nature computational science

Article

https://doi.org/10.1038/s43588-024-00605-8

Overcoming the barrier of orbital-free density functional theory for molecular systems using deep learning

Contents

Supplementary Section A Mechanism of Density Functional Theory

In this section, we introduce details in relevant theory on DFT, including the formulation of OFDFT under atomic basis, and the mechanism and details to use KSDFT to generate value and gradient data for learning KEDF. Atomic units are used through out the paper.

Supplementary Section A.1 Basic Formulation of Density Functional Theory

For brevity, the following formulation is for spinless fermions therefore only consider spacial states. For the restricted Kohn-Sham calculation we adopt for data generation, a pair of electrons of opposite spins share a common spacial orbital, which amounts to duplicate the orbitals in the following formulation.

The mechanism of DFT may be more intuitively introduced under Levy's constrained search formulation [1]. The N-electron Schrödinger equation for ground state is equivalent to the following optimization problem on N-electron wavefunctions $\psi(\mathbf{r}^{(1)}, \cdots, \mathbf{r}^{(N)})$ under the variational principle:

$$E^\star = \min_{\psi:\,\text{antisym}, \langle\psi|\psi\rangle = N} \langle\psi|\hat{T} + \hat{V}_{\text{ee}} + \hat{V}_{\text{ext}}|\psi\rangle, \tag{1}$$

where $\hat{T} := -\frac{1}{2}\sum_{i=1}^{N} \nabla^{(i)2}$ is the kinetic operator, $\hat{V}_{\text{ee}} := \sum_{1\leqslant i<j\leqslant N} \frac{1}{\|\mathbf{r}^{(i)} - \mathbf{r}^{(j)}\|}$ is the electron-electron Coulomb interaction (internal potential), and $\hat{V}_{\text{ext}} := \sum_{i=1}^{N} V_{\text{ext}}(\mathbf{r}^{(i)})$ comes from a one-body external potential $V_{\text{ext}}(\mathbf{r})$ that commonly arises from the electrostatic field of the nuclei specified by the given molecular structure $\mathcal{M} = \{\mathbf{X}, \mathbf{Z}\}$ where $\mathbf{X} := (\mathbf{x}^{(1)}, \cdots, \mathbf{x}^{(A)})$ and $\mathbf{Z} := (Z^{(1)}, \cdots, Z^{(A)})$:

$$V_{\text{ext}}(\mathbf{r}) = -\sum_{a=1}^{A} \frac{Z^{(a)}}{\|\mathbf{r} - \mathbf{x}^{(a)}\|}. \tag{2}$$

Although the optimization problem is exactly defined, directly optimizing the N-electron wavefunction is very challenging computationally. Specifying the wavefunction ψ and evaluating the energy already require an exponential cost in principle, as ψ is a function on $\mathbb{R}^{3N}$ whose dimension increases with N. To make an easier optimization problem, it is then desired to optimize a functional of the one-electron reduced density,

$$\rho_{[\psi]}(\mathbf{r}) := N \int |\psi(\mathbf{r}, \mathbf{r}^{(2)}, \cdots, \mathbf{r}^{(N)})|^2 \,\mathrm{d}\mathbf{r}^{(2)} \cdots \mathrm{d}\mathbf{r}^{(N)}, \tag{3}$$

which has an intuitive physical interpretation of charge density even under a classical view, and more importantly, the cost to specify a density is constant (with respect to N) in principle, as the density is a function on $\mathbb{R}^3$ whose dimension is constant. This is the starting point of density functional theory (DFT) [2–4], and is first formally verified by Hohenberg and Kohn [5].

In terms of the density, the external potential energy, that is, the last term in Eq. (1), is already an explicit density functional, since the external potential is one-body:

$$\langle\psi|\hat{V}_{\text{ext}}|\psi\rangle = \int \rho_{[\psi]}(\mathbf{r}) V_{\text{ext}}(\mathbf{r}) \,\mathrm{d}\mathbf{r} = E_{\text{ext}}[\rho_{[\psi]}], \quad \text{where } E_{\text{ext}}[\rho] := \int \rho(\mathbf{r}) V_{\text{ext}}(\mathbf{r}) \,\mathrm{d}\mathbf{r}. \tag{4}$$

For the other energy terms, using the density as an intermediate, the optimization problem in Eq. (1) can be equivalently (see Lieb [6] for the correspondence between the optimization space of ρ and of ψ to allow this equivalence) carried out in two levels:

$$E^\star = \min_{\rho:\geqslant 0, \int \rho(\mathbf{r})\,\mathrm{d}\mathbf{r} = N} \left(\min_{\psi:\,\text{antisym}, \rho_{[\psi]} = \rho} \langle\psi|\hat{T} + \hat{V}_{\text{ee}}|\psi\rangle \right) + E_{\text{ext}}[\rho] \tag{5}$$

$$= \min_{\rho:\geqslant 0, \int \rho(\mathbf{r})\,\mathrm{d}\mathbf{r} = N} \left\{ E[\rho] := U[\rho] + E_{\text{ext}}[\rho] \right\}. \tag{6}$$

Here, the result of the first-level optimization problem carrying out a constrained search in Eq. (5) defines a density functional,

$$U[\rho] := \min_{\psi:\,\text{antisym}, \rho_{[\psi]} = \rho} \langle\psi|\hat{T} + \hat{V}_{\text{ee}}|\psi\rangle, \tag{7}$$

called the universal functional, as it is independent of system specification (that is, $\mathcal{M}$ or V_{ext}). It is composed of the kinetic and internal potential energy of the electrons. The optimization objective is then formally converted to a density functional $E[\rho]$ as shown in Eq. (6).

Supplementary Table 1: Main notations. These notations are used consistently throughout the main text and the supplementary text.

