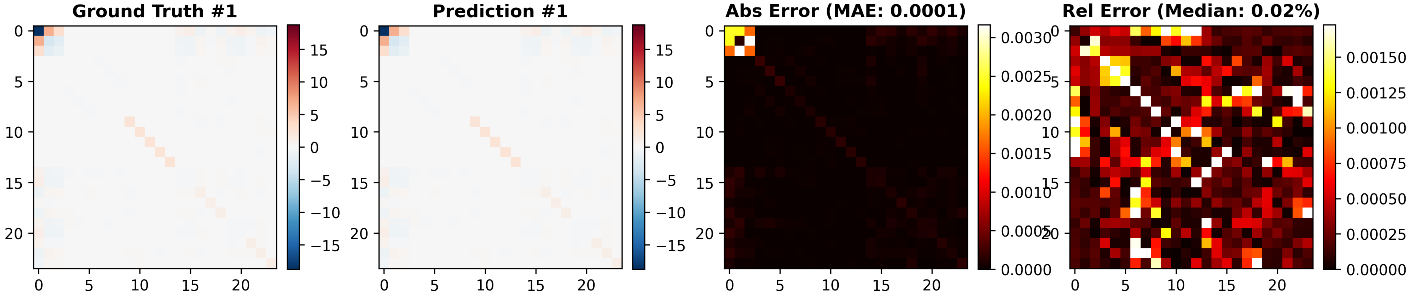
$$L = \frac{1}{N^2}\sum_i \left(H_{ij}^{true} - H_{ij}^{pre}\right)^2 + \frac{1}{N}\sum_i \left(H_{ij}^{true} - H_{ij}^{pre}\right)$$



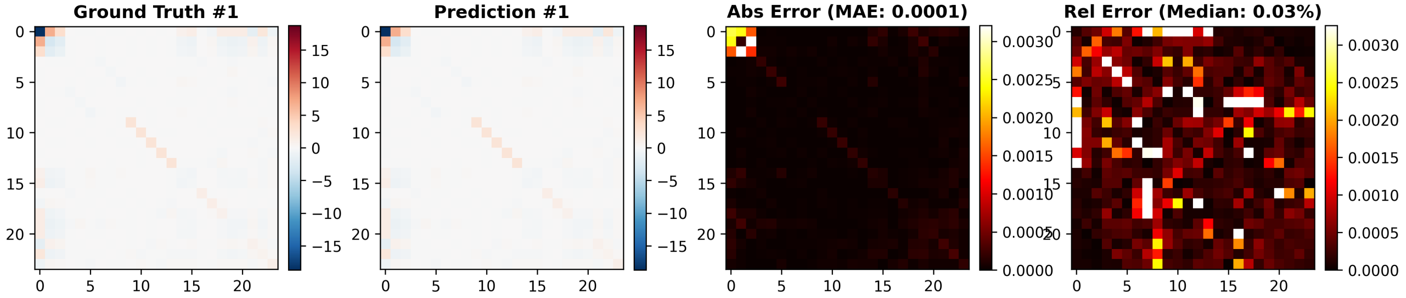


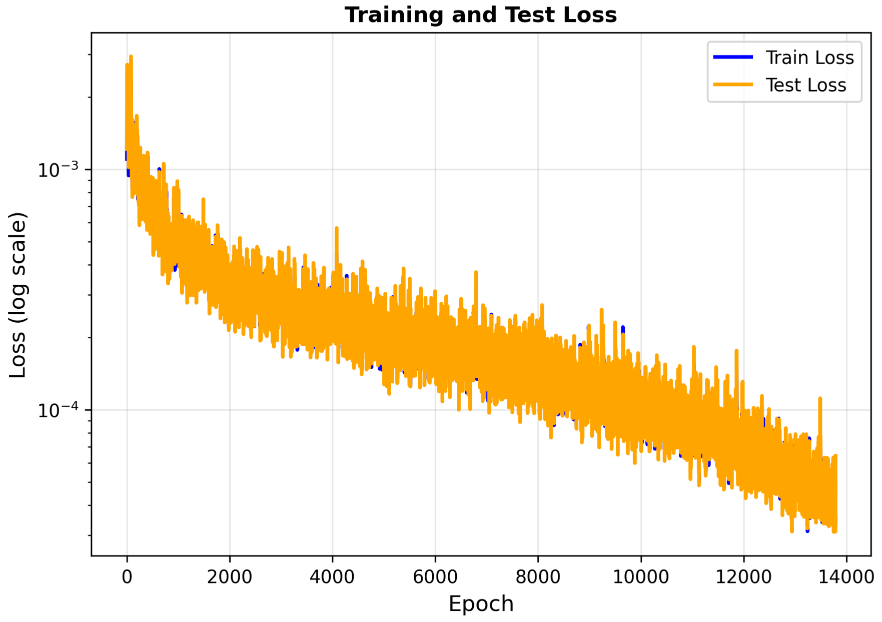
▪ Equivariant Graph Neural Network Architecture – QHNet

