Self-Consistency Training for Density-Functional-Theory Hamiltonian Prediction

He Zhang [1 2 3] **Chang Liu** [2] **Zun Wang** [2] **Xinran Wei** [2] **Siyuan Liu** [2 3] **Nanning Zheng** [1] **Bin Shao** [2] **Tie-Yan Liu** [2]

Abstract

Predicting the mean-field Hamiltonian matrix in density functional theory is a fundamental formulation to leverage machine learning for solving molecular science problems. Yet, its applicability is limited by insufficient labeled data for training. In this work, we highlight that Hamiltonian prediction possesses a self-consistency principle, based on which we propose self-consistency training, an exact training method that does not require labeled data. It distinguishes the task from predicting other molecular properties by the following benefits: **(1)** it enables the model to be trained on a large amount of unlabeled data, hence addresses the data scarcity challenge and enhances generalization; **(2)** it is more efficient than running DFT to generate labels for supervised training, since it amortizes DFT calculation over a set of queries. We empirically demonstrate the better generalization in data-scarce and out-of-distribution scenarios, and the better efficiency over DFT labeling. These benefits push forward the applicability of Hamiltonian prediction to an ever-larger scale.

1. Introduction

Calculating properties of molecules is the foundation for a wide range of industry needs including drug design, protein engineering, and material discovery. The key to these properties is the electronic structure in the molecule, for which various computational methods are proposed. Density functional theory (DFT) (Hohenberg & Kohn, 1964; Kohn & Sham, 1965; Perdew et al., 1996; Teale et al., 2022) is perhaps the most prevailing choice due to its balanced accuracy and efficiency, but still hard to meet the demand in industry. Encouraged by the impressive advancement in machine learning, researchers have trained machine learning models on datasets with property labels to directly predict properties of queried molecules (Ramakrishnan et al., 2014; Chmiela et al., 2019; Chanussot et al., 2021). For each property, a separate model (at least a separate output module) needs to be trained. A more fundamental formulation is to predict the Hamiltonian matrix (Schütt et al., 2019), or more precisely, the effective one-electron mean-field Hamiltonian matrix, *i.e.*, the Fock matrix in a DFT calculation after convergence. The Hamiltonian matrix can directly provide all the properties that a DFT calculation can (Fig. 1), waiving the need to specify the target property or train multiple models. Moreover, Hamiltonian prediction can also accelerate running DFT by providing an accurate initialization.

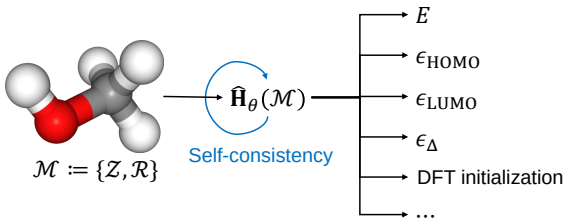


Figure 1. Hamiltonian prediction is the task to use a machine learning model to predict the mean-field Hamiltonian matrix $\hat{\mathbf{H}}_\theta(\mathcal{M})$ in density functional theory from a given molecular structure $\mathcal{M} := \{\mathcal{Z}, \mathcal{R}\}$ specified by the atomic types $\mathcal{Z}$ and coordinates $\mathcal{R}$ of atoms. It can derive various molecular properties, *e.g.*, the total energy E, the HOMO and LUMO energies $\epsilon_{\text{HOMO}}, \epsilon_{\text{LUMO}}$ and their gap ϵ_Δ for the given molecule, and can also serve as an accurate DFT initialization. We highlight in this work that the task has a self-consistency principle (the blue loop arrow), which allows training the model without labeled data.

Noticeable progress has been made for Hamiltonian prediction. Hegde & Bowen (2017) pioneered the direction using kernel ridge regression to predict semi-empirical Hamiltonian for one-dimensional systems. Schütt et al. (2019) then proposed a neural network model called SchNorb to predict Hamiltonian for small molecules, which is further enhanced for prediction efficiency by Gastegger et al. (2020). Shmilovich et al. (2022) proposed to employ atomic orbital features for Hamiltonian prediction. Noting that the Hamil-

[1]National Key Laboratory of Human-Machine Hybrid Augmented Intelligence, National Engineering Research Center for Visual Information and Applications, and Institute of Artificial Intelligence and Robotics, Xi'an Jiaotong University [2]Microsoft Research AI for Science [3]These authors did this work during an internship at Microsoft Research AI for Science. Correspondence to: Chang Liu <changliu@microsoft.com>, Nanning Zheng <nnzheng@mail.xjtu.edu.cn>.

Proceedings of the 41st International Conference on Machine Learning, Vienna, Austria. PMLR 235, 2024.

tonian matrix is composed of tensors in various orders which are equivariant to coordinate rotation in respective ways, subsequent works proposed neural network model architectures that guarantee the equivariance. Some works (Unke et al., 2021; Yu et al., 2023b; Gong et al., 2023; Yin et al., 2024) include high-order tensorial features into model input, which are processed in an equivariant way typically with tensor products. Li et al. (2022) used local frames to anchor coordinate systems with the molecule so that the prediction target is invariant. Zhang et al. (2022); Nigam et al. (2022) implemented the prediction by constructing equivariant kernels. There are works that exploited data other than the Hamiltonian directly, *e.g.*, using orbital energies (Wang et al., 2021b; Gu et al., 2022; Zhong et al., 2023) to supervise the prediction of Hamiltonian. While these prior efforts have introduced powerful architectures showing encouraging outcomes, they all rely on datasets providing Hamiltonian or orbital energy labels. Since such datasets are scarce, the applicability of Hamiltonian prediction is restricted to molecules with no more than 31 atoms (Yu et al., 2023a).[1]

In this work, we highlight a uniqueness of Hamiltonian prediction: it has a self-consistency principle (indicated by the blue loop arrow in Fig. 1), by leveraging which we design a training method that guides the model *without labeled data*. The self-consistency originates from the basic equation of DFT (Eq. (1)) that the Hamiltonian needs to satisfy. Conventional DFT solves the equation using a fixed-point iteration process called self-consistent field (SCF) iteration. In contrast, the proposed self-consistency training solves the equation by directly minimizing the residue of the equation incurred by the model-predicted Hamiltonian (Fig. 2). As the equation fully determines the prediction target, no Hamiltonian label is required, and the loss function is minimized only if the equation is satisfied and the prediction is exact. Self-consistency training compensates data scarcity with physical laws, and differentiates Hamiltonian prediction from other machine learning formulations (*e.g.*, energy prediction), in that it enables continued self-improvement without additional labeled data.

We exploit the merit of self-consistency training in two specific points. **(1)** Self-consistency training leverages unlabeled data, which allows substantial improvement of the *generalizability* of the Hamiltonian prediction model. We demonstrate that the predicted Hamiltonian as well as derived molecular properties are indeed improved by a significant margin, when labeled data is limited (data-scarce scenario) and when the model is evaluated on molecules larger than those used in training (out-of-distribution scenario).

(2) Self-consistency training on unlabeled data is more efficient than generating labels using DFT on those data for supervised learning, as we find that self-consistency training can be seen as an *amortization* of DFT calculation over a set of molecules. DFT requires multiple SCF iterations on each molecule before providing supervision, while self-consistency training only requires effectively one SCF iteration to return a training signal, hence can provide information on more molecules given the same amount of computation. The better efficiency for Hamiltonian prediction training is empirically verified in both data-scarce and out-of-distribution scenarios. More attractively, regarding physical quantities derived from the predicted Hamiltonian, self-consistency training even outperforms supervised training using full additional labels given sufficient computational budget, indicating that it is more relevant to molecular properties and real applications. We also verified the direct acceleration by self-consistency training to solve a bunch of molecules upon the conventional DFT calculation.

Finally, we demonstrate that with the above two unique benefits of self-consistency training, the applicability of Hamiltonian prediction can overcome the data limit, and is extended to molecules much larger (56 atoms) than previously reported, showing increased practical relevance. It also derives orders better molecular property results on these large molecules than end-to-end property prediction models, which are always limited by the availability of labeled data.

2. Self-Consistency Training

2.1. Preliminaries

We first provide a schematic description of the calculation mechanism of DFT and conventional supervised learning for Hamiltonian prediction before delving into self-consistency training. Appendix A provides more details.

For a given molecular structure $\mathcal{M} := \{\mathcal{Z}, \mathcal{R}\}$, where $\mathcal{Z} := \{Z^{(a)}\}_{a=1}^{A}$ and $\mathcal{R} := \{\mathbf{R}^{(a)}\}_{a=1}^{A}$ specify the atomic numbers (types) and coordinates of the A nuclei in the molecule, DFT solves the ground state of the N electrons in the molecule by minimizing electronic energy under a reduced representation of electronic state, which is N one-electron wavefunctions $\{\phi_i(\mathbf{r})\}_{i=1}^{N}$, called orbitals. Here, $\mathbf{r} \in \mathbb{R}^3$ represents the Cartesian coordinates of an electron. For practical calculation, a basis set of functions on $\mathbb{R}^3$ is used to expand the orbitals. To roughly align with the electronic structure, the basis functions depend on the molecular structure, hence are denoted as $\{\eta_{\mathcal{M},\alpha}(\mathbf{r})\}_{\alpha=1}^{B}$. Expansion coefficients of the orbitals are collected into a matrix $\mathbf{C} \in \mathbb{R}^{B\times N}$ in the following way: $\phi_i(\mathbf{r}) = \sum_{\alpha=1}^{B} \mathbf{C}_{\alpha i}\eta_{\mathcal{M},\alpha}(\mathbf{r})$.

[1] There are a few works (Li et al., 2022; Gong et al., 2023) that have demonstrated applicability to large-scale material systems. We note that this is achieved by leveraging the periodicity and locality in material systems, which do not hold perfectly in molecular systems.

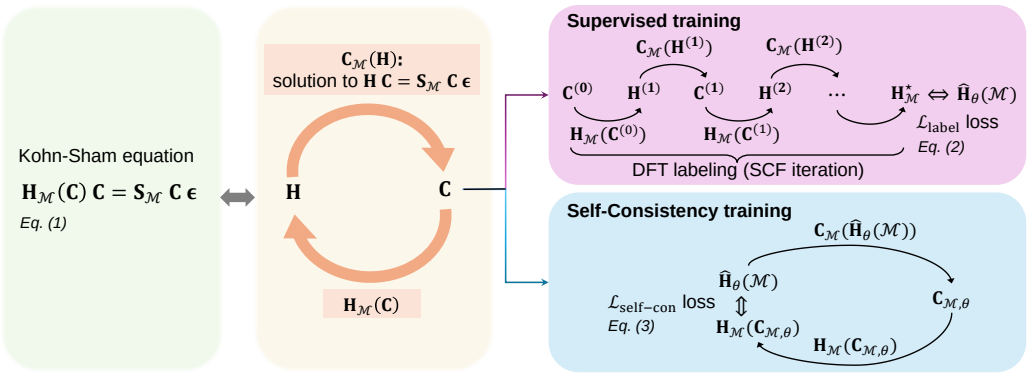


Figure 2. Illustration of the proposed self-consistency training with comparison to the conventional DFT calculation and supervised training. (**Left**) The central task of a DFT calculation is to solve the Kohn-Sham equation (Eq. (1)) for the given molecular structure $\mathcal{M}$. (**Middle**) The equation is equivalent to the condition that the eigenvectors $\mathbf{C}$ of $\mathbf{H}$ recover $\mathbf{H}$ via a known function $\mathbf{H}_\mathcal{M}(\mathbf{C})$. (**Top-Right**) To solve the equation, conventional DFT uses a fixed-point iteration (SCF iteration), which, upon convergence, gives the label $\mathbf{H}^\star_\mathcal{M}$ for supervised training (Eq. (2)) of a Hamiltonian prediction model $\hat{\mathbf{H}}_\theta(\mathcal{M})$. (**Bottom-Right**) In contrast, self-consistency training (Eq. (3)) directly minimizes the mismatch between the predicted Hamiltonian $\hat{\mathbf{H}}_\theta(\mathcal{M})$ and the matrix $\mathbf{H}_\mathcal{M}(\mathbf{C}_{\mathcal{M},\theta})$ reconstructed from its eigenvectors.

DFT typically solves the electronic energy minimization problem w.r.t $\mathbf{C}$ by directly solving the optimality equation:

$$\mathbf{H}_\mathcal{M}(\mathbf{C})\,\mathbf{C} = \mathbf{S}_\mathcal{M}\,\mathbf{C}\,\boldsymbol{\epsilon}, \tag{1}$$

which is called the Kohn-Sham equation. Here, $\mathbf{H}_\mathcal{M}(\mathbf{C})$ is a matrix-valued function with an explicit expression (given an exchange-correlation functional) (Appendix A.5). This matrix is called the Hamiltonian matrix (also noted as the Fock matrix) due to the resemblance of the equation to the Schrödinger equation. The matrix $\mathbf{S}_{\mathcal{M},\alpha\beta} := \int \eta_{\mathcal{M},\alpha}(\mathbf{r})\eta_{\mathcal{M},\beta}(\mathbf{r})\,\mathrm{d}\mathbf{r}$ is the overlap matrix of the basis, which can be computed analytically for common basis choices. Eq. (1) can be seen as a generalized eigenvalue problem defined by the matrices $\mathbf{H}_\mathcal{M}(\mathbf{C})$ and $\mathbf{S}_\mathcal{M}$, where the coefficients of orbitals $\mathbf{C}$ in the equation can be understood as eigenvectors, and the diagonal matrix $\boldsymbol{\epsilon}$ comprises eigenvalues which are referred to as orbital energies.

However, the difficulty to solve Eq. (1) is that, the matrix that defines the problem and the eigenvector solution are intertwined: the eigenvectors $\mathbf{C}$ need to recover the Hamiltonian matrix that defined the eigenvalue problem through the explicit function $\mathbf{H}_\mathcal{M}(\mathbf{C})$ (Fig. 2, middle). Conventional DFT calculation solves it using a fixed-point iteration process called self-consistent field (SCF) iteration. In each step, orbital coefficients $\mathbf{C}^{(k-1)}$ are used to construct the Hamiltonian matrix $\mathbf{H}^{(k)} := \mathbf{H}_\mathcal{M}(\mathbf{C}^{(k-1)})$, which defines a generalized eigenvalue problem $\mathbf{H}^{(k)}\mathbf{C} = \mathbf{S}_\mathcal{M}\mathbf{C}\boldsymbol{\epsilon}$, whose eigenvectors, denoted as $\mathbf{C}_\mathcal{M}(\mathbf{H}^{(k)})$, are taken as the updated orbital coefficients $\mathbf{C}^{(k)}$ (Fig. 2, top right). The converged Hamiltonian $\mathbf{H}^\star_\mathcal{M}$ and its eigenvectors hence solve Eq. (1), which then derive various molecular structures.

Hamiltonian prediction aims to bypass the SCF iteration by directly predicting $\mathbf{H}^\star_\mathcal{M}$ from molecular structure $\mathcal{M}$ using a machine-learning model $\hat{\mathbf{H}}_\theta(\mathcal{M})$,[2] where θ denotes the model parameters to be learned. The conventional way to learn such a model is by supervised learning, which requires running DFT on a set of molecular structures $\mathcal{D}$ to construct a labeled dataset $\overline{\mathcal{D}}$, on which the supervised training loss function is applied:

$$\mathcal{L}_{\text{label}}(\theta;\overline{\mathcal{D}}) := \frac{1}{|\overline{\mathcal{D}}|}\sum_{(\mathcal{M},\mathbf{H}^\star_\mathcal{M})\in\overline{\mathcal{D}}}\left\|\hat{\mathbf{H}}_\theta(\mathcal{M}) - \mathbf{H}^\star_\mathcal{M}\right\|_\mathrm{F}^2, \tag{2}$$

where $|\overline{\mathcal{D}}|$ denotes the number of samples in the set $\overline{\mathcal{D}}$. The squared Frobenius norm amounts to the mean squared error (MSE) over the matrix entries. Some works (Unke et al., 2021; Yu et al., 2023b) also include a mean absolute error (MAE) loss for more efficient learning.

2.2. Self-Consistency Training

We now describe the proposed self-consistency training for Hamiltonian prediction. It can be seen as another way to solve the Kohn-Sham equation (1), which the prediction $\hat{\mathbf{H}}_\theta(\mathcal{M})$ needs to satisfy. Recall that the equation is equivalent to the condition that $\mathbf{C} := \mathbf{C}_\mathcal{M}(\mathbf{H})$, *i.e.*, the eigenvectors of the generalized eigenvalue problem defined by the Hamiltonian matrix $\mathbf{H}$, construct the same Hamiltonian matrix, *i.e.*, $\mathbf{H}_\mathcal{M}(\mathbf{C}) = \mathbf{H}$ (Fig. 2, middle). The self-consistency training loss is hence designed to enforce this condition: the difference between the predicted Hamiltonian $\hat{\mathbf{H}}_\theta(\mathcal{M})$ and the reconstructed Hamiltonian from itself should be minimized, where the

[2]The "hat" or "circumflex" accent in the notation here is meant to represent "a neural-network estimator".

reconstruction is done by first solving for the eigenvectors $\mathbf{C}_{\mathcal{M},\theta} := \mathbf{C}_{\mathcal{M}}\big(\hat{\mathbf{H}}_\theta(\mathcal{M})\big)$ of the generalized eigenvalue problem defined by $\hat{\mathbf{H}}_\theta(\mathcal{M})$ then constructing the Hamiltonian using $\mathbf{H}_{\mathcal{M}}(\mathbf{C}_{\mathcal{M},\theta})$ (Fig. 2, bottom right). Explicitly, the self-consistency loss is:

$$\mathcal{L}_{\text{self-con}}(\theta; \mathcal{D}) := \frac{1}{|\mathcal{D}|} \sum_{\mathcal{M}\in\mathcal{D}} \left\| \hat{\mathbf{H}}_\theta(\mathcal{M}) - \mathbf{H}_{\mathcal{M}}\Big(\mathbf{C}_{\mathcal{M}}\big(\hat{\mathbf{H}}_\theta(\mathcal{M})\big)\Big) \right\|_{\text{F}}^2. \quad (3)$$

Following the practice of previous work (Unke et al., 2021; Yu et al., 2023b), we also include its MAE counterpart in place of the squared Frobenius norm into the loss. The implementation process is summarized in Alg. 1. Note that the loss only requires a set of molecular structures $\mathcal{D}$ unnecessarily with Hamiltonian labels. It thus enables leveraging numerous molecular structures for learning Hamiltonian prediction, which could substantially enhance generalizability of the prediction model to a wide range of molecules, allowing applicability beyond the limitation of labeled datasets.