Basic concepts	
$\mathbf{r} \in \mathbb{R}^3$	Electron coordinate
N	Number of electrons in a molecular system
$\psi(\mathbf{r}^{(1)}, \cdots, \mathbf{r}^{(N)})$	N-electron wavefunction
$\langle f \vert g \rangle := \int f(\mathbf{r}) g(\mathbf{r}) \,\mathrm{d}\mathbf{r}$	The standard inner product in function space
$\langle f \vert \hat{O} \vert g \rangle := \int f(\mathbf{r}) (\hat{O} g)(\mathbf{r}) \,\mathrm{d}\mathbf{r}$	Function-space inner product with operator
$(f \vert g) := \int \frac{f(\mathbf{r}) g(\mathbf{r}')}{\Vert \mathbf{r} - \mathbf{r}' \Vert} \,\mathrm{d}\mathbf{r}\mathrm{d}\mathbf{r}'$	Coulomb integral
Molecular system	
$\mathbf{x} \in \mathbb{R}^3$	Atom coordinates
$a, b \in \{1, 2, \cdots, A\}$	Indices for atoms in a molecule
$\mathbf{X} := \{\mathbf{x}^{(a)}\}_{a=1}^{A}$	Molecular conformation/geometry
$\mathbf{Z} := \{Z^{(a)}\}_{a=1}^{A}$	Molecular composition
$\mathcal{M} := \{\mathbf{X}, \mathbf{Z}\}$	Molecular structure
$\{\mathcal{M}^{(d)}\}_{d=1}^{D}$	Molecular structures in a dataset
Density functional theory	
$\Phi := \{\phi_i(\mathbf{r})\}_{i=1}^{N}$	Orbitals
$i, j \in \{1, 2, \cdots, N\}$	Indices for orbitals or electrons
$\psi_{[\Phi]}(\mathbf{r}^{(1)}, \cdots, \mathbf{r}^{(N)}) := \det[\phi_i(\mathbf{r}^{(j)})]_{ij}$	Slater determinant from orbitals $\Phi = \{\phi_i(\mathbf{r})\}_{i=1}^{N}$
$\{\eta_\alpha(\mathbf{r})\}_{\alpha=1}^{B}$	Orbital basis
Subscripts $\alpha, \beta, \gamma, \delta \in \{1, 2, \cdots, B\}$	Indices for orbital basis
$\mathbf{C}_{\alpha i}$	Orbital coefficients
$\mathbf{\Gamma}_{\alpha\beta} := \sum_{i=1}^{N} \mathbf{C}_{\alpha i} \mathbf{C}_{\beta i}$	Density matrix
$\mathbf{S}_{\alpha\beta} := \langle \eta_\alpha \vert \eta_\beta \rangle$	Overlap matrix of orbital basis
$\mathbf{D}_{\alpha\beta,\gamma\delta} := \langle \eta_\alpha \eta_\beta \vert \eta_\gamma \eta_\delta \rangle$	Overlap matrix of paired orbital basis
$\tilde{\mathbf{D}}_{\alpha\beta,\gamma\delta} := (\eta_\alpha \eta_\beta \vert \eta_\gamma \eta_\delta)$	4-center-2-electron Coulomb integral of orbital basis
$\rho(\mathbf{r})$	(One-electron reduced) density (function)
$\{\omega_\mu(\mathbf{r})\}_{\mu=1}^{M}$	Density basis
Subscripts $\mu, \nu \in \{1, 2, \cdots, M\}$	Indices for density basis
Subscript $\mu = (a, \tau)$	Atom assignment decomposition of basis index
$\mathbf{p}_\mu$	Density coefficient
$\mathbf{w}_\mu := \int \omega_\mu(\mathbf{r}) \,\mathrm{d}\mathbf{r}$	Density basis normalization vector
$\mathbf{W}_{\mu\nu} := \langle \omega_\mu \vert \omega_\nu \rangle$	Overlap matrix of density basis
$\tilde{\mathbf{W}}_{\mu\nu} := (\omega_\mu \vert \omega_\nu)$	2-center-2-electron Coulomb integral of density basis
$\mathbf{L}_{\mu,\alpha\beta} := \langle \omega_\mu \vert \eta_\alpha \eta_\beta \rangle$	Overlap matrix between density basis and orbital basis
$\mathbf{F}$	Fock matrix in KSDFT
k	Step index for SCF iteration or density optimization process ($\star$ for the converged step)
$(\mathbf{V}_{\text{eff}})_{\alpha\beta} := \langle \eta_\alpha \vert V_{\text{eff}} \vert \eta_\beta \rangle$	Effective potential matrix under orbital basis
$(\mathbf{v}_{\text{eff}})_\mu := \langle \omega_\mu \vert V_{\text{eff}} \rangle$	Effective potential vector under density basis
$\mu \in \mathbb{R}$	Chemical potential
$\boldsymbol{\varepsilon} := \mathrm{Diag}[\varepsilon_1, \cdots, \varepsilon_N]$	The diagonal $N \times N$ matrix of orbital energies
$U[\rho]$	Universal functional
$E_{\text{XC}}[\rho]$	Exchange-correlation (XC) functional
$T_{\text{S}}[\rho]$	Kinetic energy density functional (KEDF)
$T_{\text{S}}(\mathbf{p}, \mathcal{M})$	KEDF under atomic basis of density
$T_{\text{S},\theta}(\mathbf{p}, \mathcal{M})$	KEDF model/approximation
T_{APBE}	The APBE kinetic functional as the base functional $T_{\text{S,base}}$
$T_{\text{S,res}}$	Residual KEDF on top of the base functional $T_{\text{S,base}}$
E_{TXC}	Kinetic and XC functional
$\mathbf{f}$	Hellmann-Feynman force

To carry out practical computation, variants of the kinetic and internal potential energy that allow explicit calculation or have known properties are introduced, to cover the major part of the corresponding energies in $U[\rho]$. The internal potential energy is covered by its classical version, that is assuming no correlation, called the Hartree energy:

$$E_{\mathrm{H}}[\rho] := \frac{1}{2} \int \frac{\rho(\mathbf{r})\rho(\mathbf{r}')}{\|\mathbf{r}-\mathbf{r}'\|} \,\mathrm{d}\mathbf{r}\mathrm{d}\mathbf{r}'. \tag{8}$$

The kinetic energy is covered by the kinetic energy density functional (KEDF), which is defined in a similar way as the universal functional:

$$T_{\mathrm{S}}[\rho] := \min_{\psi:\,\text{antisym},\rho_{[\psi]}=\rho} \langle \psi | \hat{T} | \psi \rangle. \tag{9}$$

The remainder in the universal functional is called the *exchange-correlation (XC) functional*:

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

which is by definition also a density functional. Under this decomposition, the density optimization problem in Eq. (6) becomes:

$$E^\star = \min_{\rho:\geqslant 0,\int \rho(\mathbf{r})\,\mathrm{d}\mathbf{r}=N} \left\{ E[\rho] = T_{\mathrm{S}}[\rho] + \underbrace{E_{\mathrm{H}}[\rho] + E_{\mathrm{XC}}[\rho] + E_{\mathrm{ext}}[\rho]}_{=:E_{\mathrm{eff}}[\rho]} \right\}. \tag{10}$$

Here $E_{\mathrm{eff}}[\rho]$ is defined for future convenience and denoted after the effective-potential interpretation of its variation detailed later. Using carefully designed explicit expressions or machine-learning models to approximate the $T_{\mathrm{S}}[\rho]$ and $E_{\mathrm{XC}}[\rho]$ functionals, practical computation can be conducted. This is the formulation of orbital-free density functional theory (OFDFT). Indeed, the object to be optimized is the electron density, which is one function on the constant-dimensional space of $\mathbb{R}^3$, hence greatly reduces computation complexity over the original variational problem Eq. (1). Under properly designed $T_{\mathrm{S}}[\rho]$ and $E_{\mathrm{XC}}[\rho]$ approximations, the complexity is favorably $O(N^2)$ under atomic basis.

Considering the KEDF $T_{\mathrm{S}}[\rho]$ is more challenging to approximate than the XC functional $E_{\mathrm{XC}}[\rho]$, Kohn and Sham [7] leverage an equivalent formulation of KEDF to allow its accurate calculation, at the cost of increasing the complexity. The alternative formulation optimizes *determinantal* wavefunctions. A determinantal wavefunction for N electrons is specified by N one-electron wavefunctions $\Phi := \{\phi_i(\mathbf{r})\}_{i=1}^N$, a.k.a orbitals, following the form:

$$\psi_{[\Phi]}(\mathbf{r}^{(1)}, \cdots, \mathbf{r}^{(N)}) := \frac{1}{\sqrt{N!}} \det[\phi_i(\mathbf{r}^{(j)})]_{ij}, \quad \text{given orbitals } \Phi := \{\phi_i(\mathbf{r})\}_{i=1}^N.$$

The equivalent optimization problem in parallel with Eq. (9) is:

$$T_{\mathrm{S}}[\rho] = \min_{\{\phi_i\}_{i=1}^N:\rho_{[\Phi]}=\rho} \langle \psi_{[\Phi]} | \hat{T} | \psi_{[\Phi]} \rangle = \min_{\substack{\{\phi_i\}_{i=1}^N:\text{orthonormal},\\ \rho_{[\Phi]}=\rho}} \sum_{i=1}^N \langle \phi_i | \hat{T} | \phi_i \rangle, \tag{11}$$

$$\text{where} \quad \rho_{[\Phi]}(\mathbf{r}) := \sum_{i=1}^N |\phi_i(\mathbf{r})|^2, \quad \text{given orthonormal orbitals } \Phi := \{\phi_i(\mathbf{r})\}_{i=1}^N. \tag{12}$$

Note that the equivalence on defining $T_{\mathrm{S}}[\rho]$ by Eq. (9) and Eq. (11) is currently only known to be guaranteed if the density ρ comes from the ground state of a non-interacting system which is non-degenerate [6, Thm. 4.6]. Even there exists a density ρ that makes the determinantally-defined $T_{\mathrm{S}}[\rho]$ by Eq. (11) different [6, Thm. 4.8], the determinantally-defined $T_{\mathrm{S}}[\rho]$ still gives the right ground-state energy in density optimization (up to a closure) [6, Thm. 4.9].