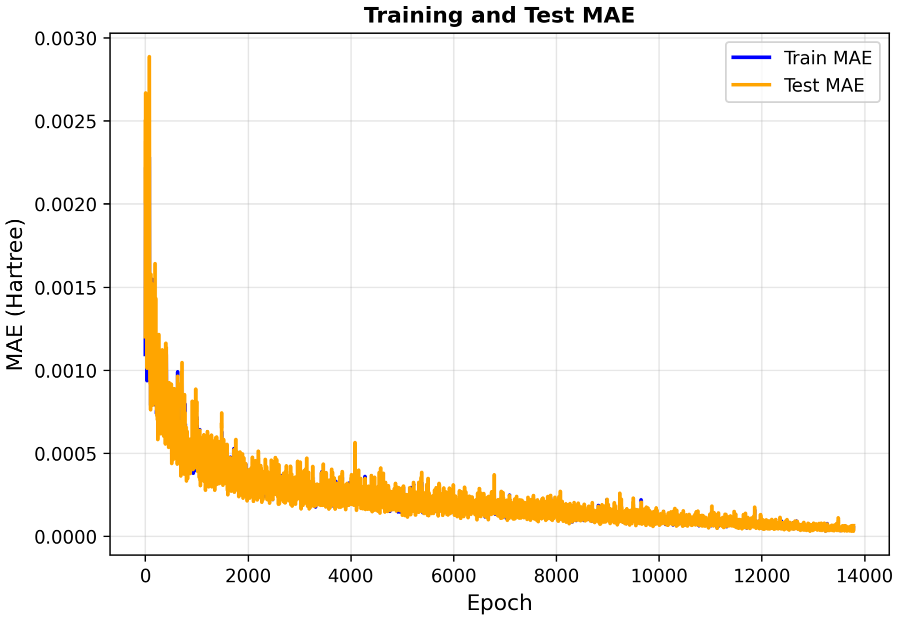
Training Set



Validation Set







GNN For Excited State – Locality and Non-Locality in Electronic Structure

$$v_{ext}\big(\boldsymbol{x},\{\boldsymbol{R}\}\big) \Leftrightarrow \rho(\boldsymbol{x}) \Leftrightarrow \{\psi_i(\boldsymbol{x})\}$$

$$\left[-\frac{1}{2}\nabla^2 + v_{ext}\big(\boldsymbol{x},\{\boldsymbol{R}\}\big) + v_H(\boldsymbol{x}) + v_{xc}(\boldsymbol{x})\right]\psi_i(\boldsymbol{x}) = \varepsilon_i\psi_i(\boldsymbol{x})$$

Runge-Gross Theorem

$$v_{ext}\big(\boldsymbol{x},\{\boldsymbol{R}\},\boldsymbol{t}\big) + \boldsymbol{C}(\boldsymbol{t}) \Leftrightarrow \rho(\boldsymbol{x},\boldsymbol{t}) \Leftrightarrow \{\psi_i(\boldsymbol{x},\boldsymbol{t})\boldsymbol{e}^{-\boldsymbol{i}\alpha(\boldsymbol{t})}\}$$