We make the following four remarks regarding the understanding of the self-consistency loss. **(1)** The self-consistency loss is distinct from regularization or self-supervised training, in the sense that the loss by itself can already drive the model towards the exact target, since the loss enforces the equation that determines the target. **(2)** We emphasize that the loss should *not* be interpreted as updating the prediction $\hat{\mathbf{H}}_\theta(\mathcal{M})$ towards the reconstructed Hamiltonian $\mathbf{H}_{\mathcal{M}}\big(\mathbf{C}_{\mathcal{M}}\big(\hat{\mathbf{H}}_\theta(\mathcal{M})\big)\big)$ as a fixed target (which is the case when the `stop_grad` operation is applied to the reconstructed Hamiltonian), and the back-propagation (*i.e.*, computation of the gradient of the loss w.r.t θ) through the Hamiltonian reconstruction process is indispensable. This is because the reconstructed Hamiltonian unnecessarily comes closer to the target solution (Pulay, 1982; Cances & Le Bris, 2000), so taking the reconstructed Hamiltonian as a constant when optimizing θ may even make the model worse. Instead, the loss aims to minimize the change from the reconstruction process. To minimize this change, both the predicted matrix and the reconstructed matrix are driven towards the solution. **(3)** One may also consider enforcing self-consistency by minimizing the difference in the derived energy after reconstruction, which meets the common stopping criterion in a DFT calculation and could hold more physical relevance. But this would require eigen-solving the reconstructed matrix and evaluating the energy from the eigenvectors, which is as costly as another Hamiltonian reconstruction, making the loss unacceptably costly to optimize. **(4)** The self-consistency loss bears some similarity to the SCF loss in DM21 (Kirkpatrick et al., 2021). Both connect a DFT solution and an exchange-correlation (XC) functional defining the DFT calculation (part of $\mathbf{H}_{\mathcal{M}}(\mathbf{C})$ in our formulation). The SCF loss is used to regularize an XC functional model with a label (solution), while we use the self-consistency loss to train a solution-prediction model given a well-established XC functional. It is future work to investigate the utility of the SCF loss for unsupervised Hamiltonian prediction.

Algorithm 1 Implementation of self-consistency loss (on one molecular structure)

Require: Molecular structure $\mathcal{M} = \{\mathcal{Z}, \mathcal{R}\}$ comprising types $\mathcal{Z} := \{Z^{(a)}\}_{a=1}^{A}$ and coordinates $\mathcal{R} := \{\mathbf{R}^{(a)}\}_{a=1}^{A}$ of its atoms, pre-computed integral matrices (*e.g.*, overlap matrix $\mathbf{S}_{\mathcal{M}}$), Hamiltonian prediction model $\hat{\mathbf{H}}_\theta(\cdot)$ to be learned.

1: Generate requisite integrals and quadrature grid for constructing Hamiltonian (Appendix B.2);
2: Predict Hamiltonian $\hat{\mathbf{H}}_\theta(\mathcal{M})$ using the model;
3: Solve for the eigenvectors $\mathbf{C}_{\mathcal{M},\theta}$ of the generalized eigenvalue problem $\hat{\mathbf{H}}_\theta(\mathcal{M})\,\mathbf{C} = \mathbf{S}_{\mathcal{M}}\,\mathbf{C}\,\boldsymbol{\epsilon}$.
4: Reconstruct Hamiltonian $\mathbf{H}_{\mathcal{M}}(\mathbf{C}_{\mathcal{M},\theta})$ following an explicit expression (Appendix A.5);
5: Compute the loss $\mathcal{L}_{\text{self-con}}(\theta; \{\mathcal{M}\})$ as the addition of the mean squared error (shown in Eq. (3)) and mean absolute error between $\hat{\mathbf{H}}_\theta(\mathcal{M})$ and $\mathbf{H}_{\mathcal{M}}(\mathbf{C}_{\mathcal{M},\theta})$.

output $\mathcal{L}_{\text{self-con}}(\theta; \{\mathcal{M}\})$.

2.3. Implementation Considerations

For stable and efficient optimization of the self-consistency loss, we mention a few technical treatments.

Back-Propagation through Eigensolver. As mentioned, back-propagation through the reconstruction process $\mathbf{H}_{\mathcal{M}}\big(\mathbf{C}_{\mathcal{M}}\big(\hat{\mathbf{H}}_\theta(\mathcal{M})\big)\big)$ is indispensable. This requires differentiation through the eigensolver $\mathbf{C}_{\mathcal{M}}(\mathbf{H})$. We leverage the eigensolver implemented in an automatic differentiation package PyTorch (Paszke et al., 2019) which automatically provides the differentiation calculation. Nevertheless, the calculation often appears numerically unstable (Ionescu et al., 2015; Wang et al., 2019), as it relies on a matrix $\mathbf{G}$ (see Appendix B.2 for detailed derivation),

$$\mathbf{G}_{ij} = \begin{cases} 1/(\epsilon_i - \epsilon_j), & i \neq j, \\ 0, & i = j, \end{cases}$$

where ϵ_i is i-th eigenvalue. When there are two close eigenvalues, the values in $\mathbf{G}$ can be exceedingly large, causing unstable training. To mitigate this instability, we introduce two treatments. The first is simply truncating the values in $\mathbf{G}$ if they are larger than a chosen threshold. The second treatment is to skip the model parameter update when the scale of the gradient w.r.t parameters exceeds a certain threshold. Appendix B.2 presents more details.

Efficient Hamiltonian Reconstruction. Evaluating the function $\mathbf{H}_{\mathcal{M}}(\mathbf{C})$ is also a costly procedure, mainly due

to two computational components. The first is the evaluation of basis functions on a quadrature grid for evaluating the exchange-correlation component of the Hamiltonian matrix (Appendix A.5). To accelerate this part, we implemented a GPU-based evaluation process of basis functions on grid points. The other costly procedure is the evaluation of the Hartree component of the Hamiltonian matrix, which requires $O(N^4)$ cost in its vanilla form. For efficient evaluation of this term, we adopt the density fitting approach (Appendix A.5), a widely used technique in DFT to reduce the complexity to $O(N^3)$ with acceptable loss of accuracy.

2.4. Amortization of DFT Calculation

As mentioned in Sec. 2.2, self-consistency training can be applied to unlimited unlabeled molecular structures, hence can substantially improve the generalizability of a Hamiltonian prediction model. Here, we point out that self-consistency training is also more efficient to improve generalizability than generating additional labels using DFT on those data and then supervising the model. This is based on the interpretation that self-consistency training is an *amortization* of DFT: for a given molecular structure $\mathcal{M}$, DFT requires *multiple* SCF iterations for convergence before it can provide a supervision on $\mathcal{M}$ (Fig. 2, top right), while self-consistency training only requires *one* SCF iteration to evaluate the loss and guide the training on $\mathcal{M}$ (Fig. 2, bottom right). This indicates that given the same amount of computational resources measured in the number of SCF iterations, self-consistency training can distribute the resource on more molecular structures, hence providing information on a larger range of the input space. This is more valuable than Hamiltonian labels on fewer molecular structures for the model to generalize to a broad range of molecular structures.

Self-consistency training can also be viewed as a way to carry out DFT calculation. Under this view, the amortization effect makes self-consistency training a more efficient method than the conventional DFT to solve a large amount of molecular structures. Apart from the amortization effect, the efficiency is also benefited from the generalization of a Hamiltonian prediction model to similar molecular structures, on which the model can already provide close results. The demand to solve a set of molecular structures is not uncommon; *e.g.*, high-throughput drug screening requires investigating a large amount of ligand-receptor compounds using DFT (Jordaan et al., 2020). Therefore, the applicability scope of Hamiltonian prediction with self-consistency training is enlarged.

3. Experimental Results

We now empirically validate the benefits of self-consistency training. We adopt QHNet (Yu et al., 2023b) as the Hamiltonian prediction model, which is an $\mathrm{SE}(3)$-equivariant graph neural network that balances efficiency and accuracy. Additional results based on alternative architectures (*e.g.*, PhiSNet (Unke et al., 2021)) are provided in Appendix D.2, which indicate the same conclusions as presented below.

We employ the following metrics to measure prediction accuracy. A direct metric is the mean absolute error (MAE) over matrix entries between the predicted and DFT-solved Hamiltonian matrices, as introduced by Schütt et al. (2019). Directly derived quantities from Hamiltonian, including orbital energies $\boldsymbol{\epsilon}$ and coefficients $\mathbf{C}$ solved from the generalized eigenvalue problem, are also used to assess accuracy, measured by MAE for $\boldsymbol{\epsilon}$ and cosine similarity for $\mathbf{C}$. We also report the MAE for three molecular properties relevant to molecular research, including the highest occupied molecular orbital energy ϵ_{HOMO}, the lowest unoccupied molecular orbital energy ϵ_{LUMO}, and their gap ϵ_{Δ}. We also assess the utility for accelerating DFT by the ratio of the number of SCF steps to convergence using the prediction as initialization over the number using the standard initialization, denoted as "SCF Accel." The conventional DIIS (Pulay, 1980) strategy is adopted for running SCF iteration, while we also present "SCF Accel." results using the second-order SCF (SOSCF) (Sun et al., 2017) iteration strategy in Appendix D.3, considering that DIIS may lead to non-monotone iterations thereby diminishes the benefit of a more accurate initialization.

3.1. Self-Consistency Training Improves Generalization

As discussed in Sec. 2.2, self-consistency training can leverage unlabeled data to improve generalizability. We validate this benefit on two challenging generalization scenarios.

Data-Scarce Scenario. For some scientific tasks with limited labels available, it is difficult for the machine learning model to achieve meaningful performance even for in-distribution (ID) generalization. To demonstrate the advantage of self-consistency training in this scenario, we first conduct generalization experiments over the conformational space. Conformations of ethanol, malondialdehyde and uracil from the MD17 dataset (Chmiela et al., 2019; Schütt et al., 2019) are considered. The training/validation/test split setting follows Schütt et al. (2019). To simulate a data-scarce setting, for each molecule, only 100 labeled conformations (denoted as $\overline{\mathcal{D}^{(1)}}$) are provided for supervised training using the supervised loss $\mathcal{L}_{\text{label}}(\theta; \overline{\mathcal{D}^{(1)}})$ (Eq. (2)). With the self-consistency loss (Eq. (3)), a large amount of additional unlabeled structures in the training set (about 24,900 structures for each molecule; denoted as $\mathcal{D}^{(2)}$) can be leveraged, in which case the resulting loss function is:

$$\mathcal{L}_{\text{label}}(\theta; \overline{\mathcal{D}^{(1)}}) + \lambda_{\text{self-con}}\, \mathcal{L}_{\text{self-con}}(\theta; \mathcal{D}^{(2)}). \tag{4}$$

See more training details in Appendix C.4.

Table 1. Generalization improvement by self-consistency training on unlabeled data in the *data-scarce* scenario (MD17 Hamiltonian). Evaluated on the test split of conformations of each molecule.

Molecule	Setting	$\mathbf{H}$ $[\mu E_\mathrm{h}]\downarrow$	$\boldsymbol{\epsilon}$ $[\mu E_\mathrm{h}]\downarrow$	$\mathbf{C}$ $[\%]\uparrow$	ϵ_HOMO $[\mu E_\mathrm{h}]\downarrow$	ϵ_LUMO $[\mu E_\mathrm{h}]\downarrow$	ϵ_Δ $[\mu E_\mathrm{h}]\downarrow$	SCF Accel. $[\%]\downarrow$
Ethanol	label	160.36	712.54	99.44	911.64	6800.84	6643.11	68.3
	label + self-con	**75.65**	**285.49**	**99.94**	**336.97**	**1203.60**	**1224.86**	**61.5**
Malondi-aldehyde	label	101.19	456.75	99.09	471.92	1093.22	1115.94	69.1
	label + self-con	**86.60**	**280.39**	**99.67**	**274.45**	**279.14**	**324.37**	**62.1**
Uracil	label	88.26	1079.51	95.83	1217.17	12496.1	11850.56	65.8
	label + self-con	**63.82**	**315.40**	**99.58**	**359.98**	**369.67**	**388.30**	**54.5**

Table 2. Generalization improvement by self-consistency training on unlabeled data in the *OOD* scenario (QH9). The model is trained on the QH9-small training split, and evaluated on the QH9-large test split directly (`zero-shot`) or after fine-tuned by `self-con`sistency loss on QH9-large training split (without labels).

Setting	$\mathbf{H}$ $[\mu E_\mathrm{h}]\downarrow$	$\boldsymbol{\epsilon}$ $[\mu E_\mathrm{h}]\downarrow$	$\mathbf{C}$ $[\%]\uparrow$	ϵ_HOMO $[\mu E_\mathrm{h}]\downarrow$	ϵ_LUMO $[\mu E_\mathrm{h}]\downarrow$	ϵ_Δ $[\mu E_\mathrm{h}]\downarrow$	SCF Accel. $[\%]\downarrow$
zero-shot	69.67	403.52	95.72	778.86	12230.49	12203.12	66.3
self-con (all-param)	65.74	375.31	**97.31**	565.50	**1130.55**	**1316.96**	**64.5**
self-con (adapter)	**64.48**	**268.83**	97.12	**449.80**	1220.54	1394.29	65.0

Prediction results on test structures are summarized in Table 1. Compared to the results of supervised (label-based) training, applying self-consistency loss on unlabeled structures leads to a substantial improvement across all evaluation metrics and molecules. Notably, it achieves a significant reduction in the Hamiltonian MAE, with decreases from 14.4% to 52.8%. The MAE of ϵ_HOMO, ϵ_LUMO and ϵ_Δ are even reduced by several folds. Applying self-consistency training also substantially improves the acceleration for conventional DFT. In addition, a point-by-point, instance-level comparison in Appendix D.5 shows that self-consistency training leads to faster SCF convergence over various molecular systems consistently, while supervised training does not. These findings underscore the attractive capability of self-consistency training in breaking the limitation of data scarcity.

Out-of-Distribution (OOD) Scenario. Yu et al. (2023a) introduced the QH9 dataset to benchmark Hamiltonian prediction over the chemical space. Their findings highlight a challenging out-of-distribution (OOD) generalization scenario: models trained on smaller molecules often struggle to generalize to larger molecules, restricting the applicability. To demonstrate the effect of better generalization using self-consistency training, we construct a similar OOD scenario. We split the molecular structures in QH9 into two subsets: QH9-small comprising molecules with no more than 20 atoms, and QH9-large with larger molecules. The two subsets are then correspondingly divided at random into distinct training/validation and training/validation/test splits (see more dataset details in Appendix C.1). The model is trained and validated on QH9-small using the supervised loss (Eq. (2)), and is tested on the QH9-large test split (dubbed `zero-shot`). With the self-consistency loss (Eq. (3)), the model is allowed to be fine-tuned (without the pretraining supervised loss) on relevant but unlabeled molecular structures, for which we take the QH9-large training split. We consider two fine-tuning settings: fine-tuning all parameters of the model, dubbed `self-con (all-param)`, or introducing an adapter module atop the model which is the only optimized component, dubbed `self-con (adapter)`. We ensure all models are sufficiently trained. Appendix C.4 shows more training details.

From the results shown in Table 2, we observe a significant improvement by fine-tuning using self-consistency on unlabeled QH9-large molecules in both fine-tuning settings. Remarkably, self-consistency reduces the MAE of ϵ_LUMO and ϵ_Δ by an order of magnitude. This result demonstrates that self-consistency training enables the flexibility to adapt a model to an OOD workload without labeled data.

3.2. Self-Consistency Training is More Efficient than DFT Labeling

As discussed in Sec. 2.4, self-consistency training can train a model more efficiently than DFT labeling followed by supervised learning, due to its amortization effect of DFT calculation. We demonstrate the empirical efficiency by comparing self-consistency training/fine-tuning in the above two scenarios to the alternative approach of generating labels by running DFT on the additional unlabeled molecular structures then applying supervised training using these extended labels (dubbed `extended-label`). We also consider a variant that conducts DFT labeling along with model training (dubbed `extended-label-online`): DFT is only run on unlabeled molecular structures in the current training batch drawn at random, and the generated labels are stored for possible use in future batches. This could be more efficient than `extended-label`.

The efficiency is monitored by the accuracy-cost curve along training. The accuracy is measured by the validation Hamil-

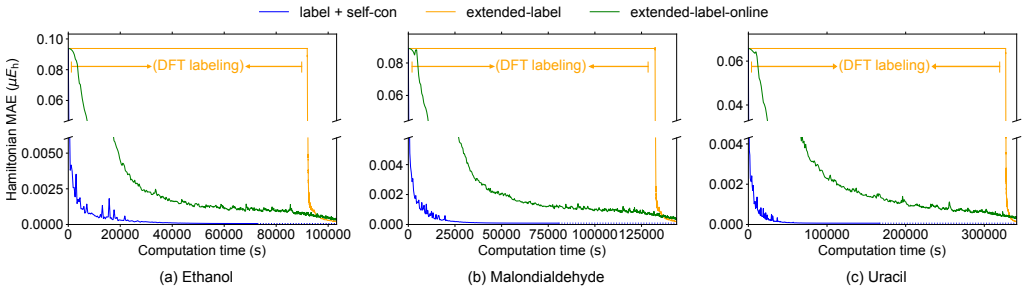


Figure 3. Efficiency comparison in the *data-scarce* scenario (MD17 Hamiltonian) among `self-con`sistency training on unlabeled data, supervised training following DFT labeling on unlabeled data (`extended-label`), and supervised training along with DFT labeling (`extended-label-online`). Dotted horizontal lines extend from the last measured point of the respective curves.

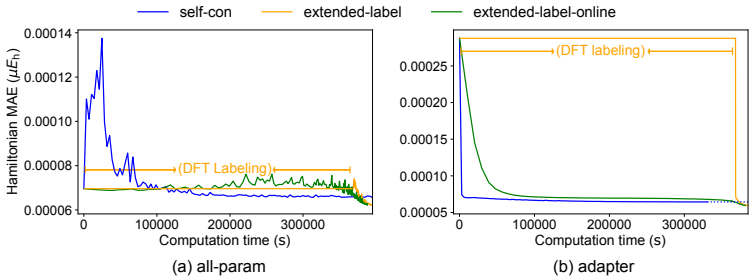


Figure 4. Efficiency comparison in the *OOD* scenario (QH9) among fine-tuning using `self-con`sistency training on unlabeled data, supervised training following DFT labeling on unlabeled data (`extended-label`), and supervised training along with DFT labeling (`extended-label-online`). Dotted horizontal lines extend from the last measured point of the respective curves.

tonian MAE. The cost can be measured by *the number of SCF steps* along self-consistency training or DFT labeling. This matches the analysis in Sec. 2.4 and is system- and implementation-independent. Results are shown in Figs. D.1 and D.2 in Appendix D.4, which validates the better efficiency in all cases.

For better practical relevance and considering the complication of the interplay between running SCF and model parameter optimization, we present results measured by the cost of real *computation time* here. All methods are implemented on a workstation equipped with an NVIDIA A100 GPU with 80 GiB memory and a 24-core AMD EPYC CPU.

Data-Scarce Scenario. Accuracy-cost curves of the three training strategies are presented in Fig. 3. We see that self-consistency training converges rapidly, achieving a low prediction error with a cheap cost. In contrast, the `extended-label` strategy keeps a plateau at first, representing the process to generate labels using DFT during which the model is not optimized. It is only after the DFT labeling process that the prediction error starts to drop. The `extended-label-online` strategy indeed improves upon `extended-label` by amortizing the labeling cost over the course of training, but it is still not as efficient as self-consistency training, whose amortization capability allows a more frequent model optimization per SCF step. We note that due to our hardware limitation, DFT labeling and model optimization are performed sequentially. The two processes can be parallelized which may further improve the efficiency of `extended-label-online` at the cost of using more machines.