In Eq. (11), the second equality holds since the density normalizes to N, and a set of (non-collinear) functions can always be orthogonalized, for example, using the Gram-Schmidt process ([8, Sec. 3.1.4]; [9, Sec. 14.3]). This equivalence can be understood from the interpretation of $T_{\mathrm{S}}[\rho]$ as the *non-interacting* portion of kinetic energy. Indeed, for a non-interacting system, there are only kinetic energy and external potential energy (that is, taking $\hat{V}_{\mathrm{ee}} = 0$), so the two-level optimization in parallel with Eq. (5) becomes:

$$\begin{aligned} E^\star &= \min_{\psi:\,\text{antisym},\langle\psi|\psi\rangle=1} \langle \psi | \hat{T} + \hat{V}_{\mathrm{ext}} | \psi \rangle \\ &= \min_{\rho:\geqslant 0,\int \rho(\mathbf{r})\,\mathrm{d}\mathbf{r}=N} \left(\min_{\psi:\,\text{antisym},\rho_{[\psi]}=\rho} \langle \psi | \hat{T} | \psi \rangle \right) + E_{\mathrm{ext}}[\rho] \\ &= \min_{\rho:\geqslant 0,\int \rho(\mathbf{r})\,\mathrm{d}\mathbf{r}=N} T_{\mathrm{S}}[\rho] + E_{\mathrm{ext}}[\rho], \end{aligned} \tag{13}$$

from which we see that $T_S[\rho]$ is the ground-state kinetic energy of the non-interacting system whose ground-state density is ρ. On the other hand, it is known that the ground-state wavefunction of a non-interacting system is commonly determinantal (at least when the ground state is non-degenerate [6, Thm. 4.6]). Hence the optimization can be rewritten as:

$$\begin{aligned} E^\star &= \min_{\{\phi_i\}_{i=1}^N} \langle \psi_{[\Phi]} | \hat{T} + \hat{V}_{\text{ext}} | \psi_{[\Phi]} \rangle \\ &= \min_{\rho:\geqslant 0, \int \rho(\mathbf{r})\,\mathrm{d}\mathbf{r} = N} \left(\min_{\{\phi_i\}_{i=1}^N : \rho_{[\Phi]} = \rho} \langle \psi_{[\Phi]} | \hat{T} | \psi_{[\Phi]} \rangle \right) + E_{\text{ext}}[\rho] \\ &= \min_{\rho:\geqslant 0, \int \rho(\mathbf{r})\,\mathrm{d}\mathbf{r} = N} \left(\min_{\substack{\{\phi_i\}_{i=1}^N : \text{orthonormal}, \\ \rho_{[\Phi]} = \rho}} \sum_{i=1}^N \langle \phi_i | \hat{T} | \phi_i \rangle \right) + E_{\text{ext}}[\rho], \end{aligned} \tag{14}$$

which indicates Eq. (11).

Back to the main problem, leveraging this knowledge about $T_S[\rho]$ in the variational problem Eq. (10) gives:

$$E^\star = \min_{\rho:\geqslant 0, \int \rho(\mathbf{r})\,\mathrm{d}\mathbf{r} = N} \left(\min_{\substack{\{\phi_i\}_{i=1}^N : \text{orthonormal}, \\ \rho_{[\Phi]} = \rho}} \sum_{i=1}^N \langle \phi_i | \hat{T} | \phi_i \rangle \right) + E_{\text{H}}[\rho] + E_{\text{XC}}[\rho] + E_{\text{ext}}[\rho],$$

which can be converted into directly optimizing the orbitals in a single-level optimization:

$$E^\star = \min_{\{\phi_i\}_{i=1}^N : \text{orthonormal}} \left\{ E[\Phi] := \sum_{i=1}^N \langle \phi_i | \hat{T} | \phi_i \rangle + \underbrace{E_{\text{H}}[\rho_{[\Phi]}] + E_{\text{XC}}[\rho_{[\Phi]}] + E_{\text{ext}}[\rho_{[\Phi]}]}_{E_{\text{eff}}[\rho_{[\Phi]}]} \right\}. \tag{15}$$

This is the formulation of Kohn-Sham density functional theory (KSDFT). With decades of development of XC functional approximations, KSDFT has achieved remarkable success and becomes among the most popular quantum chemistry method. In its formulation, the object to be optimized is a set of orbitals $\{\phi_i(\mathbf{r})\}_{i=1}^N$, which are N functions on $\mathbb{R}^3$. This is still substantially cheaper than optimizing an N-electron wavefunction on $\mathbb{R}^{3N}$, but has a complexity at least $O(N)$ times more than OFDFT, due to N times more $\mathbb{R}^3$ functions to optimize. Under atomic basis, KSDFT has a complexity at least $O(N^3)$ (using density fitting) without further approximations. In this triumphant era of deep machine learning, approximating a complicated functional is not as challenging as before. Powerful deep-learning models create the opportunity to approximate KEDF accurately enough to match successful XC functional approximations. This would enable accurate and practical OFDFT calculation, unleashing its power of lower complexity to push the accuracy-efficiency trade-off in quantum chemistry.

Supplementary Section A.2 KSDFT Calculation Produces Labels of KEDF

We now explain why a KSDFT calculation procedure could provide value and gradient labels of KEDF. Computation details under atomic basis are postponed in Supplementary Section A.4. We start by describing the typical algorithm to solve the optimization problem in KSDFT. To determine the optimal solution of orbitals $\Phi := \{\phi_i(\mathbf{r})\}_{i=1}^N$, the variation of the energy functional $E[\Phi]$ in Eq. (15) with respect to each orbital ϕ_i is required:

$$\begin{aligned} \frac{\delta E[\Phi]}{\delta \phi_i}(\mathbf{r}) &= \frac{\delta \sum_{j=1}^N (-1/2) \langle \phi_j | \nabla^2 | \phi_j \rangle}{\delta \phi_i}(\mathbf{r}) + \int \frac{\delta E_{\text{eff}}[\rho_{[\Phi]}]}{\delta \rho}(\mathbf{r}') \frac{\delta \rho_{[\Phi]}(\mathbf{r}')}{\delta \phi_i(\mathbf{r})}\,\mathrm{d}\mathbf{r}' \\ &= 2\hat{T}\phi_i(\mathbf{r}) + 2V_{\text{eff}[\rho_{[\Phi]}]}(\mathbf{r})\phi_i(\mathbf{r}), \end{aligned} \tag{16}$$

$$\text{where} \quad V_{\text{eff}[\rho]}(\mathbf{r}) := \frac{\delta E_{\text{eff}}[\rho]}{\delta \rho}(\mathbf{r}) = \underbrace{\int \frac{\rho(\mathbf{r}')}{\|\mathbf{r}' - \mathbf{r}\|}\,\mathrm{d}\mathbf{r}'}_{=:V_{\text{H}[\rho]}(\mathbf{r})} + \underbrace{\frac{\delta E_{\text{XC}}[\rho]}{\delta \rho}(\mathbf{r})}_{=:V_{\text{XC}[\rho]}(\mathbf{r})} + V_{\text{ext}}(\mathbf{r}). \tag{17}$$