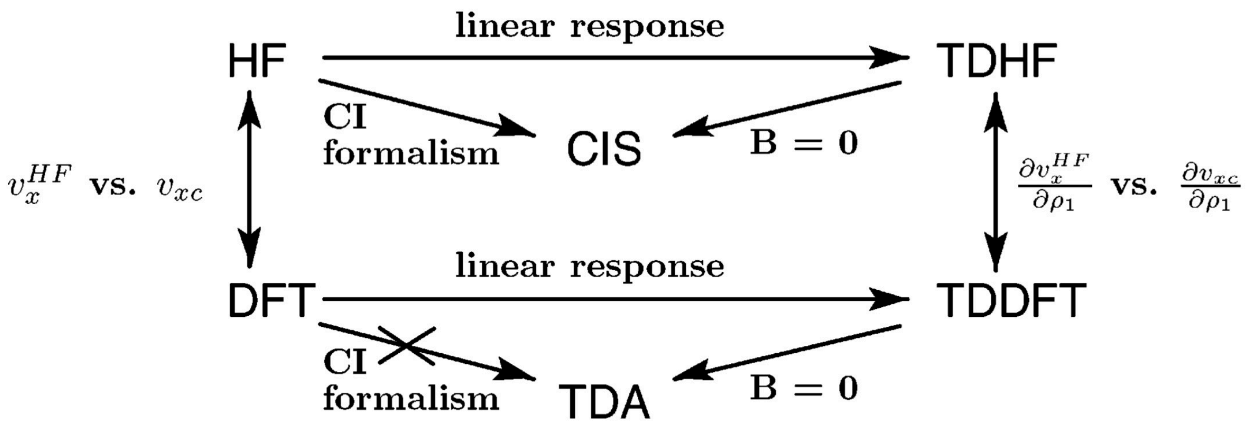
$$\left[-\frac{1}{2}\nabla^2 + v_{ext}\big(\boldsymbol{x},\{\boldsymbol{R}\},\boldsymbol{t}\big) + v_H(\boldsymbol{x},\boldsymbol{t}) + v_{xc}(\boldsymbol{x},\boldsymbol{t})\right]\psi_i(\boldsymbol{x},\boldsymbol{t}) = i\frac{\partial}{\partial t}\psi_i(\boldsymbol{x},\boldsymbol{t})$$

Linear-Response

$$F_{aa}^{(0)}x_{ai} - x_{ai}F_{ii}^{(0)} + \sum_{bj}\left(\frac{\partial F_{ai}}{\partial P_{bj}}x_{bj} + \frac{\partial F_{ai}}{\partial P_{jb}}y_{bj}\right)P_{ii}^{(0)} = \omega x_{ia}$$

$$F_{aa}^{(0)}y_{ai} - y_{ai}F_{ii}^{(0)} - \sum_{bj}P_{ii}^{(0)}\left(\frac{\partial F_{ia}}{\partial P_{bj}}x_{bj} + \frac{\partial F_{ia}}{\partial P_{jb}}y_{bj}\right) = \omega x_{ia}$$

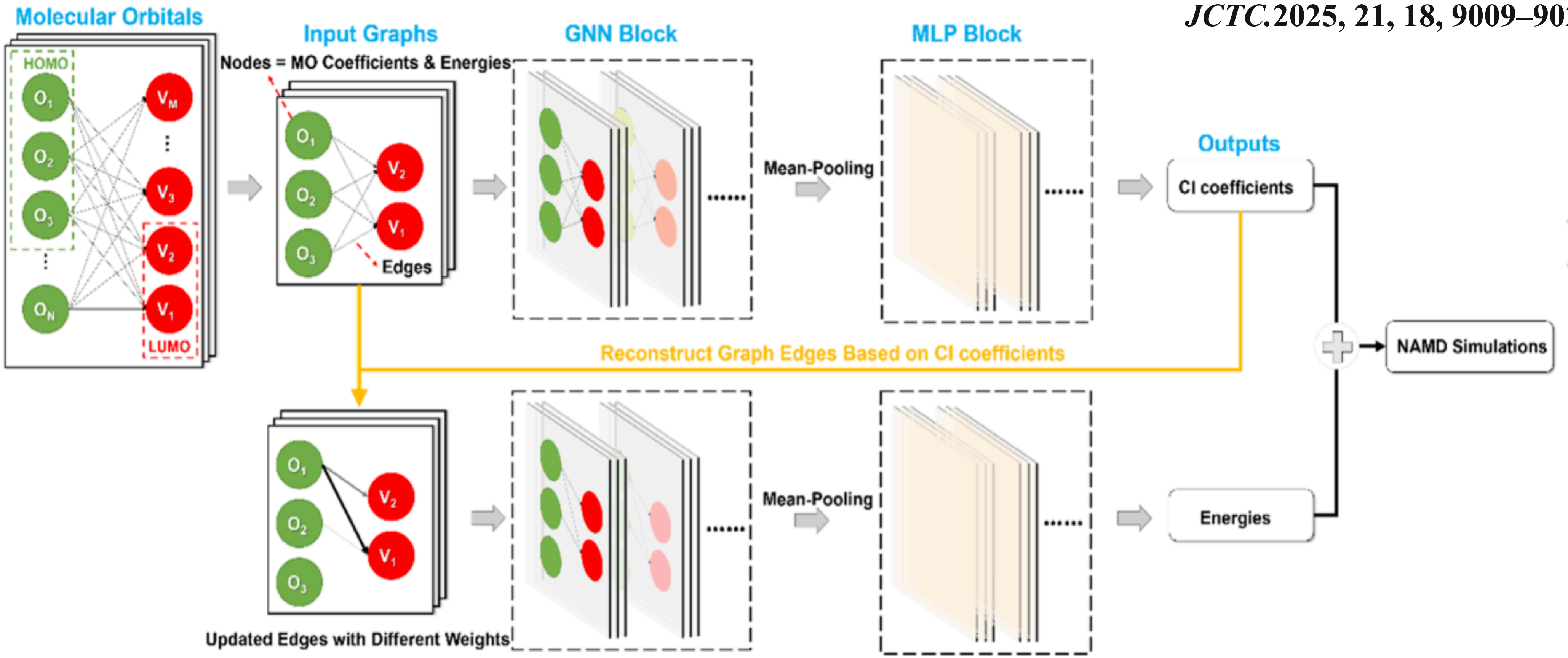
$$A = \delta_{ij}\delta_{ab}\big(\varepsilon_a - \varepsilon_i\big) + \int d\boldsymbol{r}\int \boldsymbol{r}'\phi_i(\boldsymbol{r})\phi_a^*(r)\left\{\frac{1}{\left|\boldsymbol{r}-\boldsymbol{r}'\right|} + f_{xc}\right\}\phi_b^*(r')\phi_j(\boldsymbol{r}')$$