Out-of-Distribution (OOD) Scenario. All the three training settings are run for the fine-tuning stage of the model. Curves on QH9-large validation split are presented in Fig. 4. We see again that self-consistency training achieves a high accuracy at a relatively low cost across both fine-tuning settings. In contrast, `extended-label` and `extended-label-online` require a higher computational cost to reach a comparable level of accuracy. These results indicate the better efficiency of the self-consistency training through the amortization effect.

Performance of Final Results. At the end of the accuracy-cost curves in Fig. 4, computational resource is sufficient to generate full extended labels for supervised training in the OOD scenario, which has the most abundant and direct supervision information, hence serves as an upper bound of Hamiltonian prediction performance. But as shown in Table 3, this only applies to the Hamiltonian MAE, corresponding to the directly supervised quantity, while self-consistency training still excels at derived physical quantities, especially on ϵ_{HOMO}, ϵ_{LUMO} and ϵ_{Δ}, which are directly concerned molecular properties thus more relevant to practical applications. Appendix D.1 shows a similar obser-

Table 3. Performance comparison between self-consistency training, and supervised training using *full extended labels*, in the *OOD* scenario, corresponding to the ending points of Fig. 4 (`extended-label-online` is close to `extended-label`).

FT mode	Setting	$\mathbf{H}$ [μE_h] ↓	$\boldsymbol{\epsilon}$ [μE_h] ↓	$\mathbf{C}$ [%] ↑	ϵ_HOMO [μE_h] ↓	ϵ_LUMO [μE_h] ↓	ϵ_Δ [μE_h] ↓	SCF Accel. [%] ↓
all-param	extended-label	**62.13**	**365.66**	96.89	577.46	5962.16	6137.66	65.0
	self-con	65.74	375.31	**97.31**	**565.50**	**1130.55**	**1316.96**	**64.5**
adapter	extended-label	**59.67**	330.05	96.63	541.92	6372.12	6445.33	65.2
	self-con	64.48	**268.83**	**97.12**	**449.80**	**1220.54**	**1394.29**	**65.0**

Table 4. Efficiency comparison between self-consistency training and *conventional DFT* for solving MD17 molecular structures. Computation times under the same stopping criteria are shown for solving the unlabeled molecular structures in the data-scarce scenario.

Molecule	criterion [μE_h]	$t_\text{self-con}$ [s]	t_DFT [s]
Ethanol	31.0	$\mathbf{4.50\times10^4}$	6.40×10^4
Malondialdehyde	88.9	$\mathbf{4.81\times10^4}$	1.05×10^5
Uracil	177.2	$\mathbf{1.23\times10^5}$	2.15×10^5

vation in the data-scarce scenario. This indicates that even with unlimited computational resource, self-consistency training could still be preferred than generating labels to better support real applications.

Direct Acceleration over DFT Calculation. As mentioned at the end of Sec. 2.4, self-consistency training is also a way to directly accelerate DFT calculation on a large amount of molecular structures by leveraging its amortization effect. To demonstrate the advantage empirically, we compare the computation time of self-consistency training (using Eq. (4)) $t_\text{self-con}$ and of DFT calculation t_DFT for solving the unlabeled molecular structures in the data-scarce scenario (see Sec. 3.1). The same stopping criterion is applied to both methods in each case, which is taken as the error of electronic energy (Eq. (A.24)) derived from the Hamiltonian following the convention in DFT calculation.

Results are shown in Table 4. We see that self-consistency training indeed requires lower computational cost than DFT to reach the same level of accuracy, demonstrating the practical benefit of amortization. Appendix D.4 shows more implementation details.

3.3. Self-Consistency Training Extends the Scale of Hamiltonian Prediction

After verifying the advantages of self-consistency training, we now wield this powerful tool to extend Hamiltonian prediction to molecules larger than previously reported in the field, hence enhancing the relevance to real applications.

Extension to a Larger Scale. For molecules larger than those covered by Hamiltonian prediction previously, we consider two molecules in the MD22 dataset (Chmiela et al., 2023): Ac-Ala3-NHMe (ALA3, 42 atoms) and DHA (56 atoms). Both molecules exceed the size of the largest molecule in the QH9 dataset (31 atoms) with a significant gap. Due to the extensive DFT cost, we generate labels for each molecule only on 500 randomly selected structures from MD22, which are used only for evaluation. The model is pre-trained on nearly all QH9 molecules (the QH9-full setting in Table C.2), then got `all-param` fine-tuned using the self-consistency loss (Eq. (3)) on the selected structures but without using the labels. For ease of training, we employ the MINAO initialization (Sun et al., 2018) as a base Hamiltonian and let the model predict the residual correction.

The results are presented in Table 5. Compared to the previously best setting `zero-shot` which can only be directly used right after pre-trained on QH9, fine-tuning with self-consistency substantially improves the performance on the two large molecules, with the MAE of ϵ_LUMO and ϵ_Δ reduced by an order, and at least 3x less for other properties. We note the inadequate performance of `zero-shot` is not due to insufficient training on QH9, since the validation Hamiltonian MAE of 29.06 μE_h is sufficiently low. The performance gap is due to the substantial scale gap between QH9 and MD22. This gap indicates generalization to larger-scale molecules is highly challenging. Remarkably, self-consistency training breaks the data limitation and achieves a significant performance improvement. Moreover, when considering the acceleration benefit for SCF, `zero-shot` prediction only brings a limited acceleration and even results in deceleration (on DHA). In contrast, self-consistency training consistently achieves more significant SCF acceleration. Appendix D.3 presents SCF acceleration results under the SOSCF iteration strategy, where the improvement by self-consistency training is even more significant.

Comparison to End-to-End Property Predictors. The conventional paradigm for molecular property prediction is to predict each property with a devoted model in an end-to-end (`e2e`) manner (Schütt et al., 2018; Gasteiger et al., 2020; Thölke & De Fabritiis, 2021; Liao & Smidt, 2022), while Hamiltonian prediction offers the advantage to provide all properties that DFT can provide using a single model. Moreover, self-consistency training differentiates Hamiltonian prediction from other property prediction tasks in that it enables continued improvement of Hamiltonian prediction

Table 5. Generalization to *larger-scale* molecules (from MD22) than previously reported in Hamiltonian prediction, with comparison to generalization results of `e2e` property predictors. Models are pretrained on QH9-full with labels and directly evaluated on the molecules (`zero-shot`, `e2e`), or, for the Hamiltonian model, after fine-tuned by `self-con`sistency without labels. Self-consistency training enables meaningful prediction on the larger molecules by bridging generalization gap, and significantly outperforms `e2e` predictors in molecular properties.

Molecule	Setting	$\mathbf{H}$ $[\mu E_\mathrm{h}]\downarrow$	$\boldsymbol{\epsilon}$ $[\mu E_\mathrm{h}]\downarrow$	$\mathbf{C}$ $[\%]\uparrow$	ϵ_HOMO $[\mu E_\mathrm{h}]\downarrow$	ϵ_LUMO $[\mu E_\mathrm{h}]\downarrow$	ϵ_Δ $[\mu E_\mathrm{h}]\downarrow$	SCF Accel. $[\%]\downarrow$
ALA3	zero-shot	237.71	6.54×10^3	52.24	6.90×10^3	9.51×10^4	9.79×10^4	84.6
	self-con	**52.49**	$\mathbf{1.22}\times10^3$	**94.46**	$\mathbf{2.07}\times10^3$	$\mathbf{3.76}\times10^3$	$\mathbf{2.69}\times10^3$	**64.7**
	e2e (ET)	N/A	N/A	N/A	1.74×10^5	7.72×10^3	2.38×10^5	N/A
	e2e (Equiformer)	N/A	N/A	N/A	2.38×10^5	1.16×10^4	2.27×10^5	N/A
DHA	zero-shot	397.87	1.84×10^4	20.15	1.11×10^4	1.90×10^5	1.85×10^5	170.8
	self-con	**56.12**	$\mathbf{1.81}\times10^3$	**83.51**	$\mathbf{1.99}\times10^3$	$\mathbf{4.01}\times10^3$	$\mathbf{2.34}\times10^3$	**67.0**
	e2e (ET)	N/A	N/A	N/A	2.92×10^5	2.58×10^4	3.39×10^5	N/A
	e2e (Equiformer)	N/A	N/A	N/A	3.76×10^5	2.31×10^4	4.17×10^5	N/A

without labeled data. This continued improvement can even spread to various molecular properties, which cannot be achieved by `e2e` predictors without additional labeled data.

We showcase this unique merit in this scenario of generalizing to larger-scale molecules, where the continued improvement could bridge the generalization gap. We compare the properties ϵ_HOMO, ϵ_LUMO and ϵ_Δ derived from the predicted Hamiltonian with the direct prediction results by the respective `e2e` predictors. Same as the Hamiltonian predictor, the `e2e` property predictors are trained with labels on nearly all QH9 molecules, but they do not have a self-consistency training strategy hence can only be directly applied to predict the respective properties on the much larger molecules. We consider two representative implementations of the `e2e` predictors, using the ET (Thölke & De Fabritiis, 2021) and the Equiformer (Liao & Smidt, 2022) architectures.

Results are shown in Table 5. They indicate that the `e2e` predictors significantly suffer from the generalization gap when comparing the evaluated error to the validation error (Table D.3) which are around $1\times10^3\,\mu E_\mathrm{h}$. The Hamiltonian predictor can also predict such properties as derived from the predicted Hamiltonian. Even applied right after pre-training (`zero-shot`), the Hamiltonian predictor can already provide better results than `e2e` predictors on ϵ_HOMO and ϵ_Δ. Using the unique lever of self-consistency training, the Hamiltonian predictor provides much more accurate results on all three properties, with one to two orders less MAE than `e2e` predictors. These results indicate that self-consistency training offers a promising avenue towards improving OOD prediction for molecular properties in a label-free manner.

4. Conclusion

We have presented self-consistency training for Hamiltonian prediction, a novel training method that does not require labeled data. This is a unique advantage of Hamiltonian prediction. As the self-consistency loss is designed to enforce the basic equation of DFT, it provides complete and exact information of the prediction target. Self-consistency training opens the access to gain supervision from vastly available unlabeled data, which substantially solves the data-scarce problem and allows generalization to challenging domains. We have also pointed out and empirically verified that self-consistency training is more efficient than running DFT to generate data for supervised learning, benefited from its amortization effect. Using self-consistency training, we have pushed Hamiltonian prediction to solve molecules larger than ever reported.

More broadly, since Hamiltonian matrix can derive rich molecular properties (*e.g.*, energy, HOMO-LUMO gap), the self-consistency training can also improve the prediction of these properties without labeled data, and can even supervise end-to-end prediction models. Since labeled data in the science domain in general is much less than in conventional AI domains, and generalization is more challenging due to the flexibility in the input, this way to leverage fundamental physical laws to train the model would be especially helpful.

Acknowledgements

We thank Lin Huang, Han Yang, and Yue Wang for insightful discussions on the idea and techniques; Erpai Luo for discussions on model design and help with dataset preparation; Jan Hermann, Michael Gastegger and Sebastian Ehlert for suggestions on evaluation and writing; Tao Qin, Jia Zhang and Huanhuan Xia for constructive feedback. Additionally, we acknowledge Lin Huang, Han Yang, and Jia Zhang for providing their CuDA implementation of Hamiltonian construction. We also thank anonymous reviewers and area chair for their feedback. He Zhang and Nanning Zheng were supported by the National Natural Science Foundation of China under Grant 62088102.

Impact Statement

This paper presents work whose goal is to advance the field of Machine Learning. There are many potential societal consequences of our work, none which we feel must be specifically highlighted here.

References

Becke, A. D. Density-functional thermochemistry. I. the effect of the exchange-only gradient correction. *The Journal of chemical physics*, 96(3):2155–2160, 1992.

Becke, A. D. Density-functional thermochemistry. III. The role of exact exchange. *The Journal of Chemical Physics*, 98, 1993. ISSN 00219606. doi: 10.1063/1.464913.

Cances, E. and Le Bris, C. On the convergence of SCF algorithms for the Hartree-Fock equations. *ESAIM: Mathematical Modelling and Numerical Analysis*, 34(4):749–774, 2000.

Chanussot, L., Das, A., Goyal, S., Lavril, T., Shuaibi, M., Riviere, M., Tran, K., Heras-Domingo, J., Ho, C., Hu, W., et al. Open catalyst 2020 (OC20) dataset and community challenges. *Acs Catalysis*, 11(10):6059–6072, 2021.

Chen, Y., Zhang, L., Wang, H., and E, W. DeePKS: A comprehensive data-driven approach toward chemically accurate density functional theory. *Journal of Chemical Theory and Computation*, 17(1):170–181, 2021.

Chmiela, S., Sauceda, H. E., Poltavsky, I., Müller, K.-R., and Tkatchenko, A. sGDML: Constructing accurate and data efficient molecular force fields using machine learning. *Computer Physics Communications*, 240:38–45, 2019.

Chmiela, S., Vassilev-Galindo, V., Unke, O. T., Kabylda, A., Sauceda, H. E., Tkatchenko, A., and Müller, K.-R. Accurate global machine learning force fields for molecules with hundreds of atoms. *Science Advances*, 9 (2):eadf0873, 2023.

del Mazo-Sevillano, P. and Hermann, J. Variational principle to regularize machine-learned density functionals: the non-interacting kinetic-energy functional. *arXiv preprint arXiv:2306.17587*, 2023.

Dick, S. and Fernandez-Serra, M. Machine learning accurate exchange and correlation functionals of the electronic density. *Nature communications*, 11(1):3509, 2020.

Ditchfield, R., Hehre, W. J., and Pople, J. A. Self-consistent molecular-orbital methods. ix. an extended gaussian-type basis for molecular-orbital studies of organic molecules. *The Journal of Chemical Physics*, 54(2):724–728, 1971.

Dunlap, B. I. Robust and variational fitting. *Phys. Chem. Chem. Phys.*, 2:2113–2116, 2000. doi: 10.1039/B000027M. URL http://dx.doi.org/10.1039/B000027M.

Dunning Jr, T. H. Gaussian basis sets for use in correlated molecular calculations. i. the atoms boron through neon and hydrogen. *The Journal of chemical physics*, 90(2): 1007–1023, 1989.

Fey, M. and Lenssen, J. E. Fast graph representation learning with PyTorch Geometric. *arXiv preprint arXiv:1903.02428*, 2019.

Gastegger, M., McSloy, A., Luya, M., Schütt, K. T., and Maurer, R. J. A deep neural network for molecular wave functions in quasi-atomic minimal basis representation. *The Journal of Chemical Physics*, 153(4), 2020.

Gasteiger, J., Groß, J., and Günnemann, S. Directional message passing for molecular graphs. *arXiv preprint arXiv:2003.03123*, 2020.

Gong, X., Li, H., Zou, N., Xu, R., Duan, W., and Xu, Y. General framework for E(3)-equivariant neural network representation of density functional theory Hamiltonian. *Nature Communications*, 14(1):2848, 2023.

Gu, Q., Zhang, L., and Feng, J. Neural network representation of electronic structure from ab initio molecular dynamics. *Science Bulletin*, 67(1):29–37, 2022.

Hegde, G. and Bowen, R. C. Machine-learned approximations to density functional theory hamiltonians. *Scientific reports*, 7(1):42669, 2017.

Hellweg, A. and Rappoport, D. Development of new auxiliary basis functions of the karlsruhe segmented contracted basis sets including diffuse basis functions (def2-svpd, def2-tzvppd, and def2-qvppd) for ri-mp2 and ri-cc calculations. *Physical Chemistry Chemical Physics*, 17(2): 1010–1017, 2015.

Hohenberg, P. and Kohn, W. Inhomogeneous electron gas. *Physical review*, 136(3B):B864, 1964.

Imoto, F., Imada, M., and Oshiyama, A. Order-N orbital-free density-functional calculations with machine learning of functional derivatives for semiconductors and metals. *Physical Review Research*, 3(3):033198, 2021.

Ionescu, C., Vantzos, O., and Sminchisescu, C. Matrix backpropagation for deep networks with structured layers. In *Proceedings of the IEEE international conference on computer vision*, pp. 2965–2973, 2015.

Jain, A., Ong, S. P., Hautier, G., Chen, W., Richards, W. D., Dacek, S., Cholia, S., Gunter, D., Skinner, D., Ceder,

G., and Persson, K. A. Commentary: The Materials Project: A materials genome approach to accelerating materials innovation. *APL Materials*, 1(1):011002, 07 2013. ISSN 2166-532X. doi: 10.1063/1.4812323. URL https://doi.org/10.1063/1.4812323.

Jensen, F. Polarization consistent basis sets: Principles. *The Journal of Chemical Physics*, 115(20):9113–9125, 2001.

Jordaan, M. A., Ebenezer, O., Damoyi, N., and Shapi, M. Virtual screening, molecular docking studies and dft calculations of fda approved compounds similar to the non-nucleoside reverse transcriptase inhibitor (nnrti) efavirenz. *Heliyon*, 6(8), 2020.

Kirkpatrick, J., McMorrow, B., Turban, D. H., Gaunt, A. L., Spencer, J. S., Matthews, A. G., Obika, A., Thiry, L., Fortunato, M., Pfau, D., et al. Pushing the frontiers of density functionals by solving the fractional electron problem. *Science*, 374(6573):1385–1389, 2021.

Kohn, W. and Sham, L. J. Self-consistent equations including exchange and correlation effects. *Physical review*, 140(4A):A1133, 1965.

Kudin, K. N., Scuseria, G. E., and Cancès, E. A black-box self-consistent field convergence algorithm: One step closer. *The Journal of Chemical Physics*, 116 (19):8255–8261, 04 2002. ISSN 0021-9606. doi: 10.1063/1.1470195. URL https://doi.org/10.1063/1.1470195.

Levy, M. Universal variational functionals of electron densities, first-order density matrices, and natural spin-orbitals and solution of the v-representability problem. *Proceedings of the National Academy of Sciences*, 76(12):6062–6065, 1979.

Li, H., Wang, Z., Zou, N., Ye, M., Xu, R., Gong, X., Duan, W., and Xu, Y. Deep-learning density functional theory Hamiltonian for efficient ab initio electronic-structure calculation. *Nature Computational Science*, 2(6):367–377, 2022.

Liao, Y.-L. and Smidt, T. Equiformer: Equivariant graph attention transformer for 3d atomistic graphs. *arXiv preprint arXiv:2206.11990*, 2022.

Lieb, E. H. Density functionals for Coulomb systems. *International Journal of Quantum Chemistry*, 24(3):243–277, 1983.

Nigam, J., Willatt, M. J., and Ceriotti, M. Equivariant representations for molecular Hamiltonians and N-center atomic-scale properties. *The Journal of Chemical Physics*, 156(1), 2022.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019.

Perdew, J. P., Burke, K., and Ernzerhof, M. Generalized gradient approximation made simple. *Physical review letters*, 77(18):3865, 1996.

Perdew, J. P., Ruzsinszky, A., Csonka, G. I., Vydrov, O. A., Scuseria, G. E., Constantin, L. A., Zhou, X., and Burke, K. Restoring the density-gradient expansion for exchange in solids and surfaces. *Physical review letters*, 100(13): 136406, 2008.

Pulay, P. Convergence acceleration of iterative sequences. the case of scf iteration. *Chemical Physics Letters*, 73(2): 393–398, 1980.

Pulay, P. Improved SCF convergence acceleration. *Journal of Computational Chemistry*, 3(4):556–560, 1982. doi: https://doi.org/10.1002/jcc.540030413. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/jcc.540030413.