The term $V_{\text{eff}[\rho_{[\Phi]}]}$ arises as a variation with respect to the density ρ since the orbitals affect the energy component E_{eff} apart from T_S (defined in Eq. (10)) only through the density $\rho_{[\Phi]}$ they define. By Eq. (12) we have $\frac{\delta \rho_{[\Phi]}(\mathbf{r}')}{\delta \phi_i(\mathbf{r})} = 2\phi_i(\mathbf{r})\delta(\mathbf{r} - \mathbf{r}')$, which then gives Eq. (16). Combining Eq. (16) with the variation of the orthonormal constraint yields the optimality equation for the problem Eq. (15):

$$\hat{F}_{[\rho_{[\Phi]}]}\phi_i := \hat{T}\phi_i + V_{\text{eff}[\rho_{[\Phi]}]}\phi_i = \varepsilon_i \phi_i, \quad \forall i = 1, \cdots N. \tag{18}$$

In the derivation, only the Lagrange multipliers ε_i for the normalization constraints $\langle\phi_i|\phi_i\rangle = 1$ are imposed, since from the resulting equations (18), $\{\phi_i\}_{i=1}^N$ are eigenstates of an Hermitian operator $\hat{F}_{[\rho_{[\Phi]}]}$ called the Fock operator, hence are naturally orthogonal in the general case of non-degeneracy. These equations resemble the Schrödinger equation for N non-interacting fermions, where $V_{\text{eff}[\rho_{[\Phi]}]}(\mathbf{r})$, as a function on $\mathbb{R}^3$, acts as an *effective* one-body external potential, hence the name.

Note that $V_{\text{eff}[\rho_{[\Phi]}]}$ is unknown beforehand, as itself depends on the solution of orbitals. Hence a fixed-point iteration is employed: starting from a set of initial orbitals $\Phi^{(0)} := \{\phi_i^{(0)}\}_{i=1}^N$, construct the Fock operator using results in previous iterations,

$$\hat{F}^{(k)} := \hat{T} + V_{\text{eff}}^{(k)}, \tag{19}$$

where $V_{\text{eff}}^{(k)}$ is taken as $V_{\text{eff}[\rho_{[\Phi^{(k-1)}]}]}$ following this derivation, that is, $\hat{F}^{(k)} = \hat{F}_{[\rho_{[\Phi^{(k-1)}]}]}$, and solve the corresponding eigenvalue problem for the orbitals in the current iteration:

$$\hat{F}^{(k)}\phi_i^{(k)} = \varepsilon_i^{(k)}\phi_i^{(k)}, \quad \forall i = 1, \cdots, N. \tag{20}$$

The iteration stops until "self-consistency" is achieved, that is, the eigenstate solution $\Phi^{(k)} := \{\phi_i^{(k)}\}_i$ in the current step coincides (up to an acceptable error) with the orbitals $\Phi^{(k-1)} := \{\phi_i^{(k-1)}\}_{i=1}^N$ in the previous step that define $\hat{F}^{(k)}$. This is the self-consistent field (SCF) method [10].

An important fact of SCF is that, in each iteration k, the solution $\{\phi_i^{(k)}\}_{i=1}^N$ exactly defines the ground-state of a *non-interacting* system of N fermions moving in the effective one-body potential $V_{\text{eff}}^{(k)}$ as the external potential V_{ext}. Indeed, the variational problem Eq. (14) that describes the non-interacting system can be reformulated into a single-level optimization as:

$$E^\star = \min_{\{\phi_i\}_{i=1}^N:\text{orthonormal}} \sum_{i=1}^N \langle\phi_i|\hat{T}|\phi_i\rangle + \int \rho_{[\Phi]}(\mathbf{r})V_{\text{eff}}^{(k)}(\mathbf{r})\,\mathrm{d}\mathbf{r}, \tag{21}$$

whose variation coincides with Eq. (20) thus solved by $\{\phi_i^{(k)}\}_{i=1}^N$. This reveals the relation of SCF solution to the KEDF: this solution of orbitals $\{\phi_i^{(k)}\}_{i=1}^N$ achieves the minimum non-interacting kinetic energy $\sum_{i=1}^N \langle\psi_{[\Phi]}|\hat{T}|\psi_{[\Phi]}\rangle$ among all orthonormal orbitals that lead to the same density $\rho_{[\Phi^{(k)}]}$ (otherwise Eq. (21) can be further minimized; can also be seen from the equivalence to Eq. (14)); by the alternative form of KEDF Eq. (11) as non-interacting ground-state kinetic energy, we thus have:

$$T_{\text{S}}[\rho_{[\Phi^{(k)}]}] = \langle\psi_{[\Phi^{(k)}]}|\hat{T}|\psi_{[\Phi^{(k)}]}\rangle = \sum_{i=1}^N \langle\phi_i^{(k)}|\hat{T}|\phi_i^{(k)}\rangle. \tag{22}$$

This indicates that *every SCF iteration produces a label for* T_{S}. Moreover, as the non-interacting variation problem Eq. (21) is equivalent to its two-level optimization form Eq. (14), which is in turn equivalent to the density optimization form using KEDF Eq. (13) (which explains the alternative KEDF form Eq. (11)), the density $\rho_{[\Phi^{(k)}]}$ from the solution $\Phi^{(k)} := \{\phi_i^{(k)}\}_{i=1}^N$ of each SCF iteration minimizes Eq. (13). Therefore, it satisfies the variation equation (Euler equation) of Eq. (13) (taking V_{ext} as $V_{\text{eff}}^{(k)}$) subject to the normalization constraint with Lagrange multiplier (chemical potential) $\mu^{(k)}$:

$$\frac{\delta T_{\text{S}}[\rho_{[\Phi^{(k)}]}]}{\delta\rho} + V_{\text{eff}}^{(k)} = \mu^{(k)}. \tag{23}$$

The variation of KEDF $\frac{\delta T_{\text{S}}}{\delta\rho}$ is related to the gradient with respect to density coefficients when the density ρ is expanded on a basis (see Supplementary Section A.4.3). Hence, *every SCF iteration also produces a label for the gradient of* T_{S}*, up to a projection.*

It is worth noting that these arguments still hold when the effective potential $V_{\text{eff}}^{(k)}$ in SCF iteration k is *not* $V_{\text{eff}[\rho_{[\Phi^{(k-1)}]}]}$, since the deductions from Eq. (21) to Eq. (23) only require $V_{\text{eff}}^{(k)}$ to be a one-body potential. This allows more flexible data generation process since in common DFT calculation settings, $V_{\text{eff}}^{(k)}$ indeed deviates from $V_{\text{eff}[\rho_{[\Phi^{(k-1)}]}]}$ for more stable and faster convergence, for example when using the "direct inversion in the iterative subspace" (DIIS) method [11, 12]. This also indicates that even when the XC functional used in data generation is not accurate, the generated value and gradient labels for $T_{\text{S}}[\rho]$ are still exact, since the XC functional still gives an effective one-body potential to define the non-interacting system Eq. (20) or Eq. (21), as long as it is pure (that is, only depends on density features). In this sense, data generation for KEDF is easier than that for the XC functional.

Supplementary Section A.3 Formulation under Atomic Basis

For practical calculation, KSDFT typically uses an atomic basis $\{\eta_\alpha(\mathbf{r})\}_{\alpha=1}^{B}$ to expand the orbitals for conducting the SCF iteration in Eq. (20) for molecular systems. The expansion gives:

$$\phi_i(\mathbf{r}) = \sum_\alpha \mathbf{C}_{\alpha i}\eta_\alpha(\mathbf{r}), \tag{24}$$

which converts solving for eigenfunctions into the common problem of solving for eigenvectors of a matrix. On the other hand, as emphasized in Introduction 1 and Results 2.1, we also hope to represent the density on an atomic basis $\{\omega_\mu(\mathbf{r})\}_{\mu=1}^{M}$ for efficient OFDFT implementation,

$$\rho(\mathbf{r}) = \sum_\mu \mathbf{p}_\mu \omega_\mu(\mathbf{r}). \tag{25}$$

The left-hand-sides of Eq. (24) and Eq. (25) may also be denoted as $\phi_{i,\mathbf{C}}$ or $\Phi_\mathbf{C}$ and $\rho_\mathbf{p}$ to highlight the dependency on the coefficients. Note that both the orbital basis $\{\eta_\alpha(\mathbf{r})\}_{\alpha=1}^{B}$ and density basis $\{\omega_\mu(\mathbf{r})\}_{\mu=1}^{M}$ depend on the molecular structure $\mathcal{M}$, as the location and type of each basis function is determined by the coordinates $\mathbf{x}^{(a)}$ and atomic number $Z^{(a)}$ of the corresponding atom. Nevertheless, the development in this subsection is for one given molecular system $\mathcal{M}$, so we omit its appearance for density or orbital representation. Typically, the numbers of basis B and M increase linearly with the number of electrons N, that is, $O(B) = O(M) = O(N)$.