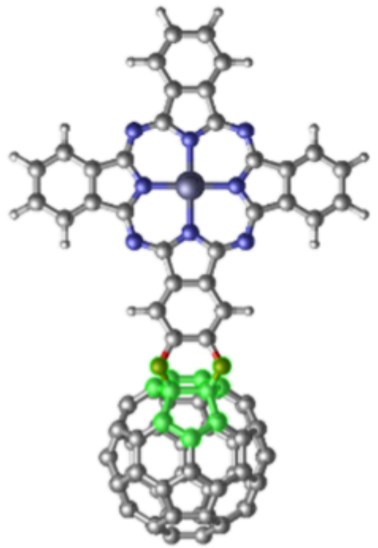
$$\begin{pmatrix} A & B \\ B^* & A^* \end{pmatrix}\begin{pmatrix} X \\ Y \end{pmatrix} = \omega\begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}\begin{pmatrix} X \\ Y \end{pmatrix}$$

$$\psi_{ex} = \sum_{ai} x_{ai}\phi_{ai} + y_{ai}\phi_{ai}$$

▪ GNN For Excited State



*JCTC.*2025, 21, 18, 9009–9022



ZnPc_56-C60

$$\begin{pmatrix} A & B \\ B^* & A^* \end{pmatrix}\begin{pmatrix} X \\ Y \end{pmatrix} = \omega \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}\begin{pmatrix} X \\ Y \end{pmatrix}$$

$$\psi_{ex} = \sum_{ai} x_{ai}\phi_{ai} + y_{ai}\phi_{ai}$$

TDA →

$$AX = \omega X$$

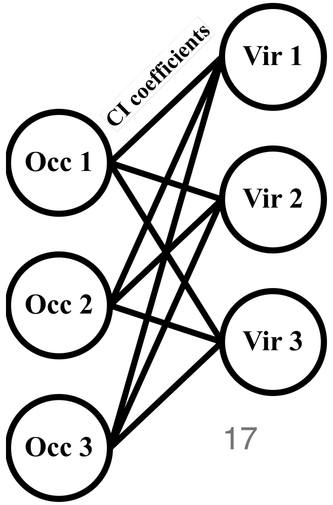
$$\psi_{ex} = \sum_{ai} x_{ai}\phi_{ai}$$

Subspace →

$$[30 \times occ, 30 \times vir]$$

$$\psi_{ex} = \sum_{ai} x_{ai}\phi_{ai}$$

GNN →



■ Summery and Next

Ground States:

- **From MD17 to Water Cluster**
- **Dataset of TADF System**
- **Overfitting !**
- **Equivariant Graph Neural Network for Neural Networ XC Functional**

Excited States:

- **A Matrix Prediction**
- **XC Functional for Excited State**