Ramakrishnan, R., Dral, P. O., Rupp, M., and Von Lilienfeld, O. A. Quantum chemistry structures and properties of 134 kilo molecules. *Scientific data*, 1(1):1–7, 2014.

Remme, R., Kaczun, T., Scheurer, M., Dreuw, A., and Hamprecht, F. A. KineticNet: Deep learning a transferable kinetic energy functional for orbital-free density functional theory. *arXiv preprint arXiv:2305.13316*, 2023.

Schütt, K. T., Sauceda, H. E., Kindermans, P.-J., Tkatchenko, A., and Müller, K.-R. Schnet–a deep learning architecture for molecules and materials. *The Journal of Chemical Physics*, 148(24), 2018.

Schütt, K. T., Gastegger, M., Tkatchenko, A., Müller, K.-R., and Maurer, R. J. Unifying machine learning and quantum chemistry with a deep neural network for molecular wavefunctions. *Nature communications*, 10(1):5024, 2019.

Seminario, J. M. Recent developments and applications of modern density functional theory. 1996.

Shmilovich, K., Willmott, D., Batalov, I., Kornbluth, M., Mailoa, J., and Kolter, J. Z. Orbital Mixer: Using atomic orbital features for basis-dependent prediction of molecular wavefunctions. *Journal of Chemical Theory and Computation*, 18(10):6021–6030, 2022.

Slater, J. C. A simplification of the Hartree-Fock method. *Physical review*, 81(3):385, 1951.

Snyder, J. C., Rupp, M., Hansen, K., Müller, K.-R., and Burke, K. Finding density functionals with machine learning. *Physical review letters*, 108(25):253002, 2012.

Song, Y., Sebe, N., and Wang, W. Why approximate matrix square root outperforms accurate SVD in global covariance pooling? In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pp. 1115–1123, 2021.

Stephens, P. J., Devlin, F. J., Chabalowski, C. F., and Frisch, M. J. Ab initio calculation of vibrational absorption and circular dichroism spectra using density functional force fields. *The Journal of physical chemistry*, 98(45):11623–11627, 1994.

Sun, Q. Co-iterative augmented hessian method for orbital optimization. *arXiv preprint arXiv:1610.08423*, 2016.

Sun, Q., Yang, J., and Chan, G. K.-L. A general second order complete active space self-consistent-field solver for large-scale systems. *Chemical Physics Letters*, 683: 291–299, 2017.

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. PySCF: the python-based simulations of chemistry framework. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 8(1):e1340, 2018.

Teale, A. M., Helgaker, T., Savin, A., Adamo, C., Aradi, B., Arbuznikov, A. V., Ayers, P. W., Baerends, E. J., Barone, V., Calaminici, P., et al. DFT exchange: sharing perspectives on the workhorse of quantum chemistry and materials science. *Physical chemistry chemical physics*, 24(47):28700–28781, 2022.

Thölke, P. and De Fabritiis, G. Equivariant transformers for neural network based molecular potentials. In *International Conference on Learning Representations*, 2021.

Unke, O., Bogojeski, M., Gastegger, M., Geiger, M., Smidt, T., and Müller, K.-R. SE(3)-equivariant prediction of molecular wavefunctions and electronic densities. *Advances in Neural Information Processing Systems*, 34: 14434–14447, 2021.

Wang, W., Dang, Z., Hu, Y., Fua, P., and Salzmann, M. Backpropagation-friendly eigendecomposition. *Advances in Neural Information Processing Systems*, 32, 2019.

Wang, W., Dang, Z., Hu, Y., Fua, P., and Salzmann, M. Robust differentiable svd. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(9):5472–5487, 2021a.

Wang, Y. A., Govind, N., and Carter, E. A. Orbital-free kinetic-energy density functionals with a density-dependent kernel. *Physical Review B*, 60(24):16350, 1999.

Wang, Z., Ye, S., Wang, H., He, J., Huang, Q., and Chang, S. Machine learning method for tight-binding hamiltonian parameterization from ab-initio band structure. *npj Computational Materials*, 7(1):11, 2021b.

Witt, W. C., Beatriz, G., Dieterich, J. M., and Carter, E. A. Orbital-free density functional theory for materials research. *Journal of Materials Research*, 33(7):777–795, 2018.

Yin, S., Zhu, X., Gao, T., Zhang, H., Wu, F., and He, L. Harmonizing covariance and expressiveness for deep Hamiltonian regression in crystalline material research: a hybrid cascaded regression framework. *arXiv preprint arXiv:2401.00744*, 2024.

Yu, H., Liu, M., Luo, Y., Strasser, A., Qian, X., Qian, X., and Ji, S. QH9: A quantum hamiltonian prediction benchmark for QM9 molecules. *arXiv preprint arXiv:2306.09549*, 2023a.

Yu, H., Xu, Z., Qian, X., Qian, X., and Ji, S. Efficient and equivariant graph networks for predicting quantum Hamiltonian. *arXiv preprint arXiv:2306.04922*, 2023b.

Zhang, H., Liu, S., You, J., Liu, C., Zheng, S., Lu, Z., Wang, T., Zheng, N., and Shao, B. Overcoming the barrier of orbital-free density functional theory for molecular systems using deep learning. *Nature Computational Science*, pp. 1–14, 2024.

Zhang, L., Onat, B., Dusson, G., McSloy, A., Anand, G., Maurer, R. J., Ortner, C., and Kermode, J. R. Equivariant analytical mapping of first principles Hamiltonians to accurate and transferable materials models. *Npj Computational Materials*, 8(1):158, 2022.

Zhong, Y., Yu, H., Su, M., Gong, X., and Xiang, H. Transferable equivariant graph neural networks for the hamiltonians of molecules and solids. *npj Computational Materials*, 9(1):182, 2023.

A. Brief Introduction to Density Functional Theory

A.1. Background of Electronic Structure Methods

All properties of a molecule is determined by the result of interaction among the electrons and nuclei in the molecule. As nuclei are much heavier than electrons, they are typically treated as classical particles, while the electrons are governed by the Schrödinger equation. Therefore, the state of the A nuclei is specified by the molecular structure $\mathcal{M} := \{\mathcal{R}, \mathcal{Z}\}$, where $\mathcal{Z} := \{Z^{(a)}\}_{a=1}^{A}$ and $\mathcal{R} := \{\mathbf{R}^{(a)}\}_{a=1}^{A}$ specifies the atomic numbers (species) and coordinates of the nuclei, while the state of the N electrons is specified by the wavefunction $\psi(\mathbf{r}^{(1)}, \cdots, \mathbf{r}^{(N)})$. The squared modulus of the wavefunction $\left|\psi(\mathbf{r}^{(1)}, \cdots, \mathbf{r}^{(N)})\right|^2$ represents the joint distribution of the N electrons. Since electrons are indistinguishable, the density function $\left|\psi(\mathbf{r}^{(1)}, \cdots, \mathbf{r}^{(N)})\right|^2$ is permutation symmetric, hence the wavefunction $\psi(\mathbf{r}^{(1)}, \cdots, \mathbf{r}^{(N)})$ is permutation symmetric or antisymmetric, *i.e.*, it keeps or changes sign (*i.e.*, phase change of 0 or π) when the coordinates of two particles are exchanged. For electrons, their statistical behavior indicates that their wavefunction is antisymmetric (electrons are fermions).

Commonly, only the stationary states of electrons in a given molecular structure $\mathcal{M}$ are concerned, since the evolution of electrons is much faster than the motion of nuclei so their state instantly becomes stationary for any given molecular structure. The stationary states are determined by the stationary Schrödinger equation: $\hat{\mathcal{H}}_{\mathcal{M}}\psi = E\psi$, *i.e.*, they are eigenstates of the Hamiltonian operator $\hat{\mathcal{H}}_{\mathcal{M}}$. The Hamiltonian operator $\hat{\mathcal{H}}_{\mathcal{M}} := \hat{T} + \hat{V}_{\text{ee}} + \hat{V}_{\text{ext},\mathcal{M}}$ is composed of the kinetic energy operator $\hat{T}\psi := -\frac{1}{2}\sum_{i=1}^{N} \nabla^2_{\mathbf{r}^{(i)}}\psi$ (atomic units are used throughout), the internal potential energy operator among electrons $\hat{V}_{\text{ee}}\psi := \sum_{1\leqslant i<j\leqslant N} \frac{1}{\|\mathbf{r}^{(i)}-\mathbf{r}^{(j)}\|}\psi$, and the external potential energy operator $\hat{V}_{\text{ext},\mathcal{M}}\psi := \sum_{i=1}^{N} V_{\text{ext},\mathcal{M}}(\mathbf{r}^{(i)})\psi$ where $V_{\text{ext},\mathcal{M}}(\mathbf{r}) := -\sum_{a=1}^{A} \frac{Z^{(a)}}{\left\|\mathbf{r}-\mathbf{R}^{(a)}\right\|}$ is the external potential generated by the nuclei. Note that the latter two operators are multiplicative, *i.e.*, their action on a wavefunction is the multiplication with the corresponding potential function. The Hamiltonian operator is Hermitian, hence its eigenvalues are always real. Moreover, since this Hamiltonian operator is also real (in physics term, it is time-reversible) (particularly, it does not involve magnetic fields or spin-orbital coupling), every eigenstate of it has a real-valued eigenfunction. Hence from now on, it suffices to only consider real-valued wavefunctions.

Solving an eigenvalue problem is challenging especially when N is large. On the other hand, most of the concerned properties of a molecule only involve the ground state, *i.e.*, the eigenstate with the lowest eigenvalue (energy). Hence an alternative form to solve the electronic ground state can be composed as an optimization problem, known as the variational formulation:

$$E^{\star}_{\mathcal{M}} = \min_{\psi:\text{ antisym},\langle\psi|\psi\rangle=1} \langle\psi|\hat{\mathcal{H}}_{\mathcal{M}}|\psi\rangle, \tag{A.1}$$

where $\langle\psi|\phi\rangle$ denotes the integral of $\psi^*\phi$ w.r.t all their arguments, and $\langle\psi|\hat{\mathcal{H}}_{\mathcal{M}}|\phi\rangle := \langle\psi|\hat{\mathcal{H}}_{\mathcal{M}}\phi\rangle$ (which is also $\langle\hat{\mathcal{H}}_{\mathcal{M}}\psi|\phi\rangle$ since $\hat{\mathcal{H}}_{\mathcal{M}}$ is Hermitian). Various ways to parameterize the wavefunction ψ and estimate and optimize the energy are proposed. Regardless, this formulation optimizes a function on $\mathbb{R}^{3N}$, whose complexity may increase exponentially w.r.t system size N, limiting the scale of practically applicable systems.

Before going on, we note that although wavefunctions are in general complex-valued, it is sufficient to only consider real-valued wavefunctions for solving static (*i.e.*, no time evolution) electronic state without magnetic fields and ignoring spin-orbital coupling, since the Hamiltonian operator $\hat{\mathcal{H}}_{\mathcal{M}}$ in such cases are not only Hermitian but also time-reversible (meaning that $\hat{\mathcal{H}}_{\mathcal{M}}$ is "real", $\hat{\mathcal{H}}^*_{\mathcal{M}} = \hat{\mathcal{H}}_{\mathcal{M}}$), each of whose eigenstate has a real-valued eigenfunction. For this reason, we only consider real-valued functions (including orbitals and basis functions), and do not distinguish matrix transpose and Hermitian conjugate.

A.2. Basic Idea of DFT

Density functional theory (DFT) is motivated to address the exponentially complex optimization space. It aims to optimize the (one-electron reduced) density $\rho(\mathbf{r})$, a function on a fixed-dimensional space $\mathbb{R}^3$. It is a reduced quantity to describe the electronic state. The electron density corresponding to the electronic state specified by wavefunction ψ is the marginal distribution (up to a factor of the number of electrons) of the joint distribution:

$$\rho_{[\psi]}(\mathbf{r}) := N \int |\psi(\mathbf{r}, \mathbf{r}^{(2)}, \cdots, \mathbf{r}^{(N)})|^2 \, \mathrm{d}\mathbf{r}^{(2)} \cdots \mathrm{d}\mathbf{r}^{(N)}, \tag{A.2}$$

which is independent of the variable for which the marginalization is conducted due to the indistinguishability. It is how one straightforwardly perceives electron density, which is a valid concept also under the classical view.

Now the question is, whether optimizing the density is sufficient to determine the electronic ground state, considering that the density is only a reduced quantity. This is first answered affirmatively in the seminal work by Hohenberg & Kohn (1964), but it would be more explicit to deduce the answer following Levy's constrained search formulation (Levy, 1979) of Eq. (A.1):

$$E^\star_{\mathcal{M}} = \min_{\psi\colon \text{antisym}, \langle\psi|\psi\rangle = N} \langle\psi|\hat{\mathcal{H}}_{\mathcal{M}}|\psi\rangle = \min_{\rho \geqslant 0, \langle\mathbb{1}|\rho\rangle = N} \Big(\min_{\psi\colon \text{antisym}, \rho_{[\psi]} = \rho} \langle\psi|\hat{\mathcal{H}}_{\mathcal{M}}|\psi\rangle \Big), \tag{A.3}$$

where $\mathbb{1}$ denotes the constant 1-valued function. Note that when viewing $\min_{\psi\colon \text{antisym}, \rho_{[\psi]}=\rho} \langle\psi|\hat{\mathcal{H}}_{\mathcal{M}}|\psi\rangle$ as a functional of density ρ, the optimization problem in Eq. (A.3) indicates that the ground-state energy and density can indeed be solved by optimizing the density. Among the three components of $\hat{\mathcal{H}}_{\mathcal{M}}$, the $\hat{V}_{\text{ext},\mathcal{M}}$ term already makes a density functional, since $\langle\psi|\hat{V}_{\text{ext},\mathcal{M}}|\psi\rangle = \sum_{i=1}^N \int V_{\text{ext},\mathcal{M}}(\mathbf{r}^{(i)}) \big|\psi(\mathbf{r}^{(1)}, \cdots, \mathbf{r}^{(N)})\big|^2 \,\mathrm{d}\mathbf{r}^{(1)} \cdots \mathrm{d}\mathbf{r}^{(N)} = \frac{1}{N} \sum_{i=1}^N \int V_{\text{ext},\mathcal{M}}(\mathbf{r}^{(i)}) \rho_{[\psi]}(\mathbf{r}^{(i)}) \,\mathrm{d}\mathbf{r}^{(i)} = \langle V_{\text{ext},\mathcal{M}}|\rho_{[\psi]}\rangle$ is independent of ψ once $\rho_{[\psi]}$ is fixed. So Eq. (A.3) can be formulated as:

$$E^\star_{\mathcal{M}} = \min_{\rho \geqslant 0, \langle\mathbb{1}|\rho\rangle = N} \underbrace{\Big(\min_{\psi\colon \text{antisym}, \rho_{[\psi]} = \rho} \langle\psi|\hat{T} + \hat{V}_{\text{ee}}|\psi\rangle \Big)}_{=: F[\rho]} + \langle V_{\text{ext},\mathcal{M}}|\rho\rangle,$$

where $F[\rho]$ is called the universal functional comprising the kinetic and internal potential energy minimally attainable for the given density ρ. Its name follows the fact that it does not depend on molecular structure $\mathcal{M}$ and applies to any system.

As the universal functional is still quite implicit to carry out practical calculation, approximations are considered to cover the major part of the kinetic energy and of the internal potential energy. For the latter, the classical internal potential energy can be used, which ignores electron correlation and adopts an explicit expression in terms of $\rho(\mathbf{r})$:

$$E_{\text{H}}[\rho] := \frac{1}{2} \int \frac{\rho(\mathbf{r})\rho(\mathbf{r}')}{\|\mathbf{r} - \mathbf{r}'\|} \,\mathrm{d}\mathbf{r}\mathrm{d}\mathbf{r}'. \tag{A.4}$$

It is also called the Hartree energy, hence the notation. For the kinetic part, the kinetic energy density functional (KEDF) is introduced following a similar formulation as the definition of the universal functional:

$$T_{\text{S}}[\rho] := \min_{\psi\colon \text{antisym}, \rho_{[\psi]} = \rho} \langle\psi|\hat{T}|\psi\rangle. \tag{A.5}$$

The rest part of the kinetic and internal potential energy is called the exchange-correlation (XC) energy:

$$E_{\text{XC}}[\rho] := F[\rho] - T_{\text{S}}[\rho] - E_{\text{H}}[\rho],$$

and the variational problem to solve the electronic ground state of molecule in structure $\mathcal{M}$ becomes:

$$E^\star_{\mathcal{M}} = \min_{\rho \geqslant 0, \langle\mathbb{1}|\rho\rangle = N} T_{\text{S}}[\rho] + E_{\text{H}}[\rho] + E_{\text{XC}}[\rho] + \langle V_{\text{ext},\mathcal{M}}|\rho\rangle. \tag{A.6}$$

Although the exact expression of E_{XC} in terms of ρ is still unknown, it makes only a minor part of the total electronic energy and is more flexible to approximate. Over the past decades, researchers have developed many successful approximations (Becke, 1993; Stephens et al., 1994; Perdew et al., 1996; 2008). Deep learning has also been leveraged for developing an approximation (Dick & Fernandez-Serra, 2020; Chen et al., 2021; Kirkpatrick et al., 2021). As for the KEDF, there are methods that also directly approximate the density functional (Slater, 1951; Wang et al., 1999; Witt et al., 2018), which are now called orbital-free density functional theory. Nevertheless, approximating KEDF is harder and requires higher accuracy, since it accounts for a major part of energy. It is also an active research direction to leverage machine learning models to approximate the functional more accurately (Snyder et al., 2012; Imoto et al., 2021; Remme et al., 2023; del Mazo-Sevillano & Hermann, 2023; Zhang et al., 2024).