Supplementary Section A.3.1 KSDFT under Atomic Basis

For the SCF iteration in Eq. (20), using the expansion of orbitals in Eq. (24), it becomes: $\sum_\beta \mathbf{C}_{\beta i}^{(k)} \hat{F}^{(k)} \eta_\beta(\mathbf{r}) = \varepsilon_i^{(k)} \sum_\beta \mathbf{C}_{\beta i}^{(k)} \eta_\beta(\mathbf{r}),\ \forall i = 1, \cdots, N$. Integrating each function equation with basis function $\eta_\alpha(\mathbf{r})$ gives: $\sum_\beta \mathbf{C}_{\beta i}^{(k)} \langle \eta_\alpha | \hat{F}^{(k)} | \eta_\beta \rangle = \varepsilon_i^{(k)} \sum_\beta \mathbf{C}_{\beta i}^{(k)} \langle \eta_\alpha | \eta_\beta \rangle$, which can then be formulated as a generalized eigenvalue problem in matrix form:

$$\mathbf{F}^{(k)}\mathbf{C}^{(k)} = \mathbf{S}\mathbf{C}^{(k)}\boldsymbol{\varepsilon}^{(k)}, \tag{26}$$

$$\text{where} \quad \mathbf{F}_{\alpha\beta}^{(k)} := \langle \eta_\alpha | \hat{F}^{(k)} | \eta_\beta \rangle \overset{\text{Eq. (19)}}{=} \underbrace{\langle \eta_\alpha | \hat{T} | \eta_\beta \rangle}_{= -\frac{1}{2}\int \eta_\alpha(\mathbf{r})\nabla^2\eta_\beta(\mathbf{r})\,\mathrm{d}\mathbf{r} =: \mathbf{T}_{\alpha\beta}} + \underbrace{\langle \eta_\alpha | V_{\text{eff}}^{(k)} | \eta_\beta \rangle}_{=: (\mathbf{V}_{\text{eff}}^{(k)})_{\alpha\beta}}, \tag{27}$$

$$\mathbf{S}_{\alpha\beta} := \langle \eta_\alpha | \eta_\beta \rangle, \quad \boldsymbol{\varepsilon}^{(k)} := \begin{pmatrix} \varepsilon_1^{(k)} & & \\ & \ddots & \\ & & \varepsilon_N^{(k)} \end{pmatrix}.$$

Here, $\mathbf{F}^{(k)}$ is called the Fock matrix, and $\mathbf{S}$ is the overlap matrix of the orbital basis.

To show the expression of the Fock matrix, we first give the expression of the density defined by the orbital coefficients from Eq. (12) and Eq. (24):

$$\begin{aligned} \rho_\mathbf{C}(\mathbf{r}) := \rho_{[\Phi_\mathbf{C}]}(\mathbf{r}) &= \sum_{i=1}^{N} \left| \sum_\alpha \mathbf{C}_{\alpha i}\eta_\alpha(\mathbf{r}) \right|^2 = \sum_{i=1}^{N} \sum_{\alpha\beta} \mathbf{C}_{\alpha i}\mathbf{C}_{\beta i}\eta_\alpha(\mathbf{r})\eta_\beta(\mathbf{r}) \\ &= \sum_{\alpha\beta} \mathbf{\Gamma}_{\alpha\beta}\eta_\alpha(\mathbf{r})\eta_\beta(\mathbf{r}), \end{aligned} \tag{28}$$

where we have defined the density matrix corresponding to the orbital coefficients:

$$\mathbf{\Gamma} := \mathbf{C}\mathbf{C}^\top, \quad \mathbf{\Gamma}_{\alpha\beta} := \sum_{i=1}^{N} \mathbf{C}_{\alpha i}\mathbf{C}_{\beta i}. \tag{29}$$

Note that Eq. (12) requires orthonormal orbitals, hence Eq. (28) requires $\mathbf{C}$ to satisfy the corresponding orthonormality constraint shown in Eq. (39) below. Orbital coefficient solutions $\mathbf{C}^{(k)}$ in SCF iterations satisfy this constraint as explained later.

In the derivation of SCF iteration, $V_{\text{eff}}^{(k)}$ is taken as $V_{\text{eff}[\rho_{[\Phi^{(k-1)}]}]}$, that is, $\hat{F}^{(k)} = \hat{F}_{[\rho_{[\Phi^{(k-1)}]}]}$. This allows explicit calculation of $\mathbf{V}_{\text{eff}}^{(k)}$ as $\mathbf{V}_{\text{eff}\,\mathbf{C}^{(k-1)}}$ based on Eq. (17) and $\mathbf{F}^{(k)}$ as $\mathbf{F}_{\mathbf{C}^{(k-1)}}$, for which we introduce the following series of definitions:

$$\begin{aligned}
&\mathbf{F}_{\mathbf{C}} := [\langle \eta_\alpha | \hat{F}_{[\rho_{\mathbf{C}}]} | \eta_\beta \rangle]_{\alpha\beta} = \mathbf{T} + \mathbf{V}_{\text{eff}\mathbf{C}} \quad (\mathbf{T} \text{ defined in Eq. (27)}), \\
&\mathbf{V}_{\text{eff}\mathbf{C}} := [\langle \eta_\alpha | V_{\text{eff}[\rho_{\mathbf{C}}]} | \eta_\beta \rangle]_{\alpha\beta} = \mathbf{V}_{\text{H}\mathbf{C}} + \mathbf{V}_{\text{XC}\mathbf{C}} + \mathbf{V}_{\text{ext}}, \qquad (30)
\end{aligned}$$

$$\begin{aligned}
\text{where} \quad (\mathbf{V}_{\text{H}\mathbf{C}})_{\alpha\beta} &:= \langle \eta_\alpha | V_{\text{H}[\rho_{\mathbf{C}}]} | \eta_\beta \rangle \overset{\text{Eqs. (17, 28)}}{=} \iint \eta_\alpha(\mathbf{r}) \eta_\beta(\mathbf{r}) \frac{\sum_{\gamma\delta} \mathbf{\Gamma}_{\gamma\delta} \eta_\gamma(\mathbf{r}') \eta_\delta(\mathbf{r}')}{\|\mathbf{r} - \mathbf{r}'\|} \, \mathrm{d}\mathbf{r}' \mathrm{d}\mathbf{r} \\
&= \sum_{\gamma\delta} \tilde{\mathbf{D}}_{\alpha\beta,\gamma\delta} \mathbf{\Gamma}_{\gamma\delta} = (\tilde{\mathbf{D}} \bar{\mathbf{\Gamma}})_{\alpha\beta}, \qquad (31) \\
&\text{where } \tilde{\mathbf{D}}_{\alpha\beta,\gamma\delta} := (\eta_\alpha \eta_\beta | \eta_\gamma \eta_\delta), \ \bar{\mathbf{\Gamma}} \text{ is the vector of flattened } \mathbf{\Gamma}, \qquad (32) \\
(\mathbf{V}_{\text{XC}\mathbf{C}})_{\alpha\beta} &:= \langle \eta_\alpha | V_{\text{XC}[\rho_{\mathbf{C}}]} | \eta_\beta \rangle = \int V_{\text{XC}[\rho_{\mathbf{C}}]}(\mathbf{r}) \eta_\alpha(\mathbf{r}) \eta_\beta(\mathbf{r}) \, \mathrm{d}\mathbf{r}, \\
(\mathbf{V}_{\text{ext}})_{\alpha\beta} &:= \langle \eta_\alpha | V_{\text{ext}} | \eta_\beta \rangle = -\sum_{a=1}^{A} Z^{(a)} \int \frac{\eta_\alpha(\mathbf{r}) \eta_\beta(\mathbf{r})}{\|\mathbf{r} - \mathbf{x}^{(a)}\|} \, \mathrm{d}\mathbf{r}.
\end{aligned}$$