A.3. Kohn-Sham DFT

Considering the difficulty of directly approximating the KEDF, Kohn & Sham (1965) exploited properties of the KEDF and developed a method that evaluates the kinetic energy directly. Note that in the optimization of the definition of KEDF in

Eq. (A.5), there is no interaction among electrons ($\hat{T}$ operates one-body-wise). It is known that the ground-state wavefunction solution of non-interacting systems is in the form of a determinant (at least in absense of degeneracy (Lieb, 1983, Thm. 4.6)), which, instead of a general function on $\mathbb{R}^{3N}$, is composed of N functions $\Phi := \{\phi_i(\mathbf{r})\}_{i=1}^N$ on $\mathbb{R}^3$ called orbitals:

$$\psi_{[\Phi]}(\mathbf{r}^{(1)}, \cdots, \mathbf{r}^{(N)}) := \frac{1}{\sqrt{N!}} \det[\phi_i(\mathbf{r}^{(j)})]_{ij}. \tag{A.7}$$

The optimization problem in the definition of KEDF in Eq. (A.5) can be equivalently formulated as:[3]

$$T_{\mathrm{S}}[\rho] = \min_{\{\phi_i\}_{i=1}^N : \rho_{[\psi_{[\Phi]}]}=\rho} \langle \psi_{[\Phi]} | \hat{T} | \psi_{[\Phi]} \rangle = \min_{\substack{\{\phi_i\}_{i=1}^N : \text{orthonormal}, \\ \rho_{[\psi_{[\Phi]}]}=\rho}} \sum_{i=1}^N \langle \phi_i | \hat{T} | \phi_i \rangle, \tag{A.8}$$

where the second expression to optimize orthonormal orbitals, $\langle \phi_i | \phi_j \rangle = \delta_{ij}$, is valid since any set of functions can be orthonormalized by *e.g.*, the Gram-Schmidt process without changing the corresponding density and kinetic energy. This is desired to simplify calculation, for which the kinetic energy calculation is simplified in Eq. (A.8), and the density (Eq. (A.2)) substituted by Eq. (A.7) can also be simplified as:

$$\rho_{[\psi_{[\Phi]}]}(\mathbf{r}) = \sum_{i=1}^N |\phi_i(\mathbf{r})|^2.$$

Using this simplified formulation Eq. (A.8), the original optimization problem Eq. (A.6) for solving the electronic structure given molecular structure $\mathcal{M}$ becomes:

$$\begin{aligned} E^\star_{\mathcal{M}} &= \min_{\rho : \geqslant 0, \langle \mathbb{1} | \rho \rangle = N} \left\{ \left(\min_{\substack{\{\phi_i\}_{i=1}^N : \text{orthonormal}, \\ \rho_{[\psi_{[\Phi]}]}=\rho}} \sum_{i=1}^N \langle \phi_i | \hat{T} | \phi_i \rangle \right) + E_{\mathrm{H}}[\rho] + E_{\mathrm{XC}}[\rho] + \langle V_{\mathrm{ext},\mathcal{M}} | \rho \rangle \right\} \\ &= \min_{\substack{\{\phi_i\}_{i=1}^N : \\ \text{orthonormal}}} \left\{ E_{\mathcal{M}}[\Phi] := \sum_{i=1}^N \langle \phi_i | \hat{T} | \phi_i \rangle + E_{\mathrm{H}}[\rho_{[\psi_{[\Phi]}]}] + E_{\mathrm{XC}}[\rho_{[\psi_{[\Phi]}]}] + \langle V_{\mathrm{ext},\mathcal{M}} | \rho_{[\psi_{[\Phi]}]} \rangle \right\}. \end{aligned} \tag{A.9}$$

In this way, the query for directly evaluating $T_{\mathrm{S}}[\rho]$ is avoided by an exact estimation using the orbitals. This formulation optimizes N functions on $\mathbb{R}^3$ instead of one function on $\mathbb{R}^3$, hence the complexity is increased by at least an order of N. This formulation is called the Kohn-Sham DFT, and has become the default DFT formulation due to its success to solve molecular system problems computationally (Seminario, 1996; Jain et al., 2013).

To solve Eq. (A.9), standard DFT solves the equation of optimality, which is derived by taking the variation of $E_{\mathcal{M}}[\Phi]$ w.r.t each orbital under the orthonormality constraint. The variation of $E_{\mathcal{M}}[\Phi]$ is:

$$\begin{aligned} \frac{\delta E_{\mathcal{M}}[\Phi]}{\delta \phi_i}(\mathbf{r}) &= \frac{\delta \sum_{j=1}^N (-\frac{1}{2}) \langle \phi_j | \nabla^2 | \phi_j \rangle}{\delta \phi_i}(\mathbf{r}) + \int \frac{\delta \big(E_{\mathrm{H}}[\rho] + E_{\mathrm{XC}}[\rho] + \langle V_{\mathrm{ext},\mathcal{M}} | \rho \rangle \big)}{\delta \rho(\mathbf{r}')} \Big|_{\rho=\rho_{[\psi_{[\Phi]}]}} \frac{\delta \rho_{[\psi_{[\Phi]}]}(\mathbf{r}')}{\delta \phi_i(\mathbf{r})} \, \mathrm{d}\mathbf{r}' \\ &= 2\hat{T}\phi_i(\mathbf{r}) + 2 \Bigg(\underbrace{\int \frac{\rho(\mathbf{r}')}{\|\mathbf{r}' - \mathbf{r}\|} \, \mathrm{d}\mathbf{r}'}_{=: V_{\mathrm{H}[\rho]}(\mathbf{r})} + \underbrace{\frac{\delta E_{\mathrm{XC}}[\rho]}{\delta \rho}(\mathbf{r})}_{=: V_{\mathrm{XC}[\rho]}(\mathbf{r})} \Bigg) \Bigg|_{\rho=\rho_{[\psi_{[\Phi]}]}} \phi_i(\mathbf{r}) + 2V_{\mathrm{ext},\mathcal{M}}(\mathbf{r})\phi_i(\mathbf{r}). \end{aligned} \tag{A.10}$$

By introducing the (one-electron effective) Hamiltonian operator, or more commonly called the Fock operator in DFT,

$$\hat{H}_{\mathcal{M},[\rho]} := \hat{T} + \hat{V}_{\mathrm{H}[\rho]} + \hat{V}_{\mathrm{XC}[\rho]} + \hat{V}_{\mathrm{ext},\mathcal{M}}, \tag{A.11}$$

where the latter three operators act on a function by multiplying the function with the respective potential energy function, the variation can be written as:

$$\frac{\delta E_{\mathcal{M}}[\Phi]}{\delta \phi_i}(\mathbf{r}) = 2\hat{H}_{\mathcal{M},[\rho_{[\psi_{[\Phi]}]}]}\phi_i. \tag{A.12}$$

[3] Assume the queried density ρ comes from the set of densities of the ground state of all non-interacting systems. Although this set still has ρ that violates the equivalence to Eq. (A.5) (Lieb, 1983, Thm. 4.8), the determinantal definition Eq. (A.8) still recovers the ground-state energy if optimized on this set (Lieb, 1983, Thm. 4.9).

For the orthonormality constraint, first consider the normalization constraint and introduce Lagrange multipliers $\{\varepsilon_i\}_{i=1}^N$ for them. The corresponding variation is:

$$\frac{\delta \sum_{j=1}^N \varepsilon_j(\langle\phi_j|\phi_j\rangle - 1)}{\delta\phi_i}(\mathbf{r}) = 2\varepsilon_i\phi_i(\mathbf{r}),$$

which leads to the optimality equation:

$$\hat{H}_{\mathcal{M},[\rho_{[\psi_{[\Phi]}]}]}\phi_i(\mathbf{r}) = \varepsilon_i\phi_i(\mathbf{r}), \quad \forall i = 1, \cdots, N. \tag{A.13}$$

This is known as the Kohn-Sham equation (in function form). From this equation, the optimal solution of orbitals are eigenstates of the operator $\hat{H}_{\mathcal{M},[\rho_{[\psi_{[\Phi]}]}]}$, which can be verified to be Hermitian. Hence, in the general case where there is no degeneracy, different orbitals in the solution are naturally orthogonal, so there is no need to further enforce this constraint explicitly.

A.4. Practical Calculation under a Basis

Vectorizing a function as the expansion coefficient vector on a basis function set is an effective and controllable way to represent a function numerically. For molecules, as the electrons distribute around atoms in the molecule, commonly adopted basis functions are atom-centered functions. To allow analytical calculation of integrals, the functions typically take a Gaussian form for the radial variable (*i.e.*, the distance from the center nucleus of this basis function) multiplied with a spherical harmonic function for the angular variables (or equivalently a monomial of the three coordinates) (Ditchfield et al., 1971; Hellweg & Rappoport, 2015; Dunning Jr, 1989; Jensen, 2001). Different chemical elements usually have different sets of basis functions. To expand the orbitals in a molecule, the basis set is the union of basis functions centered at each of the atoms in the molecule. We collectively label them with one index α, and denote them as $\{\eta_{\mathcal{M},\alpha}(\mathbf{r})\}_{\alpha=1}^B$. The number of basis functions B for a molecular system typically increases linearly with the number of electrons N in the system.

The orbitals can then be represented as expansion coefficients $\mathbf{C}$:

$$\phi_i(\mathbf{r}) = \sum_{\alpha=1}^B \mathbf{C}_{\alpha i}\,\eta_{\mathcal{M},\alpha}(\mathbf{r}). \tag{A.14}$$

Next we show the derivation for the optimality equation for $\mathbf{C}$. Given that orthonormality constraint is satisfied, the density corresponding to the orbital state specified by $\mathbf{C}$ is:

$$\rho_{\mathcal{M},\mathbf{C}}(\mathbf{r}) = \sum_{\alpha,\beta}\sum_{i=1}^N \mathbf{C}_{\alpha i}\,\mathbf{C}_{\beta i}\,\eta_{\mathcal{M},\alpha}(\mathbf{r})\,\eta_{\mathcal{M},\beta}(\mathbf{r}) = \sum_{\alpha,\beta}(\mathbf{C}\mathbf{C}^\top)_{\alpha\beta}\,\eta_{\mathcal{M},\alpha}(\mathbf{r})\,\eta_{\mathcal{M},\beta}(\mathbf{r}). \tag{A.15}$$

The Kohn-Sham equation presented in Eq. (A.13) is turned into $\sum_\alpha \mathbf{C}_{\alpha i}\hat{H}_{\mathcal{M},[\rho_{\mathcal{M},\mathbf{C}}]}\eta_{\mathcal{M},\alpha}(\mathbf{r}) = \sum_\alpha \varepsilon_i\mathbf{C}_{\alpha i}\eta_{\mathcal{M},\alpha}(\mathbf{r})$. Integrating both sides with basis function $\eta_{\mathcal{M},\beta}(\mathbf{r})$ gives:

$$\mathbf{H}_{\mathcal{M}}(\mathbf{C})\,\mathbf{C} = \mathbf{S}\,\mathbf{C}\,\boldsymbol{\varepsilon}, \tag{A.16}$$

where:

$$\big(\mathbf{H}_{\mathcal{M}}(\mathbf{C})\big)_{\alpha\beta} := \langle\eta_{\mathcal{M},\alpha}|\hat{H}_{\mathcal{M},[\rho_{\mathcal{M},\mathbf{C}}]}|\eta_{\mathcal{M},\beta}\rangle, \tag{A.17}$$

is the Hamiltonian matrix ($\hat{H}_{\mathcal{M},[\rho_{\mathcal{M},\mathbf{C}}]}$ defined in Eq. (A.11)),

$$(\mathbf{S}_{\mathcal{M}})_{\alpha\beta} := \langle\eta_{\mathcal{M},\alpha}|\eta_{\mathcal{M},\beta}\rangle,$$

is the overlap matrix of the atomic basis, and

$$\boldsymbol{\varepsilon} := \mathrm{Diag}(\varepsilon_1, \cdots, \varepsilon_N),$$

is a diagonal matrix comprising the eigenvalues. This is the matrix form of the Kohn-Sham equation Eq. (A.13), as presented in Eq. (1) in the main paper.

To solve Eq. (A.16), conventional DFT calculation uses a fixed-point iteration process known as the self-consistent field (SCF) iteration. At each iteration step k, the last orbital solution $\mathbf{C}^{(k-1)}$ is used to construct the Hamiltonian matrix $\mathbf{H}^{(k)} := \mathbf{H}_{\mathcal{M}}(\mathbf{C}^{(k-1)})$, and the updated orbital solution $\mathbf{C}^{(k)}$ for this step is derived by solving $\mathbf{H}^{(k)}\mathbf{C} = \mathbf{S}\mathbf{C}\boldsymbol{\varepsilon}$. There are variants that accelerate the iteration, *e.g.*, the direct inversion in the iterative subspace (DIIS) method (Pulay, 1982; Kudin et al., 2002), which constructs $\mathbf{H}^{(k)}$ not only using $\mathbf{H}_{\mathcal{M}}(\mathbf{C}^{(k-1)})$ but also using Hamiltonian matrices from previous steps.

In contrast, our self-consistency approach (Eq. (3)) solves Eq. (A.16) directly, by minimizing the violation of the equality in terms of the Hamiltonian matrix, $\left\|\hat{\mathbf{H}}_\theta(\mathcal{M}) - \mathbf{H}_{\mathcal{M}}\Big(\mathbf{C}_{\mathcal{M}}\big(\hat{\mathbf{H}}_\theta(\mathcal{M})\big)\Big)\right\|_{\mathrm{F}}^2$, where $\mathbf{C}_{\mathcal{M}}\big(\hat{\mathbf{H}}_\theta(\mathcal{M})\big)$ denotes the orbital coefficients solved from $\hat{\mathbf{H}}_\theta(\mathcal{M})\mathbf{C} = \mathbf{S}_{\mathcal{M}}\mathbf{C}\boldsymbol{\varepsilon}$.

A.5. Details to Construct the Hamiltonian Matrix

The definition of the Hamiltonian matrix is given by Eq. (A.17) as the product of the Hamiltonian operator on basis functions. The operator is in turn defined by Eq. (A.11) and Eq. (A.10), following which the Hamiltonian matrix can be computed from the equations below:

$$\mathbf{H}_{\mathcal{M}}(\mathbf{C}) = \mathbf{T}_{\mathcal{M}} + \mathbf{V}_{\mathrm{H},\mathcal{M}}(\mathbf{C}) + \mathbf{V}_{\mathrm{XC},\mathcal{M}}(\mathbf{C}) + \mathbf{V}_{\mathrm{ext},\mathcal{M}}, \tag{A.18}$$

where:

$$\begin{aligned}
(\mathbf{T}_{\mathcal{M}})_{\alpha\beta} &:= \langle \eta_{\mathcal{M},\alpha} | \hat{T} | \eta_{\mathcal{M},\beta} \rangle = -\frac{1}{2}\int \eta_{\mathcal{M},\alpha}(\mathbf{r}) \nabla^2 \eta_{\mathcal{M},\beta}(\mathbf{r})\, \mathrm{d}\mathbf{r}, \\
(\mathbf{V}_{\mathrm{H},\mathcal{M}}(\mathbf{C}))_{\alpha\beta} &:= \langle \eta_{\mathcal{M},\alpha} | V_{\mathrm{H}[\rho_{\mathcal{M},\mathbf{C}}]} | \eta_{\mathcal{M},\beta} \rangle \overset{\text{Eqs. (A.10, A.15)}}{=} \sum_{\gamma\delta} (\tilde{\mathbf{D}}_{\mathcal{M}})_{\alpha\beta,\gamma\delta} (\mathbf{C}\mathbf{C}^\top)_{\gamma\delta}, \\
&\text{where } (\tilde{\mathbf{D}}_{\mathcal{M}})_{\alpha\beta,\gamma\delta} := \iint \frac{\eta_{\mathcal{M},\alpha}(\mathbf{r})\eta_{\mathcal{M},\beta}(\mathbf{r})\eta_{\mathcal{M},\gamma}(\mathbf{r}')\eta_{\mathcal{M},\delta}(\mathbf{r}')}{\|\mathbf{r}-\mathbf{r}'\|}\, \mathrm{d}\mathbf{r}'\mathrm{d}\mathbf{r}, \\
(\mathbf{V}_{\mathrm{XC},\mathcal{M}}(\mathbf{C}))_{\alpha\beta} &:= \langle \eta_{\mathcal{M},\alpha} | V_{\mathrm{XC}[\rho_{\mathcal{M},\mathbf{C}}]} | \eta_{\mathcal{M},\beta} \rangle = \int V_{\mathrm{XC}[\rho_{\mathcal{M},\mathbf{C}}]}(\mathbf{r}) \eta_{\mathcal{M},\alpha}(\mathbf{r}) \eta_{\mathcal{M},\beta}(\mathbf{r})\, \mathrm{d}\mathbf{r},
\end{aligned} \tag{A.19}$$

$$(\mathbf{V}_{\mathrm{ext},\mathcal{M}})_{\alpha\beta} := \langle \eta_{\mathcal{M},\alpha} | V_{\mathrm{ext},\mathcal{M}} | \eta_{\mathcal{M},\beta} \rangle = -\sum_{a=1}^{A} Z^{(a)} \int \frac{\eta_{\mathcal{M},\alpha}(\mathbf{r})\eta_{\mathcal{M},\beta}(\mathbf{r})}{\left\|\mathbf{r}-\mathbf{R}^{(a)}\right\|}\, \mathrm{d}\mathbf{r}. \tag{A.20}$$

Under the mentioned type of basis functions $\{\eta_{\mathcal{M},\alpha}(\mathbf{r})\}_{\alpha=1}^{B}$, integrals $\mathbf{S}_{\mathcal{M}}$, $\mathbf{T}_{\mathcal{M}}$, $\tilde{\mathbf{D}}_{\mathcal{M}}$, and $\mathbf{V}_{\mathrm{ext},\mathcal{M}}$ can be evaluated analytically. To evaluate $\mathbf{V}_{\mathrm{XC},\mathcal{M}}(\mathbf{C})$, the integral can be evaluated on a quadrature grid, for which common XC functional approximations provide a way to evaluate $V_{\mathrm{XC}[\rho_{\mathcal{M},\mathbf{C}}]}$ on each grid point.

Density Fitting Note that directly calculating $\mathbf{V}_{\mathrm{H},\mathcal{M}}(\mathbf{C})$ following Eq. (A.19) requires $O(B^4) = O(N^4)$ complexity, which soon dominates the cost and restricts the applicability to large systems. There is a widely adopted approach in DFT to reduce the complexity for this term, called *density fitting* (Dunlap, 2000). It is motivated by noting $V_{\mathrm{H}[\rho_{\mathcal{M},\mathbf{C}}]}(\mathbf{r})$ as defined in Eq. (A.10) involves an integral with density function $\rho_{\mathcal{M},\mathbf{C}}(\mathbf{r})$, which, by Eq. (A.15), involves a double summation that incurs $O(N^2)$ cost. Eq. (A.15) can be seen as expanding the density function onto the paired basis set $\{\eta_{\mathcal{M},\alpha}(\mathbf{r})\eta_{\mathcal{M},\beta}(\mathbf{r})\}_{\alpha,\beta=1,\cdots,B}$ of size B^2. It is hence possible to reduce the complexity by projecting the density function onto an *auxiliary basis set* $\{\omega_{\mathcal{M},\mu}(\mathbf{r})\}_{\mu=1}^{M}$ of size $M = O(N)$. The projected density can be represented by the corresponding coefficients $\mathbf{p}$ in the way that:

$$\rho_{\mathcal{M},\mathbf{p}}(\mathbf{r}) := \sum_{\mu=1}^{M} \mathbf{p}_\mu \omega_{\mathcal{M},\mu}(\mathbf{r}). \tag{A.21}$$

The projection is done by finding the coefficients $\mathbf{p}$ that minimizes the difference from $\rho_{\mathcal{M},\mathbf{p}}(\mathbf{r})$ to $\rho_{\mathcal{M},\mathbf{C}}(\mathbf{r})$. Note that the purpose of density fitting here is to reduce cost complexity for calculating $\mathbf{V}_{\mathrm{H},\mathcal{M}}$, the operator matrix corresponding to the Hartree energy defined in Eq. (A.4). Therefore, the difference is preferred to be measured in Hartree energy:

$$\begin{aligned}
E_{\mathrm{H}}[\rho_{\mathcal{M},\mathbf{p}} - \rho_{\mathcal{M},\mathbf{C}}] &= \iint \frac{\big(\rho_{\mathcal{M},\mathbf{p}}(\mathbf{r}) - \rho_{\mathcal{M},\mathbf{C}}(\mathbf{r})\big)\big(\rho_{\mathcal{M},\mathbf{p}}(\mathbf{r}') - \rho_{\mathcal{M},\mathbf{C}}(\mathbf{r}')\big)}{\|\mathbf{r}-\mathbf{r}'\|}\, \mathrm{d}\mathbf{r}\mathrm{d}\mathbf{r}' \\
&= \mathbf{p}^\top \tilde{\mathbf{W}}_{\mathcal{M}}\, \mathbf{p} - 2\mathbf{p}^\top \tilde{\mathbf{L}}_{\mathcal{M}}\, \mathrm{vec}(\mathbf{C}\mathbf{C}^\top) + \mathrm{vec}(\mathbf{C}\mathbf{C}^\top)^\top \tilde{\mathbf{D}}_{\mathcal{M}}\, \mathrm{vec}(\mathbf{C}\mathbf{C}^\top),
\end{aligned}$$

where $\mathrm{vec}(\mathbf{C}\mathbf{C}^\top) \in \mathbb{R}^{B^2}$ denotes the vector of the flattened density matrix $\mathbf{C}\mathbf{C}^\top \in \mathbb{R}^{B\times B}$, and the pre-computed constant integral matrices are defined by: $(\tilde{\mathbf{W}}_\mathcal{M})_{\mu\nu} := \iint \frac{\omega_{\mathcal{M},\mu}(\mathbf{r})\omega_{\mathcal{M},\nu}(\mathbf{r}')}{\|\mathbf{r}-\mathbf{r}'\|}\,\mathrm{d}\mathbf{r}\mathrm{d}\mathbf{r}'$, $(\tilde{\mathbf{L}}_\mathcal{M})_{\mu,\alpha\beta} := \iint \frac{\omega_{\mathcal{M},\mu}(\mathbf{r})\eta_{\mathcal{M},\alpha}(\mathbf{r}')\eta_{\mathcal{M},\beta}(\mathbf{r}')}{\|\mathbf{r}-\mathbf{r}'\|}\,\mathrm{d}\mathbf{r}\mathrm{d}\mathbf{r}'$, and $(\tilde{\mathbf{D}}_\mathcal{M})_{\alpha\beta,\gamma\delta} := \iint \frac{\eta_{\mathcal{M},\alpha}(\mathbf{r})\eta_{\mathcal{M},\beta}(\mathbf{r})\eta_{\mathcal{M},\gamma}(\mathbf{r}')\eta_{\mathcal{M},\delta}(\mathbf{r}')}{\|\mathbf{r}-\mathbf{r}'\|}\,\mathrm{d}\mathbf{r}\mathrm{d}\mathbf{r}'$, which can be computed analytically using common basis sets. As a quadratic form, the solution is:

$$\mathbf{p}_\mathcal{M}(\mathbf{C}) := \tilde{\mathbf{W}}_\mathcal{M}^{-1}\tilde{\mathbf{L}}_\mathcal{M}\,\mathrm{vec}(\mathbf{C}\mathbf{C}^\top). \tag{A.22}$$

Note that since the auxiliary basis is usually not complete to expand the paired basis, the projected density $\rho_{\mathcal{M},\mathbf{p}_\mathcal{M}(\mathbf{C})}$ is an approximation to the original density $\rho_{\mathcal{M},\mathbf{C}}$.

Using density fitting, the Hartree operator matrix $\mathbf{V}_{\mathrm{H},\mathcal{M}}(\mathbf{C})$ defined by Eq. (A.19) can be approximately estimated by substituting the projected density $\rho_{\mathcal{M},\mathbf{p}_\mathcal{M}(\mathbf{C})}(\mathbf{r})$, given by Eq. (A.21) and Eq. (A.22), into the Hartree potential $V_{\mathrm{H}[\rho_{\mathcal{M},\mathbf{p}_\mathcal{M}(\mathbf{C})}]}$ defined by Eq. (A.10):

$$(\mathbf{V}_{\mathrm{H},\mathcal{M}}(\mathbf{C}))_{\alpha\beta} \approx \big(\mathbf{p}_\mathcal{M}(\mathbf{C})^\top\tilde{\mathbf{L}}_\mathcal{M}\big)_{\alpha\beta}. \tag{A.23}$$

Since the calculation of $\mathbf{p}_\mathcal{M}(\mathbf{C})$ following Eq. (A.22) has complexity $O(M^3)+O(MB^2)+O(NB^2)=O(N^3)$, and the complexity of Eq. (A.23) itself has complexity $O(MB^2)=O(N^3)$, the overall complexity for estimating $\mathbf{V}_{\mathrm{H},\mathcal{M}}(\mathbf{C})$ using density fitting is $O(N^3)$, which reduces the original quartic $O(N^4)$ complexity.

Alternative Derivation We would like to mention an alternative derivation of the Hamiltonian matrix as an amendment. This derivation is to first parameterize the optimization problem using a function basis then deriving the optimality condition in matrix form. Noting that under a basis set $\{\eta_{\mathcal{M},\alpha}(\mathbf{r})\}_{\alpha=1}^B$, the orbital functions can be parameterized using the orbital coefficient matrix $\mathbf{C}$ as $\Phi_{\mathcal{M},\mathbf{C}}$ in the form of Eq. (A.14), the corresponding optimization problem Eq. (A.9) can be converted into a usual optimization problem on vectors/matrix (instead of on functions): $E^\star_\mathcal{M} =$

$$\min_{\substack{\mathbf{C}\in\mathbb{R}^{B\times N}:\\ \mathbf{C}^\top\mathbf{S}_\mathcal{M}\mathbf{C}=\mathbf{I}}}\left\{E_\mathcal{M}(\mathbf{C}) := E_\mathcal{M}[\Phi_{\mathcal{M},\mathbf{C}}] = \begin{pmatrix}\mathrm{vec}(\mathbf{T}_\mathcal{M})^\top\mathrm{vec}(\mathbf{\Gamma}(\mathbf{C})) + \frac{1}{2}\mathrm{vec}(\mathbf{\Gamma}(\mathbf{C}))^\top\tilde{\mathbf{D}}_\mathcal{M}\mathrm{vec}(\mathbf{\Gamma}(\mathbf{C}))\\ +E_{\mathrm{XC}}\big[\sum_{\alpha,\beta}\mathbf{\Gamma}(\mathbf{C})_{\alpha\beta}\eta_{\mathcal{M},\alpha}\eta_{\mathcal{M},\beta}\big] + \mathrm{vec}(\mathbf{V}_{\mathrm{ext},\mathcal{M}})^\top\mathrm{vec}(\mathbf{\Gamma}(\mathbf{C}))\end{pmatrix} =: E_\mathcal{M}\big(\mathbf{\Gamma}(\mathbf{C})\big)\right\}, \tag{A.24}$$

where the constraint comes from the orthonormality of orbitals $\delta_{ij} = \langle\phi_{\mathcal{M},\mathbf{C},i}|\phi_{\mathcal{M},\mathbf{C},j}\rangle = \sum_{\alpha,\beta}\mathbf{C}_{\alpha i}\mathbf{C}_{\beta j}\langle\eta_{\mathcal{M},\alpha}|\eta_{\mathcal{M},\beta}\rangle = \sum_{\alpha,\beta}\mathbf{C}_{\alpha i}\mathbf{C}_{\beta j}(\mathbf{S}_\mathcal{M})_{\alpha\beta} = (\mathbf{C}^\top\mathbf{S}_\mathcal{M}\mathbf{C})_{ij}$, and the density matrix is defined by $\mathbf{\Gamma}(\mathbf{C}) := \mathbf{C}\mathbf{C}^\top$. The expression for the XC energy part comes from the density function expression under a basis, *i.e.*, Eq. (A.15). Noting that the energy expression depends on $\mathbf{C}$ only through the density matrix $\mathbf{\Gamma}(\mathbf{C})$, we finally denote the optimization objective as $E_\mathcal{M}\big(\mathbf{\Gamma}(\mathbf{C})\big)$. Introducing Lagrange multipliers grouped into a symmetric matrix $\boldsymbol{\epsilon}$ (since the constraint is symmetric) for the constraint and taking the gradient w.r.t $\mathbf{C}$, we have the optimality condition: $\nabla_\mathbf{C}E_\mathcal{M}\big(\mathbf{\Gamma}(\mathbf{C})\big) = \nabla_\mathbf{C}\,\mathrm{tr}\left(\boldsymbol{\epsilon}^\top(\mathbf{C}^\top\mathbf{S}_\mathcal{M}\mathbf{C}-\mathbf{I})\right)$. Using the chain rule and that all matrices except $\mathbf{C}$ are symmetric, we have:

$$\nabla_\mathbf{C}E_\mathcal{M}\big(\mathbf{\Gamma}(\mathbf{C})\big) = 2\nabla_\mathbf{\Gamma}E_\mathcal{M}\big(\mathbf{\Gamma}(\mathbf{C})\big)\,\mathbf{C}, \tag{A.25}$$

and that $\nabla_\mathbf{C}\,\mathrm{tr}\left(\boldsymbol{\epsilon}^\top(\mathbf{C}^\top\mathbf{S}_\mathcal{M}\mathbf{C}-\mathbf{I})\right) = 2\mathbf{S}_\mathcal{M}\mathbf{C}\boldsymbol{\epsilon}$. The optimality equation then becomes:

$$\nabla_\mathbf{\Gamma}E_\mathcal{M}\big(\mathbf{\Gamma}(\mathbf{C})\big)\,\mathbf{C} = \mathbf{S}_\mathcal{M}\mathbf{C}\boldsymbol{\epsilon}. \tag{A.26}$$

When optimality is achieved, $\mathbf{C}$ is the eigenvectors of the Hermitian (symmetric) matrix $\nabla_\mathbf{\Gamma}E_\mathcal{M}\big(\mathbf{\Gamma}(\mathbf{C})\big)$, so in the common situation that there is no degenerated state, the eigenvectors are already orthogonal, *i.e.*, the non-diagonal part of the constraint $\mathbf{C}^\top\mathbf{S}_\mathcal{M}\mathbf{C} = \mathbf{I}$ is satisfied. Therefore, the multipliers only need to handle the normalization constraints hence only the diagonal part of $\boldsymbol{\epsilon}$ is effective. This reduces $\boldsymbol{\epsilon}$ in Eq. (A.26) to a diagonal matrix. In this way, Eq. (A.26) becomes identical to Eq. (A.16), which indicates:

$$\mathbf{H}_\mathcal{M}(\mathbf{C}) = \nabla_\mathbf{\Gamma}E_\mathcal{M}\big(\mathbf{\Gamma}(\mathbf{C})\big). \tag{A.27}$$

The equivalence to the first definition in Eq. (A.17) together with Eq. (A.11) and Eq. (A.10) can be seen from the relation between variation and gradient: for a general functional $F[\cdot]$ and a general parameterized function $f_\theta(x)$, the relation is:

$$\frac{\partial F[f_\theta]}{\partial\theta} = \int\frac{\delta F[f_\theta]}{\delta f}(x)\frac{\partial f_\theta}{\partial\theta}(x)\,\mathrm{d}x.$$

Using this equation and noting that $(\nabla_{\mathbf{C}} E)_{\alpha i}$ means $\frac{\partial E}{\partial \mathbf{C}_{\alpha i}}$ and $E_{\mathcal{M}}(\mathbf{C}) := E_{\mathcal{M}}[\Phi_{\mathcal{M},\mathbf{C}}]$ from Eq. (A.24), we have:

$$\begin{aligned}
\big(\nabla_{\mathbf{C}} E_{\mathcal{M}}(\mathbf{C})\big)_{\alpha i} &= \int \sum_{j=1}^{N} \frac{\delta E_{\mathcal{M}}[\Phi_{\mathcal{M},\mathbf{C}}]}{\delta \phi_{\mathcal{M},\mathbf{C},j}}(\mathbf{r}) \big(\nabla_{\mathbf{C}} \phi_{\mathcal{M},\mathbf{C},j}(\mathbf{r})\big)_{\alpha i} \,\mathrm{d}\mathbf{r} = \int \frac{\delta E_{\mathcal{M}}[\Phi_{\mathcal{M},\mathbf{C}}]}{\delta \phi_{\mathcal{M},\mathbf{C},i}}(\mathbf{r}) \big(\nabla_{\mathbf{C}} \phi_{\mathcal{M},\mathbf{C},i}(\mathbf{r})\big)_{\alpha i} \,\mathrm{d}\mathbf{r} \\
&\overset{\text{Eqs. (A.12, A.14)}}{=} 2 \int \hat{H}_{\mathcal{M},[\rho_{\mathcal{M},\mathbf{C}}]} \phi_{\mathcal{M},\mathbf{C},i}(\mathbf{r})\, \eta_{\mathcal{M},\alpha}(\mathbf{r}) \,\mathrm{d}\mathbf{r} = 2 \int \sum_{\beta} \mathbf{C}_{\beta i} \hat{H}_{\mathcal{M},[\rho_{\mathcal{M},\mathbf{C}}]} \eta_{\mathcal{M},\beta}(\mathbf{r})\, \eta_{\mathcal{M},\alpha}(\mathbf{r}) \,\mathrm{d}\mathbf{r} \\
&= 2 \sum\nolimits_{\beta} \mathbf{C}_{\beta i} \langle \eta_{\mathcal{M},\alpha} | \hat{H}_{\mathcal{M},[\rho_{\mathcal{M},\mathbf{C}}]} | \eta_{\mathcal{M},\beta} \rangle \overset{\text{Eqs. (A.17)}}{=} 2\big(\mathbf{H}_{\mathcal{M}}(\mathbf{C})\, \mathbf{C}\big)_{\alpha i},
\end{aligned}$$

which means $\nabla_{\mathbf{C}} E_{\mathcal{M}}(\mathbf{C}) = 2\mathbf{H}_{\mathcal{M}}(\mathbf{C})\, \mathbf{C}$. On the other hand, noting $E_{\mathcal{M}}(\mathbf{C}) = E_{\mathcal{M}}\big(\mathbf{\Gamma}(\mathbf{C})\big)$ from Eq. (A.24) and noting Eq. (A.25), we also have $\nabla_{\mathbf{C}} E_{\mathcal{M}}(\mathbf{C}) = 2\nabla_{\mathbf{\Gamma}} E_{\mathcal{M}}\big(\mathbf{\Gamma}(\mathbf{C})\big)\, \mathbf{C}$. This also gives $\mathbf{H}_{\mathcal{M}}(\mathbf{C}) = \nabla_{\mathbf{\Gamma}} E_{\mathcal{M}}\big(\mathbf{\Gamma}(\mathbf{C})\big)$, *i.e.*, Eq. (A.27). From Eq. (A.27), the detailed construction of the Hamiltonian matrix Eq. (A.18) to (A.20) can be recovered using the detailed expressions in Eq. (A.24).

B. Additional Technical Details

In this section, we present additional details regarding the implementation of the Hamiltonian prediction model and the self-consistency loss.

B.1. Model Implementation Details

QHNet. We build our model upon the official QHNet codebase[4], which is an $\mathrm{SE}(3)$-equiavariant graph neural network for Hamiltonian prediction (Yu et al., 2023b). With careful architecture design, QHNet achieves a good balance between inference efficiency and accuracy. Its architecture is composed of four key modules: node-wise interaction, diagonal pair, non-diagonal pair and expansion. Given the atom types Z and positions $\mathcal{R}$ as inputs, the QHNet model employs five layers of node-wise interaction to extract $\mathrm{SE}(3)$-equivariant atomic features. Subsequently, the features of diagonal/non-diagonal atom pairs are fed into diagonal/non-diagonal pair modules respectively to build pairwise representations $\mathbf{f}_{aa}$ (diagonal) and $\mathbf{f}_{ab}$ (non-diagonal), where a and b denote the atom index. The expansion module then transforms these pairwise representations into blocks of the Hamiltonian matrix. Further information can be found in the original paper (Yu et al., 2023b). For our experimental studies, all models are configured with the default parameters specified for QHNet. The neural network codebase is developed using PyTorch(Paszke et al., 2019) and PyTorch-Geometric (Fey & Lenssen, 2019).

Adapter Module. As outlined in Sec. 3.1, to facilitate the generalization of the Hamiltonian model in the OOD scenario, we apply self-consistency loss for fine-tuning the QHNet model with two fine-tuning approaches: `all-param` and `adapter`. Specifically, we construct the `adapter` using three modules: diagonal pair module, non-diagonal pair module and expansion module. and then insert it atop the original QHNet model. A schematic illustration of the adapter module is provided in Fig. B.1. Given the input molecule, the pretrained QHNet model is initially used to produce the initial Hamiltonian matrix blocks $\hat{\mathbf{H}}$[5], along with the final atomic representations $\mathbf{h}$ and the final pairwise representations $\mathbf{f}$. Subsequently, the atomic representations are fed into corresponding diagonal or non-diagonal pair modules respectively to build pairwise representations $\mathbf{f}^{'}$. Afterward, the pairwise representations of the QHNet model and the adapter module are combined (*e.g.*, $\mathbf{f}^{'}_{aa} + t_1 \cdot \mathbf{f}_{aa}$, with t_1 as a learnable combination coefficient). The combined pairwise representations are first fed into a linear layer and then employed by the expansion module to produce the refinement Hamiltonian ($\hat{\mathbf{H}}^{'}$). Finally, we take the combination of the initial Hamiltonian and the refinement Hamiltonian (*e.g.*, $\hat{\mathbf{H}}^{'}_{aa} + o_1 \cdot \hat{\mathbf{H}}_{aa}$, with o_1 as a learnable combination coefficient) as the final output $\hat{\mathbf{H}}^{''}$. It is important to note that the combination of pairwise representations and Hamiltonian blocks, whether diagonal or non-diagonal, is conducted independently, and the combination coefficients are distinct for each pair (*i.e.*, $t_1 \neq t_2$ and $o_1 \neq o_2$).