In defining $\tilde{\mathbf{D}}$, we have used the notation of Coulomb integral for brevity:

$$(f|g) := \int \frac{f(\mathbf{r}) g(\mathbf{r}')}{\|\mathbf{r} - \mathbf{r}'\|} \, \mathrm{d}\mathbf{r} \mathrm{d}\mathbf{r}'. \qquad (33)$$

In practice, integrals in $\mathbf{S}$, $\mathbf{T}$, $\mathbf{V}_{\text{ext}}$, and the Coulomb integral $\tilde{\mathbf{D}}$ can be calculated analytically under Gaussian-Type Orbitals (GTO) as the basis $\{\eta_\alpha\}_{\alpha=1}^{B}$. The integral in $\mathbf{V}_{\text{XC}\mathbf{C}}$ is conducted numerically on a quadrature grid, as typically used in a DFT calculation. After convergence, the electronic energy can be calculated from the orbital coefficients $\mathbf{C}$ by:

$$E(\mathbf{C}) := E[\Phi_{\mathbf{C}}] \overset{\text{Eq. (15)}}{=} T_{\text{S}}(\mathbf{C}) + \underbrace{E_{\text{H}}(\mathbf{C}) + E_{\text{XC}}(\mathbf{C}) + E_{\text{ext}}(\mathbf{C})}_{=:E_{\text{eff}}(\mathbf{C})}, \qquad (34)$$

$$\begin{aligned}
\text{where} \quad T_{\text{S}}(\mathbf{C}) &:= \sum_{i=1}^{N} \langle \phi_{i,\mathbf{C}} | \hat{T} | \phi_{i,\mathbf{C}} \rangle = \sum_{\alpha\beta} \mathbf{\Gamma}_{\alpha\beta} \mathbf{T}_{\alpha\beta} = \bar{\mathbf{\Gamma}}^\top \bar{\mathbf{T}}, \qquad (35) \\
E_{\text{H}}(\mathbf{C}) &:= E_{\text{H}}[\rho_{\mathbf{C}}] = \frac{1}{2} \sum_{\alpha\beta\gamma\delta} \mathbf{\Gamma}_{\alpha\beta} \tilde{\mathbf{D}}_{\alpha\beta,\gamma\delta} \mathbf{\Gamma}_{\gamma\delta} = \frac{1}{2} \bar{\mathbf{\Gamma}}^\top \tilde{\mathbf{D}} \bar{\mathbf{\Gamma}}, \qquad (36) \\
E_{\text{XC}}(\mathbf{C}) &:= E_{\text{XC}}[\rho_{\mathbf{C}}] = E_{\text{XC}}\Big[\sum_{\alpha\beta} \mathbf{\Gamma}_{\alpha\beta} \eta_\alpha \eta_\beta\Big], \qquad (37) \\
E_{\text{ext}}(\mathbf{C}) &:= E_{\text{ext}}[\rho_{\mathbf{C}}] = \sum_{\alpha\beta} \mathbf{\Gamma}_{\alpha\beta} (\mathbf{V}_{\text{ext}})_{\alpha\beta} = \bar{\mathbf{\Gamma}}^\top \bar{\mathbf{V}}_{\text{ext}}, \qquad (38)
\end{aligned}$$

where $\bar{\mathbf{\Gamma}}$, $\bar{\mathbf{T}}$, $\bar{\mathbf{V}}_{\text{ext}}$ are the vectors of flattened $\mathbf{\Gamma}$, $\mathbf{T}$, $\mathbf{V}_{\text{ext}}$, respectively. The term $E_{\text{XC}}[\rho_{\mathbf{C}}]$ is again calculated by numerically integrating the defined density by $\mathbf{C}$ on a quadrature grid.

Computational Complexity Note that the construction of $\mathbf{V}_{\text{H}\mathbf{C}}$ from Eq. (31) and the evaluation of $E_{\text{H}}(\mathbf{C})$ from Eq. (36) require $O(B^4) = O(N^4)$ complexity. Even when using density fitting which decreases the complexity to $O(N^2)$, the complexity in each SCF iteration of KSDFT is $O(N^3)$ since the complexity of density fitting itself is $O(N^3)$ (see Supplementary Section A.4.1).

Orbital Orthonormality Under the atomic basis, orbital orthonormality $\langle \phi_i | \phi_j \rangle = \delta_{ij}$ becomes $\sum_{\alpha\beta} \mathbf{C}_{\alpha i} \mathbf{C}_{\beta j} \langle \eta_\alpha | \eta_\beta \rangle = \delta_{ij}$, or in matrix form,

$$\mathbf{C}^\top \mathbf{S} \mathbf{C} = \mathbf{I}. \qquad (39)$$

As mentioned after Eq. (18), only the normalization constraints need to be taken care of, as the orbitals are eigenstates of an Hermitian operator hence are already orthogonal if non-degenerate. This property transmits to the matrix form of the problem:

$$\mathbf{C}_{:i}^\top \mathbf{S} \mathbf{C}_{:i} = 1, \quad \forall i = 1, \cdots, N. \qquad (40)$$

(This can also be directly verified in the matrix form: for $i \neq j$, $(\mathbf{C}^\top \mathbf{S} \mathbf{C})_{ij} = \mathbf{C}_{:i}^\top \mathbf{S} \mathbf{C}_{:j} \overset{\text{Eq. (26)}}{=} \mathbf{C}_{:i}^\top \frac{1}{\varepsilon_j} \mathbf{F} \mathbf{C}_{:j} \overset{\mathbf{F}\text{ is Hermitian}}{=} \frac{1}{\varepsilon_j} (\mathbf{F} \mathbf{C}_{:i})^\top \mathbf{C}_{:j} \overset{\text{Eq. (26)}}{=} \frac{\varepsilon_i}{\varepsilon_j} (\mathbf{S} \mathbf{C}_{:i})^\top \mathbf{C}_{:j} \overset{\mathbf{S}\text{ is Hermitian}}{=} \frac{\varepsilon_i}{\varepsilon_j} (\mathbf{C}^\top \mathbf{S} \mathbf{C})_{ij}$, which indicates $\big(1 - \frac{\varepsilon_i}{\varepsilon_j}\big) (\mathbf{C}^\top \mathbf{S} \mathbf{C})_{ij} = 0$, thus $(\mathbf{C}^\top \mathbf{S} \mathbf{C})_{ij} = 0$ assuming non-degeneracy $\varepsilon_i \neq \varepsilon_j$.) Fulfilling the orthonormality constraint then only needs to normalize each eigenvector $\tilde{\mathbf{C}}_{:i}^{(k)}$ of the problem in Eq. (26) to form $\mathbf{C}^{(k)}$; explicitly, $\mathbf{C}_{:i}^{(k)} = \tilde{\mathbf{C}}_{:i}^{(k)} / \sqrt{\tilde{\mathbf{C}}_{:i}^{(k)\top} \mathbf{S} \tilde{\mathbf{C}}_{:i}^{(k)}}$.