B.2. Self-Consistency Loss

Back-Propagation through Eigensolver. As described in Sec. 2.3, the evaluation of self-consistency loss $\mathcal{L}_{\text{self-con}}$ (Eq. (3)) requires the eigenvectors $\mathbf{C}_{\mathcal{M},\theta}$ of the generalized eigenvalue problem (Line 3 in Alg. 1), necessitating the back-propagation

[4] https://github.com/divelab/AIRS/tree/main/OpenDFT/QHNet, the code is available under the terms of the GPL-3.0 license

[5] The model-predicted Hamiltonian should be formally denoted as $\hat{\mathbf{H}}_{\theta}(\mathcal{M})$, and we omit θ and $\mathcal{M}$ for brevity

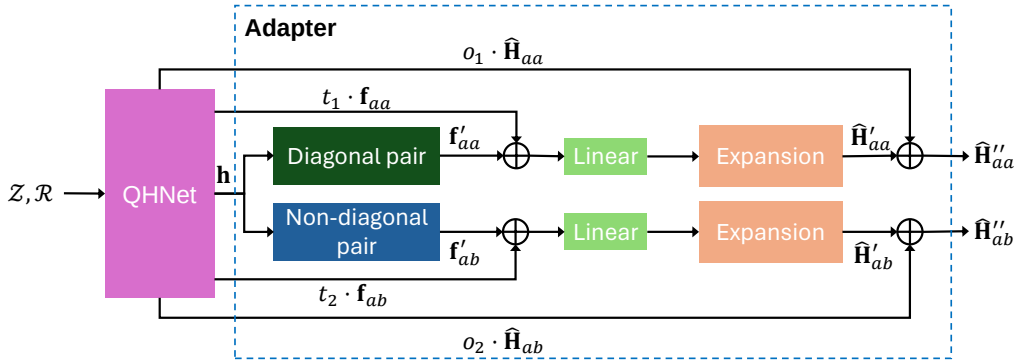


Figure B.1. The whole architecture of the adapter module. Given the atom types $\mathcal{Z}$ and positions $\mathcal{R}$ as inputs, the pretrained QHNet model is used to produce atomic representations $\mathbf{h}$, pairwise representations $\mathbf{f}$ and the initial Hamiltonian prediction $\hat{\mathbf{H}}$. Subsequently, the adapter module is utilized to produce refinement Hamiltonian $\hat{\mathbf{H}}'$ based on $\mathbf{h}$ and $\mathbf{f}$. Finally, the refinement Hamiltonian is combined with the initial Hamiltonian prediction as the final output $\hat{\mathbf{H}}''$. t_1, t_2, o_1 and o_2 denote learnable combination coefficients. a and b denote the indexes of atoms.

through an eigensolver. Thus we need to compute the gradient of the loss function $\mathcal{L}_{\text{self-con}}$ w.r.t the matrix $\hat{\mathbf{H}}_\theta(\mathcal{M})$. In our practical implementation, we solve the generalized eigenvalue problem for each molecule with two steps: **(1)** Solve the eigenvalue problem for matrix $\mathbf{S}_\mathcal{M}$, $\mathbf{\Lambda}, \mathbf{U} = \texttt{EigSol}(\mathbf{S}_\mathcal{M})$ and then define $\mathbf{A} = \mathbf{U}\mathbf{\Lambda}^{-1/2}$. This leads to the transformation of the Hamiltonian matrix to $\tilde{\mathbf{H}}_\theta = \mathbf{A}^\top \hat{\mathbf{H}}_\theta(\mathcal{M})\mathbf{A}$; **(2)** Solve the eigenvalue problem for the transformed Hamiltonian matrix $\tilde{\mathbf{H}}_\theta$, $\boldsymbol{\epsilon}, \tilde{\mathbf{C}} = \texttt{EigSol}(\tilde{\mathbf{H}}_\theta)$, from which the eigenvectors of the original problem are recovered as $\mathbf{C}_{\mathcal{M},\theta} = \mathbf{A}\tilde{\mathbf{C}}$. Following these steps, the self-consistency loss $\mathcal{L}_{\text{self-con}}(\mathbf{H}_\mathcal{M}(\mathbf{C}_{\mathcal{M},\theta}))$ is calculated using the eigenvectors $\mathbf{C}_{\mathcal{M},\theta}$ that have been derived.

The partial derivatives of self-consistency loss $\mathcal{L}_{\text{self-con}}$ w.r.t the transformed matrix $\tilde{\mathbf{H}}_\theta$ can be expressed as: $\nabla_{\tilde{\mathbf{H}}_\theta}\mathcal{L}_{\text{self-con}} = \tilde{\mathbf{C}}\big(\mathbf{G} \circ (\tilde{\mathbf{C}}^\top \nabla_{\tilde{\mathbf{C}}}\mathcal{L}_{\text{self-con}})\big)\tilde{\mathbf{C}}^\top$, where

$$\mathbf{G}_{ij} = \begin{cases} 1/(\epsilon_i - \epsilon_j), & i \neq j, \\ 0, & i = j, \end{cases}$$

with ϵ_i representing the i-th eigenvalues. The gradient $\nabla_{\tilde{\mathbf{C}}}\mathcal{L}_{\text{self-con}}$ is calculated using the chain rule $\nabla_{\tilde{\mathbf{C}}}\mathcal{L}_{\text{self-con}} = \nabla_{\tilde{\mathbf{C}}}\mathbf{C}\,\nabla_{\mathbf{C}_{\mathcal{M},\theta}}\mathcal{L}_{\text{self-con}} = \text{vec}^{-1}\big((\mathbf{I}_N \otimes \mathbf{A}^\top)\,\text{vec}(\nabla_{\mathbf{C}_{\mathcal{M},\theta}}\mathcal{L}_{\text{self-con}})\big)$, where vec and vec^{-1} denote the vectorization operator and its inverse operator, $\otimes$ denotes the Kronecker product operator, and $\mathbf{I}_N$ denotes the N-dimensional identity matrix. Then we can derive the partial derivative w.r.t the original Hamiltonian matrix $\hat{\mathbf{H}}_\theta(\mathcal{M})$ as: $\nabla_{\hat{\mathbf{H}}_\theta(\mathcal{M})}\mathcal{L}_{\text{self-con}} = \nabla_{\tilde{\mathbf{H}}_\theta}\hat{\mathbf{H}}_\theta(\mathcal{M})\,\nabla_{\tilde{\mathbf{H}}_\theta}\mathcal{L}_{\text{self-con}} = \text{vec}^{-1}\big(\mathbf{A} \otimes \mathbf{A}\,\text{vec}(\nabla_{\tilde{\mathbf{H}}_\theta}\mathcal{L}_{\text{self-con}})\big)$. Consequently, the partial derivatives w.r.t $\hat{\mathbf{H}}_\theta(\mathcal{M})$ rely on the matrix $\mathbf{G}$, which can lead to a large gradient when two eigenvalues are close.

To mitigate this instability and promote stable training, we introduce two treatments. The first is to limit the magnitude of gradient by applying truncation on matrix $\mathbf{G}$ in the `backward` function of PyTorch:

$$\tilde{\mathbf{G}}_{ij} = \begin{cases} T \cdot \text{sgn}(\epsilon_i - \epsilon_j), & \text{if } 1/|\epsilon_i - \epsilon_j| > T, \\ \mathbf{G}_{ij}, & \text{if } 1/|\epsilon_i - \epsilon_j| \leq T, \end{cases}$$

where T is a threshold determined by taking the 60-th percentile of absolute values of all $\mathbf{G}$ entries, and $\text{sgn}(\cdot)$ denotes the sign function. The technique is chosen for its simplicity and effectiveness, and there exist other methods for addressing this issue (Song et al., 2021; Wang et al., 2021a). The second treatment is to skip model parameter update when the scale of the gradient w.r.t parameters exceeds a certain threshold g_s, which is determined through cross-validation.

Efficient Hamiltonian Reconstruction To reconstruct the Hamiltonian $\mathbf{H}_\mathcal{M}(\mathbf{C}_{\mathcal{M},\theta})$, we first generate requisite integrals and quadrature grid using PySCF and then compute the Hamiltonian using PyTorch according to standard SCF procedure. Yet, this involves two costly steps. **(1)** Evaluating atomic basis functions on generated quadrature grid points is computationally

Table B.1. Comparison of training time per iteration for the QHNet model with and without the incorporation of the self-consistency loss. The batch size is maintained at 5 across all configurations, and the average training time is calculated over 50 iterations. Unit: s.

Model	Ethanol	Malondialdehyde	Uracil
QHNet *without* self-con	0.283	0.289	0.355
QHNet *with* self-con	0.375	0.401	0.613

expensive. To accelerate this computation, we re-implement the evaluation of basis functions on GPU. Moreover, the grid level determines the number of grid points and in turn influences the construction accuracy of the exchange-correlation potential $\mathbf{V}_{\mathrm{XC},\mathcal{M}}(\mathbf{C})$. Empirically, we find that a grid level of 2 strikes an optimal balance between construction accuracy and computational efficiency. **(2)** The computation of the Hartree component entails a $O(N^4)$ complexity (Line 6 in Alg. 1, N is the number of electrons). As the molecular size increases, this computation becomes highly costly. To enable an efficient evaluation of the Hartree matrix, we apply the density fitting technique widely used in DFT programs to reduce the computational complexity from $O(N^4)$ to $O(N^3)$. Leveraging the two techniques leads to a significant acceleration for the Hamiltonian reconstruction, enabling faster self-consistency training. Moreover, integral matrices that are solely dependent on the molecular conformation (*i.e.*, $\mathbf{T}_{\mathcal{M}}$, $\mathbf{V}_{\mathrm{ext},\mathcal{M}}$) are pre-computed and stored in the database. These pre-computed matrices are then loaded as needed during the training process.

Computational Complexity. As the self-consistency loss is constructed following the standard SCF procedure, it possesses the same computational complexity as one SCF iteration under the Kohn-Sham DFT formulation. After the application of the density fitting technique, the computational complexity becomes $O(N^3)$ (N denotes the number of electrons in a molecule). Note that self-consistency training only brings extra computational cost during training, while keeps the same cost for Hamiltonian prediction. The empirical time cost of the Hamiltonian prediction model, both with and without the incorporation of self-consistency loss, are detailed in Table B.1.

C. Experimental Study Settings

In this section, we provide further data preparation and training details for the empirical study presented in Sec. 3.

C.1. Dataset Preparation

To demonstrate the benefits of self-consistent training, we first conduct experiments on two generalization scenarios, corresponding to two molecular datasets, MD17 and QH9, respectively. Afterward, we evaluate the applicability of the Hamiltonian prediction model on large-scale molecules, for which we adopt the MD22 dataset.

Table C.1. Statistics of the MD17 dataset (Schütt et al., 2019).

Molecule	Train (labeled)	Train (unlabeled)	Validation	Test	Molecular size
Ethanol	100	24,900	500	4,500	9
Malondialdehyde	100	24,900	500	1,478	19
Uracil	100	24,900	500	4,500	26

MD17. To evaluate the benefit of self-consistency training in improving generalization for the data-scarce scenario, we adopt the MD17 dataset (Schütt et al., 2019), and focus on three conformational spaces of ethanol ($\mathrm{C_2H_5OH}$), malondialdehyde ($\mathrm{CH_2(CHO)_2}$) and uracil ($\mathrm{C_4H_4N_2O_2}$). The Hamiltonian matrices in this dataset are calculated with the PBE (Perdew et al., 1996) exchange-correlation functional and the Def2SVP Gaussian-type orbital (GTO) basis set. We follow the split setting used by Schütt et al. (2019) to divide the structures of each molecule into training/validation/test sets. Moreover, we randomly select 100 labels from the training set for supervised training, while the remaining training structures are utilized as unlabeled data for self-consistency training. The detailed statistics of three conformational spaces are summarized in Table C.1.

Table C.2. Statistics of the QH9 dataset (Yu et al., 2023a).

Data setting	Training	Validation	Test
QH9-small	94,001	10,000	N/A
QH9-large	18,000	2,000	6,830
QH9-full	124,289	6,542	N/A

QH9. To evaluate the benefit of self-consistency training in improving generaliation for the out-of-distribution (OOD) scenario, we adopt the QH9 dataset[6] (Yu et al., 2023a). This dataset is proposed to benchmark Hamiltonian prediction methods in chemical space, consisting of two subsets: QH-stable and QH-dynamic. Here we adopt the QH-stable subset (dubbed as QH9 hereafter), which consists of 130,831 stable small organic molecules with no more than 9 heavy atoms, as well as their corresponding Hamiltonian matrices. The Hamiltonian matrices are calculated with the B3LYP (Becke, 1992) exchange-correlation functional and the Def2SVP GTO basis set. To simulate an OOD benchmark, we divide the QH9 dataset into two subsets by molecular size (QH9-small and QH9-large) and further partition them into training/validation/test sets. Additionally, we establish a separate split setting for the generalization study on large-scale molecules(referred to as QH9-full). Comprehensive statistics related to these division settings are detailed in Table C.2.

MD22. To evaluate the applicability of the Hamiltonian prediction model on large-scale molecules, we adopt the MD22 dataset (Chmiela et al., 2023) and focus on the Ac-Ala3-NHMe (ALA3) and DHA molecules. Since the MD22 does not provide Hamiltonian labels (and energy and force labels are provided under a different exchange-correlation functional PBE), we randomly sample 500 structures for each molecule as our benchmark and use PySCF (Sun et al., 2018) to generate Hamiltonian matrices for these structures with the B3LYP exchange-correlation functional and the Def2SVP GTO basis set.

C.2. DFT Implementation Details

For this study, all DFT calculations, including those for evaluating the SCF acceleration ratio (Sec. 3.1-3.3) and for benchmarking the DFT computation cost (Sec. 3.2) are performed using the PySCF software (Sun et al., 2018) with its default parameter settings.

C.3. Hardware Configurations

All neural network models are trained and evaluated on a workstation equipped with a Nvidia A100 GPU with 80 GiB memory and a 24-core AMD EPYC CPU, which is also used for DFT calculations. Note that all the computation times reported in the empirical study are benchmarked on this specific hardware configuration to ensure consistency in comparison. However, it is also recognized that both neural network models and DFT computations have the potential to be parallelized and accelerated using multiple GPUs or CPU cores. Given this capability for parallel processing, establishing a perfectly equitable hardware benchmarking environment for both approaches is challenging.

C.4. Training Details

Data-Scarce Scenario. We first describe the training details utilized in the empirical study of Sec. 3.1. For the self-consistency training setting, we set the total training iterations to 200k for three conformational spaces following Yu et al. (2023b). The weighting factor $\lambda_{\text{self-con}}$ is set to 10 across all molecules. Considering that the efficacy of the supervised learning setting might be limited by scarce labeled data, we allocate a higher number of training iterations (*i.e.*, 500k) to this strategy to ensure the model reaches its optimal performance within the constraints of the available labels. For all experimental conditions and datasets, we maintain a consistent batch size of 5. We utilize a polynomial decay learning rate scheduler to modulate the learning rate (LR) during training, where the polynomial power is set to 5 for self-consistency training and 14 for supervised learning based on empirical trials. Notably, the scheduler increases the learning rate gradually during the first 10k warm-up iterations. The learning rate starts at 0 and peaks at a maximum of 1×10^{-3} across all training scenarios. When addressing the supervised training settings with extended labeled data as mentioned in Sec. 3.2, we adopt the same training hyperparameters as those used in the supervised learning setting of Sec. 3.1, with the singular adjustment of setting the polynomial power to 5.

[6]The dataset is licensed under a Creative Commons Attribution-NonCommercial-ShareAlike 4.0 International License.

Out-of-Distribution Scenario. As outlined in Sec. 3.1, to benchmark the OOD generalization performance, we initially train the QHNet model on the QH9-small subset, and then fine-tune the model on unlabeled large molecules. We continue to use the polynomial learning rate schedule, which includes a warm-up phase. Additional training hyperparameters are detailed in Table C.3.

Table C.3. Training hyper-parameters in the OOD scenario.

Training Phase	Batch size	Maximum LR	Polynomial Power	Iterations	Warm-up Iterations
Pretraining	32	5×10^{-4}	1	300k	1k
Fine-tuning	5	2×10^{-5}	8	200k	3k

Large-Scale Molecular Systems. As discussed in Sec. 3.3, we adopt a two-stage training strategy to generalize the Hamiltonian model to large-scale MD22 structures. We still employ the polynomial learning rate schedule with the warm-up stage, and detail other training hyper-parameters in Table C.4.

D. Additional Experimental Results

D.1. Generalization Results

As noted in Sec. 3.1, while supervised training can outperform self-consistency training with adequate computational resources, the advantage in terms of Hamiltonian MAE does not consistently extend to molecular properties. This discrepancy has been observed in the OOD generalization scenario in Table 3 of Sec. 3.1. Correspondingly, the results presented in Table C.5 show that there exists a comparable trend in the data-scarce scenario.

D.2. Results of Alternative Model Architectures

For further validate the advantage of self-consistency training with alternative architectures, we investigate its benefit using the PhiSNet (Unke et al., 2021) architecture, which is another performant model for Hamiltonian prediction on molecules. As shown in Table D.1, the results exhibit the same conclusion as shown in Table 1: compared to the results of supervised training (`label`), applying self-consistency loss (`label+self-con`) on unlabeled structures leads to a remarkable improvement across all evaluation metrics. Notably, the MAEs for HOMO ϵ_{HOMO}, LUMO ϵ_{LUMO} and HOMO-LUMO gap ϵ_{Δ} are reduced by several folds. These results demonstrate the generality of self-consistency training for improving the performance of general architectures.

D.3. The Impact of SCF Iteration Strategies

As mentioned in Sec. 3, to illustrate the accuracy of Hamiltonian prediction, we assess its capability for accelerating DFT when using the prediction as initialization. Following previous studies (Yu et al., 2023b;a), all DFT calculations are carried out using the PySCF (Sun et al., 2018) software, with the PBE XC functional and the Def2SVP basis set being adopted. The direct inversion in iterative subspace (DIIS) (Pulay, 1980) iteration strategy is employed, following the defaults. The results in Tables 1-3 and 5 show that the Hamiltonian prediction model leads to a substantial SCF acceleration across various molecular systems, and applying self-consistency training can further improve the acceleration gains. Nevertheless, we find that the iteration strategy can considerably influence SCF convergence, which may diminish the benefit of a more accurate initialization. For example, it is known that DIIS may show a non-monotone iteration behavior, meaning that the Hamiltonian in the next iteration may not be closer to the final solution than the Hamiltonian in the current iteration (Sun, 2016; Sun et al., 2017). Hence, even when the initial Hamiltonian (predicted by the model) is closer to the final solution, the Hamiltonian in the next iteration may still be farther away from the solution ("DIIS algorithm does not honor the initial guess well. The optimization procedure may lead the wavefunction anywhere in the variational space" (Sun, 2016)). To verify this point, we attempt to use the second-order SCF (SOSCF) iteration strategy (Sun et al., 2017) in place of DIIS for running the SCF iteration and summarize the results in Table D.2. SOSCF directly engages in orbital optimization, hence guaranteeing monotonicity. In the evaluation setting of OOD generalization on QH9-large test molecules (in parallel with Table 3), we observe a 57.8% and 56.2% SCF acceleration for the `extended-label` and `self-con` settings respectively. The speedup is indeed improved to the DIIS speedup of 65.0% and 64.5% for the respective settings, justifying the speculation. Moreover, in the evaluation setting of large-scale generalization on ALA3 and DHA structures from MD22 (in parallel with

Table C.4. Training hyper-parameters in the large-scale molecular systems.

Training Phase	Batch size	Maximum LR	Polynomial Power	Iterations	Warm-up Iterations
Pretraining	32	5×10^{-4}	1	400k	5k
Fine-tuning (ALA3)	2	2×10^{-5}	8	100k	10k
Fine-tuning (DHA)	1	2×10^{-5}	8	200k	10k

Table C.5. Performance comparison between `self-con`sistency training, and supervised training using *full extended labels*, in the *data-scarce* scenario (in parallel with Table 3), corresponding to the ending points of Fig. 3 (`extended-label-online` is close to `extended-label`).

Molecule	Setting	$\mathbf{H}$ [μE_h] ↓	ϵ [μE_h] ↓	$\mathbf{C}$ [%] ↑	ϵ_HOMO [μE_h] ↓	ϵ_LUMO [μE_h] ↓	ϵ_Δ [μE_h] ↓	SCF Accel. [%] ↓
Ethanol	extended-label	**58.28**	986.84	99.94	**230.20**	2902.14	2723.64	63.5
	label + self-con	95.90	**340.56**	**99.94**	403.60	**1426.20**	**1370.35**	**61.5**
Malondi-aldehyde	extended-label	**71.45**	1014.12	99.63	**199.48**	414.58	415.91	66.6
	label + self-con	86.60	**280.39**	**99.67**	274.45	**279.14**	**324.37**	**62.1**
Uracil	extended-label	**52.53**	**288.29**	99.38	**306.05**	**294.54**	398.08	58.1
	label + self-con	63.82	315.40	**99.58**	359.98	369.67	**388.30**	**54.5**

Table 5), self-consistency training achieves a 47.5% and 37.0% SCF acceleration respectively, substantially better than DIIS speedup of 64.7% and 67.0%. These results support that employing the SOSCF convergence method can better honor the quality of the initial guess. Additionally, for DHA structures where the low-quality `zero-shot` prediction results in a deceleration (170.8%), SOSCF can lead to worse convergence performance (231.8%). Notably, the observed speedup appears to be more pronounced on the larger molecular systems (*e.g.*, ALA3 and DHA), which indicates that molecular systems listed in Table 3 may be already easy to converge with DFT and thereby difficult to accelerate further.

D.4. Amortization Effect of Self-Consistency Training

As mentioned in Sec. 3.2, we directly access the amortization effect by comparing the computational cost of self-consistency training with that of DFT for solving a bunch of structures. It should be noted that measuring the computational cost of DFT on all unlabeled training structures is impractical, thus we run DFT on 50 randomly picked structures for each molecule. The mean computation time derived from these 50 structures serves as a benchmark to approximate the overall computational time required for the complete set of structures.

To further demonstrate the amortization efficiency of self-consistency training, we also measure the computational cost by "the number of consumed SCF iterations" and present the accuracy-cost curves in Figs. D.1 and D.2. The results indicate the same conclusion as shown in Figs. 3 and 4: self-consistency training can achieve a satisfying prediction accuracy even with less SCF steps than the number of SCF steps in the DFT calculation for labeling the molecular structures. Even in the 'extended-label-online' setting where the data is generated along with the training of the model, self-consistency training can still achieve a better accuracy given the same budget of SCF iterations.

D.5. More Results of SCF Acceleration

To comprehensively investigate the significance of our method in accelerating SCF convergence, we present a detailed point-by-point comparison of SCF acceleration for different Hamiltonian prediction models. The results on three datasets are summarized in Figs. D.3-D.5. Remarkably, the models with self-consistency training always lead to faster convergence than conventional MINAO guess across various settings. In contrast, the label-based training method for uracil in Table 1 (see Fig. D.3(c)) and the `zero-shot` setting for two MD22 molecules in Table 5 (see Fig. D.5) result in slower convergence than MINAO on some structures, while applying self-consistency training can eliminate this issue.

D.6. Large-Scale Generalization Results

As discussed in Sec. 3.3, we assess the generalization of self-consistency training to large-scale MD22 structures by comparing it with two state-of-the-art end-to-end (`e2e`) property prediction models. For this purpose, we choose two

Table D.1. Generalization improvement by self-consistency training on unlabeled data *on various model architectures* in the *data-scarce* scenario (*MD17-Ethanol* Hamiltonian). Evaluated on the test split of conformations of the molecule. The setting is in parallel with Table 1.

Architecture	Setting	$\mathbf{H}$ $[\mu E_\mathrm{h}]\downarrow$	$\boldsymbol{\epsilon}$ $[\mu E_\mathrm{h}]\downarrow$	$\mathbf{C}$ $[\%]\uparrow$	ϵ_HOMO $[\mu E_\mathrm{h}]\downarrow$	ϵ_LUMO $[\mu E_\mathrm{h}]\downarrow$	ϵ_Δ $[\mu E_\mathrm{h}]\downarrow$	SCF Accel. $[\%]\downarrow$
QHNet	label	160.36	712.54	99.44	911.64	6800.84	6643.11	68.3
	label + self-con	**75.65**	**285.49**	**99.94**	**336.97**	**1203.60**	**1224.86**	**61.5**
PhiSNet	label	116.72	2702.13	98.63	1887.50	7954.97	6834.62	65.69
	label + self-con	**93.77**	**475.29**	**99.91**	**602.96**	**1645.25**	**1689.17**	**62.87**

Table D.2. SCF acceleration performance under two SCF iteration strategies, DIIS and SOSCF. (Results in the main paper are under DIIS.) Evaluated on the QH9-large test split in the *OOD* scenario (supervised training uses full extended labels; same setting as Table 3) and on the MD22 molecules in the larger-scale generalization scenario (same setting as Table 5).

Evaluation setting	Model setting	Iteration Strategy	SCF Accel. $[\%]\downarrow$
QH9-large	extended-label	DIIS	65.0
		SOSCF	57.8
	self-con	DIIS	64.5
		SOSCF	**56.2**
MD22 (ALA3)	zero-shot	DIIS	84.6
		SOSCF	71.8
	self-con	DIIS	64.7
		SOSCF	**47.5**
MD22 (DHA)	zero-shot	DIIS	170.8
		SOSCF	231.8
	self-con	DIIS	67.0
		SOSCF	**37.0**

advanced `e2e` architectures, ET[7] (Thölke & De Fabritiis, 2021) and Equiformer[8] (Liao & Smidt, 2022), as our baselines. They are representatives for equivariant architectures that utilize vector features and high-order tensor features, respectively. Despite the availability of pretrained models for these methods, they are not directly applicable for our analysis because they were originally trained on the QM9 dataset, which differs from the QH9 dataset used in our study. The two datasets use distinct orbital basis sets for DFT calculations, resulting in slightly different label distributions. To ensure a fair comparison, we retrain the ET and Equiformer models on the QH9-full training split, with the default hyper-parameter settings specified in their codebases. The performance of the two `e2e` predictors on the QH9-full validation set, as shown in Table D.3, aligns with the outcomes reported in their respective original publications, validating the effectiveness of our model replication. The substantial performance disparity between the QH9-full validation set and two MD22 molecules implies a significant generalization challenge for `e2e` predictors. Fortunately, this challenge can be potentially alleviated by applying self-consistency training to Hamiltonian prediction.

E. Limitations and Future work

Even though self-consistency training has shown improved efficiency than conventional DFT calculation for training a Hamiltonian prediction model or for solving a bunch of molecular structures (Sec. 3.2) by amortizing the cost of SCF iterations over queried molecular structures, the computational complexity remains the same as that of DFT. This complexity may still limit the applicability (but to a higher level) of Hamiltonian prediction to large molecular systems such as biomacromolecules. It would be a promising future work to reduce the complexity of evaluating the self-consistency loss by leveraging techniques from linear-scaling DFT algorithms. The Hamiltonian prediction model we used in this study, although is already more efficient than a few alternatives, still requires considerable cost to evaluate, due to *e.g.* the use of computationally expensive tensor product operations. This calls for designing more efficient neural network architectures for Hamiltonian prediction. Moreover, current Hamiltonian prediction models only support prediction under a specific basis set, which has a restricted flexibility to trade-off efficiency and accuracy, and is hard to leverage data under different choices of

[7] https://github.com/torchmd/torchmd-net, the code is freely available under the terms of the MIT license.
[8] https://github.com/atomicarchitects/equiformer, the code is freely available under the terms of the MIT license.

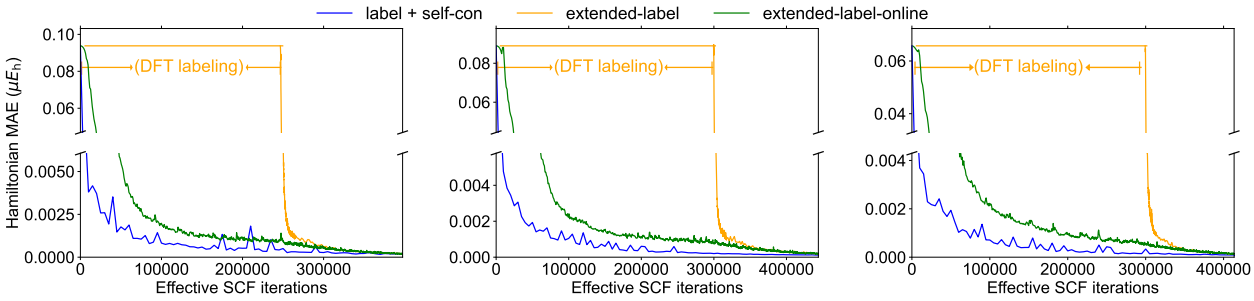


Figure D.1. Efficiency comparison in the *data-scarce* scenario (MD17 Hamiltonian) among `self-con`sistency training on unlabeled data, supervised training following DFT labeling on unlabeled data (`extended-label`), and supervised training along with DFT labeling (`extended-label-online`). The setting is in parallel with Fig. 3, with the only difference that the cost is measured by *the effective number of SCF iterations* consumed along the training process.

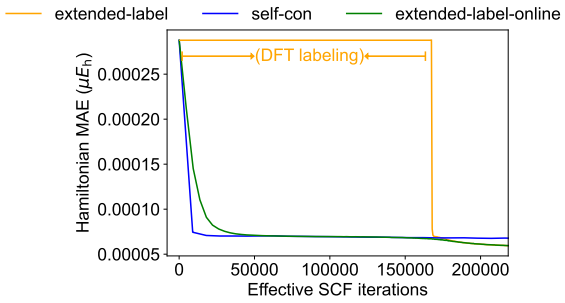
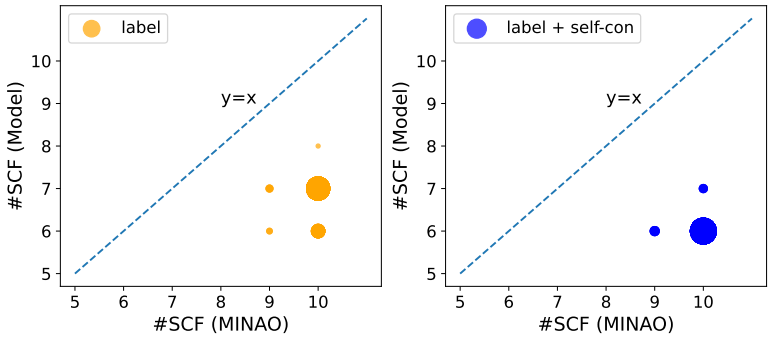


Figure D.2. Efficiency comparison in the *OOD* scenario (QH9) among fine-tuning using `self-con`sistency training on unlabeled data, supervised training following DFT labeling on unlabeled data (`extended-label`), and supervised training along with DFT labeling (`extended-label-online`). The setting is in parallel with Fig. 4(b) using the `adapter` fine-tuning strategy, with the only difference that the cost is measured by *the effective number of SCF iterations* consumed along the training process.

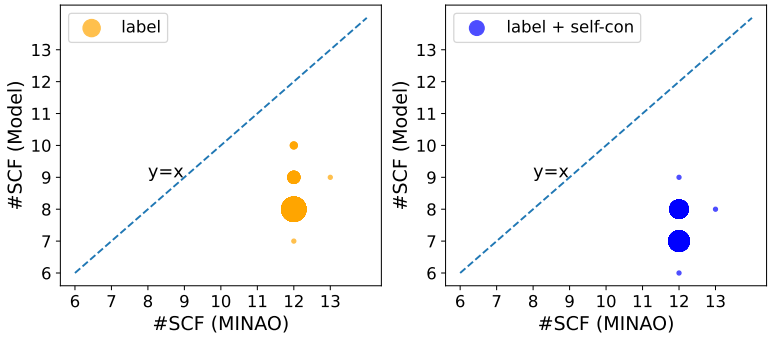
basis. A possible solution to this restriction is including the overlap matrix into the model input, which conveys information about the basis set in a form relevant to the given molecular structure.

Table D.3. Generalization results of `e2e` property predictors on *larger-scale* MD22 molecules. Models are pretrained on the QH9-full training split and directly evaluated on the large-scale molecules. The `e2e` predictors significantly suffer from the generalization gap. QH9(valid) denotes the QH9-full validation split.

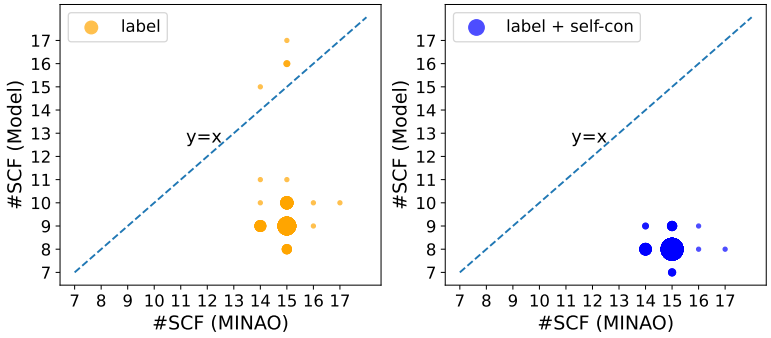
Model	ϵ_{HOMO} $[\mu E_{\text{h}}] \downarrow$			ϵ_{LUMO} $[\mu E_{\text{h}}] \downarrow$			ϵ_{Δ} $[\mu E_{\text{h}}] \downarrow$		
	QH9 (valid)	ALA3	DHA	QH9 (valid)	ALA3	DHA	QH9 (valid)	ALA3	DHA
e2e (ET)	818.77	1.74×10^5	2.92×10^5	540.22	7.72×10^4	2.58×10^4	1.38×10^3	2.38×10^5	3.39×10^5
e2e (Equiformer)	646.42	2.38×10^5	3.76×10^5	488.40	1.16×10^4	2.31×10^4	1.15×10^3	2.27×10^6	4.17×10^6



(a) Comparison of SCF acceleration on *MD17-ethanol* structures



(b) Comparison of SCF acceleration on *MD17-malondialdehyde* structures



(c) Comparison of SCF acceleration on *MD17-uracil* structures

Figure D.3. Comparison of SCF acceleration on *MD17* structures (in parallel with Table 1). Each sub-figure shows a scatter plot of the number of converged SCF steps from two initial guesses: MINAO (**x**-axis), and the predicted Hamiltonian (**y**-axis) by a model trained using labels (left) and additionally using self-consistency training (right). All figures are plotted using 50 data points.

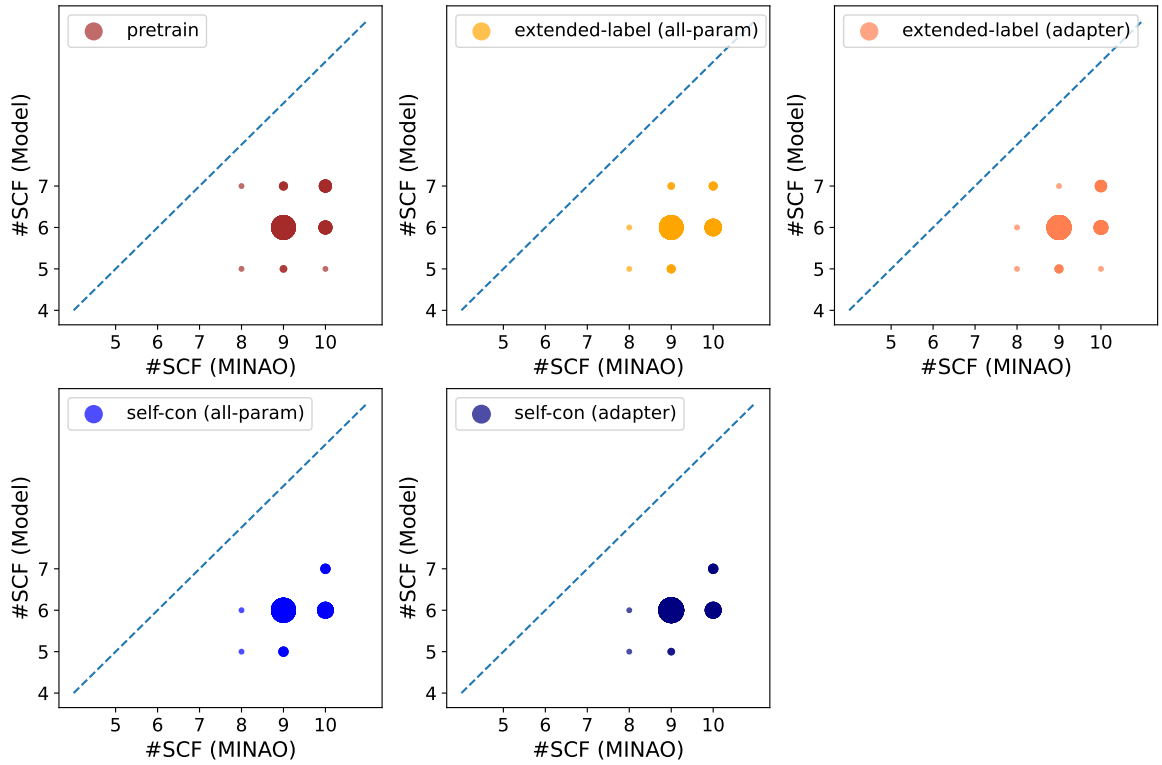
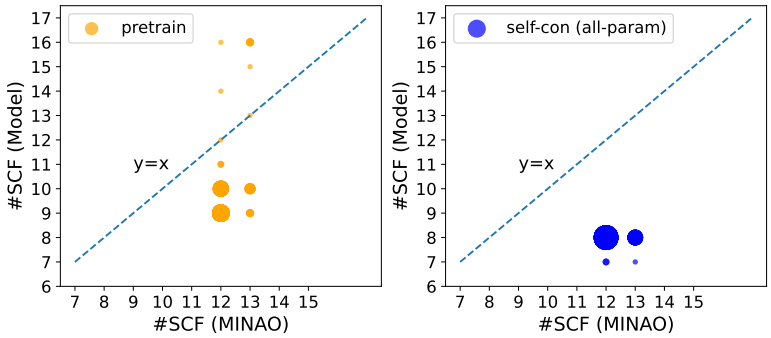
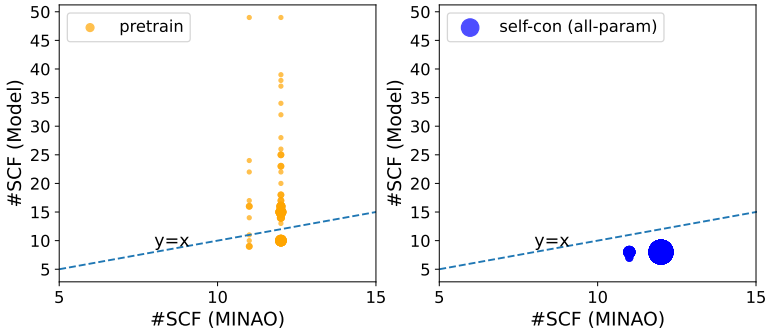


Figure D.4. Comparison of SCF acceleration on *QH9-large* structures (in parallel with Table 3). Each figure shows a scatter plot of the number of converged SCF steps from two initial guesses: MINAO (**x**-axis), and the predicted Hamiltonian (**y**-axis) by a model pretrained using labels (top left) and additionally finetuned using labels (top middle and right) or self-consistency loss (bottom left and middle) with two fine-tuning strategies. All figures are plotted using 50 data points.



(a) Comparison of SCF acceleration on *MD22-ALA3* structures



(b) Comparison of SCF acceleration on *MD22-DHA* structures

Figure D.5. Comparison of SCF acceleration on *MD22* structures (in parallel with Table 5). Each sub-figure shows a scatter plot of the number of converged SCF steps from two initial guesses: MINAO (**x**-axis), and the predicted Hamiltonian (**y**-axis) by a model pretrained using labels (left) and additionally finetuned using self-consistency loss with the `all-param` strategy (right). All figures are plotted using 50 data points.