Relation to Direct Gradient Derivation The matrix form of the optimality equation under basis hence the SCF iteration problem Eq. (26) can also be derived directly from Eq. (15) by taking the gradient of the energy function of coefficients: $E(\mathbf{C}) := E[\Phi_{\mathbf{C}}] = E\big[\{\sum_\alpha \mathbf{C}_{\alpha i}\eta_\alpha\}_{i=1}^N\big]$. Its gradient is related to the variation of the functional of orbitals by integral with the basis:

$$\big(\nabla_{\mathbf{C}} E(\mathbf{C})\big)_{\alpha i} = \int \frac{\delta E[\Phi_{\mathbf{C}}]}{\delta \phi_i}(\mathbf{r})\big(\nabla_{\mathbf{C}}\phi_{i,\mathbf{C}}(\mathbf{r})\big)_{\alpha i}\,\mathrm{d}\mathbf{r} = \int \frac{\delta E[\Phi_{\mathbf{C}}]}{\delta \phi_i}(\mathbf{r})\eta_\alpha(\mathbf{r})\,\mathrm{d}\mathbf{r}. \tag{41}$$

The variation is given by Eq. (16), which is $2\hat{F}_{[\rho_{\Phi_{\mathbf{C}}}]}\phi_{i,\mathbf{C}}(\mathbf{r}) = 2\sum_\beta \mathbf{C}_{\beta i}\hat{F}_{[\rho_{\Phi_{\mathbf{C}}}]}\eta_\beta(\mathbf{r})$, which turns the gradient into matrix form:

$$\nabla_{\mathbf{C}} E(\mathbf{C}) = 2\mathbf{F}_{\mathbf{C}}\mathbf{C}. \tag{42}$$

For the orbital orthonormality constraint, as mentioned, only the normalization constraints require explicit treatment. By introducing Lagrange multiplier ε_i for each constraint in Eq. (40) and taking the gradient for the corresponding Lagrange term gives $\nabla_{\mathbf{C}}\sum_{i=1}^N \varepsilon_i\big(\mathbf{C}_{:i}^\top \mathbf{S}\mathbf{C}_{:i} - 1\big) = 2\mathbf{S}\mathbf{C}\boldsymbol{\varepsilon}$. This leads to the optimality equation in matrix form:

$$\mathbf{F}_{\mathbf{C}}\mathbf{C} = \mathbf{S}\mathbf{C}\boldsymbol{\varepsilon}. \tag{43}$$

By constructing the corresponding fixed-point iteration, Eq. (26) is derived.

Accelerating and Stabilizing SCF Iteration As mentioned, $\mathbf{F}^{(k)}$ and $\mathbf{V}_{\text{eff}}^{(k)}$ may be taken differently from $\mathbf{F}_{\mathbf{C}^{(k-1)}}$ and $\mathbf{V}_{\text{eff}\,\mathbf{C}^{(k-1)}}$ for more stable and faster convergence. The direct inversion in the iterative subspace (DIIS) method [11, 12] is a popular choice for this. In DIIS, the Fock matrix $\mathbf{F}^{(k)}$ in the eigenvalue problem Eq. (20) for each SCF iteration k is taken as a weighted mixing of the vanilla Fock matrices $\mathbf{F}_{\mathbf{C}^{(k')}}$, $k' < k$ in previous steps:

$$\mathbf{F}^{(k)} := \sum_{k'=0}^{k-1} \pi_{k'}^{(k)} \mathbf{F}_{\mathbf{C}^{(k')}},$$

where $\{\pi_{k'}^{(k)}\}_{k'=0}^{k-1}$ are the weights that are positive and normalized $\sum_{k'=0}^{k-1}\pi_{k'}^{(k)} = 1$. Due to the normalization, the kinetic part $\mathbf{T}$ of the matrix remains the same, so it agrees with the form in Eq. (27) (or Eq. (19) in operator form), where:

$$\mathbf{V}_{\text{eff}}^{(k)} := \sum_{k'=0}^{k-1} \pi_{k'}^{(k)} \mathbf{V}_{\text{eff}\,\mathbf{C}^{(k')}}. \tag{44}$$

Supplementary Section A.3.2 OFDFT under Atomic Basis

To solve the optimization problem of OFDFT in Eq. (10), it is unnatural to construct a fixed-point SCF iteration process from its variation in Eq. (23). Hence, direct gradient-based density optimization is conducted. For this, the energy functional of density function in Eq. (10) needs to be converted into a function of density coefficients using the basis expansion of density function in Eq. (25):

$$E(\mathbf{p}) := E[\rho_{\mathbf{p}}] = E\Big[\sum_\mu \mathbf{p}_\mu \omega_\mu\Big] = T_{\text{S}}(\mathbf{p}) + \underbrace{E_{\text{H}}(\mathbf{p}) + E_{\text{XC}}(\mathbf{p}) + E_{\text{ext}}(\mathbf{p})}_{=E_{\text{eff}}(\mathbf{p}) := E_{\text{eff}}[\rho_{\mathbf{p}}]}, \tag{45}$$

$$\text{where}\quad T_{\text{S}}(\mathbf{p}) := T_{\text{S}}[\rho_{\mathbf{p}}] = T_{\text{S}}\Big[\sum_\mu \mathbf{p}_\mu\omega_\mu\Big], \tag{46}$$

$$E_{\text{H}}(\mathbf{p}) := E_{\text{H}}[\rho_{\mathbf{p}}] = \frac{1}{2}\iint \frac{\sum_\mu \mathbf{p}_\mu\omega_\mu(\mathbf{r})\sum_\nu \mathbf{p}_\nu\omega_\nu(\mathbf{r}')}{\|\mathbf{r}-\mathbf{r}'\|}\,\mathrm{d}\mathbf{r}\mathrm{d}\mathbf{r}' = \frac{1}{2}\mathbf{p}^\top\tilde{\mathbf{W}}\mathbf{p}, \tag{47}$$

$$E_{\text{XC}}(\mathbf{p}) := E_{\text{XC}}[\rho_{\mathbf{p}}] = E_{\text{XC}}\Big[\sum_\mu \mathbf{p}_\mu\omega_\mu\Big], \tag{48}$$

$$E_{\text{ext}}(\mathbf{p}) := E_{\text{ext}}[\rho_{\mathbf{p}}] = \int \sum_\mu \mathbf{p}_\mu\omega_\mu(\mathbf{r})V_{\text{ext}}(\mathbf{r})\,\mathrm{d}\mathbf{r} = \mathbf{p}^\top\mathbf{v}_{\text{ext}}, \tag{49}$$

$$\text{where}\quad \tilde{\mathbf{W}}_{\mu\nu} := (\omega_\mu|\omega_\nu), \quad (\mathbf{v}_{\text{ext}})_\mu := \langle\omega_\mu|V_{\text{ext}}\rangle.$$

The Coulomb integral notation $(\omega_\mu|\omega_\nu)$ is defined in Eq. (33). Recall that we have omitted the dependency of density basis $\{\omega_\mu\}_\mu$ hence of the functions for example $T_{\text{S}}(\mathbf{p})$ on the molecular structure $\mathcal{M}$ in this subsection. Integrals for $\tilde{\mathbf{W}}$ and $\mathbf{v}_{\text{ext}}$ can be calculated directly [13, 14] under Gaussian-Type Orbitals (GTO) as the basis $\{\omega_\mu\}_\mu$, using software libraries for example `libcint` [15] in PySCF [16]. The term $E_{\text{XC}}[\rho_{\mathbf{p}}]$ is calculated by numerically integrating the defined density $\rho_{\mathbf{p}}$ on a quadrature grid as typically used in a DFT calculation. In our M-OFDFT, $T_{\text{S}}(\mathbf{p})$ is calculated directly from the coefficient $\mathbf{p}$ using the deep-learning model $T_{\text{S},\theta}(\mathbf{p}, \mathcal{M})$.

To carry out direct optimization using a learned KEDF model $T_{\text{S},\theta}(\mathbf{p})$, the gradient of the electronic energy in Eq. (45) is required, which is given by:

$$\nabla_{\mathbf{p}} E_\theta(\mathbf{p}) = \nabla_{\mathbf{p}} T_{\text{S},\theta}(\mathbf{p}) + \underbrace{\tilde{\mathbf{W}}\mathbf{p} + \nabla_{\mathbf{p}} E_{\text{XC}}(\mathbf{p}) + \mathbf{v}_{\text{ext}}}_{\nabla_{\mathbf{p}} E_{\text{eff}}(\mathbf{p})}. \tag{50}$$

This gradient is then used to update the density coefficient after projected onto the linear subspace of normalized densities, following Eq. (2).

Relation to Derivation as Integral of the Variation with Basis The gradient $\nabla_{\mathbf{p}} E(\mathbf{p})$ can also be derived by the relation between gradient and variation that we have already seen in Eq. (41):

$$\big(\nabla_{\mathbf{p}} E(\mathbf{p})\big)_\mu = \big(\nabla_{\mathbf{p}} E[\rho_{\mathbf{p}}]\big)_\mu = \int \frac{\delta E[\rho_{\mathbf{p}}]}{\delta \rho}(\mathbf{r}) \big(\nabla_{\mathbf{p}} \rho_{\mathbf{p}}(\mathbf{r})\big)_\mu \,\mathrm{d}\mathbf{r} = \int \frac{\delta E[\rho_{\mathbf{p}}]}{\delta \rho}(\mathbf{r}) \omega_\mu(\mathbf{r}) \,\mathrm{d}\mathbf{r}. \tag{51}$$

Integrating the variations given in Eq. (17) with the basis functions $\{\omega_\mu\}_\mu$ recovers the Hartree energy gradient $\tilde{\mathbf{W}}\mathbf{p}$ and external energy gradient $\mathbf{v}_{\text{ext}}$ in Eq. (50). The formula also applies to the gradient of the kinetic energy $\nabla_{\mathbf{p}} T_{\text{S}}(\mathbf{p})$ and the gradient of the XC energy $\nabla_{\mathbf{p}} E_{\text{XC}}(\mathbf{p})$.

Automatic Differentiation Implementation for Calculating the Gradient In the implementation of M-OFDFT, the gradient of the KEDF model $\nabla_{\mathbf{p}} T_{\text{S},\theta}(\mathbf{p}, \mathcal{M})$ is evaluated directly using automatic differentiation [17], which can be conveniently done if implementing the model using common deep-learning programming frameworks, for example, PyTorch [18]. To calculate $\nabla_{\mathbf{p}} E_{\text{XC}}(\mathbf{p})$ conveniently, we also re-implemented the PBE XC functional [19] in PySCF using PyTorch and evaluate its gradient also by automatic differentiation. For material systems, automatic differentiation implementation of OFDFT is also developed recently [20]. When using the residual version $T_{\text{S,res},\theta}$ of KEDF model, which is detailed in Eq. (78) in Supplementary Section B.4 later, the base KEDF (taken as the APBE KEDF [21]) is also implemented in this way.

Computational Complexity As will be detailed in Supplementary Section B, the Transformer-based [22] KEDF model for our M-OFDFT has a quadratic complexity $O(A^2) = O(N^2)$. The PBE functional [19] for E_{XC} and the APBE functional [21] for the base KEDF are at the GGA level (generalized gradient approximation), so evaluating the energies amounts to calculating the density features with $O(M)$ cost on each grid point, in total with $O(MN_{\text{grid}})$ cost where N_{grid} is the number of grid points, and then conducting the quadrature with $O(N_{\text{grid}})$ cost. The complexity for these energies is thus $O(MN_{\text{grid}})$ which is also quadratic $O(N^2)$ since $N_{\text{grid}} = O(N)$ (though with a large prefactor). Evaluating the gradient using automatic differentiation is in the same order of evaluating the function, hence also has $O(N^2)$ complexity. Evaluating $E_{\text{H}}(\mathbf{p})$ and $E_{\text{ext}}(\mathbf{p})$ using Eq. (47) and Eq. (49) and their gradients using Eq. (50) require $O(M^2) = O(N^2)$ complexity. Therefore, the complexity in M-OFDFT has a quadratic complexity $O(N^2)$, which is indeed lower than that of KSDFT (detailed in Supplementary Section A.3.1 above).

Besides the advantage in asymptotic complexity, the fact that M-OFDFT is implemented in PyTorch(see Supplementary Section B.1.6) enables it to leverage GPUs efficiently. These factors jointly facilitate the much higher throughput of M-OFDFT than KSDFT.

Supplementary Section A.4 Label Calculation under Atomic Basis

This subsection details the calculation of the data tuple $(\mathbf{p}^{(k)}, T_{\text{S}}^{(k)}, \nabla_{\mathbf{p}} T_{\text{S}}^{(k)})$ for learning a KEDF model from the orbital coefficients $\mathbf{C}^{(k)}$ of the orbital solution $\Phi^{(k)} := \{\phi_i^{(k)}\}_{i=1}^N$ in each SCF iteration k. Following the previous subsection, we omit the appearance of $\mathcal{M}$ for density or orbital representation (for example, in $T_{\text{S}}(\mathbf{p}, \mathcal{M})$) and omit the index d for different molecular systems. We insist keeping the k index to reflect that the deduction is based on the solution in an SCF iteration but does not apply to arbitrary orbital coefficients $\mathbf{C}$.

Supplementary Section A.4.1 Density Fitting

We start with calculating the density coefficient $\mathbf{p}^{(k)}$ under the density basis $\{\omega_\mu(\mathbf{r})\}_{\mu=1}^M$ for representing the density defined by the orbital coefficient solution $\mathbf{C}^{(k)}$. This process is called *density fitting* [23], which is also used in KSDFT for acceleration, in which context the atomic basis for the density is also called *auxiliary basis*. The density coefficient $\mathbf{p}^{(k)}$ needs to fit represented density $\rho_{\mathbf{p}^{(k)}}$ by Eq. (25) to the density $\rho_{\mathbf{C}^{(k)}}$ defined by Eq. (28). Noting that Eq. (28) essentially expands the density onto the paired basis $\{\eta_\alpha(\mathbf{r})\eta_\beta(\mathbf{r})\}_{\alpha\beta}$ with coefficient as the vector $\bar{\mathbf{\Gamma}}^{(k)}$ of flattened $\mathbf{\Gamma}^{(k)}$, this is the classical coordinate transformation problem from the paired basis to the density basis. The classical approach is by minimizing the standard L2-norm of the residual density:

$$\int \left|\rho_{\mathbf{p}^{(k)}}(\mathbf{r}) - \rho_{\mathbf{C}^{(k)}}(\mathbf{r})\right|^2 \mathrm{d}\mathbf{r} = {\mathbf{p}^{(k)}}^\top \mathbf{W}\mathbf{p}^{(k)} - 2{\mathbf{p}^{(k)}}^\top \mathbf{L}\bar{\mathbf{\Gamma}}^{(k)} + {\bar{\mathbf{\Gamma}}^{(k)}}^\top \mathbf{D}\bar{\mathbf{\Gamma}}^{(k)},$$

where $\mathbf{W}_{\mu\nu} := \langle\omega_\mu|\omega_\nu\rangle$, $\mathbf{L}_{\mu,\alpha\beta} := \langle\omega_\mu|\eta_\alpha\eta_\beta\rangle$, and $\mathbf{D}_{\alpha\beta,\gamma\delta} := \langle\eta_\alpha\eta_\beta|\eta_\gamma\eta_\delta\rangle$ are the overlap matrices of the density basis, between the density basis and the paired basis, and of the paired basis, respectively. Noting that this is an quadratic form of $\mathbf{p}^{(k)}$, we know the solution is $\mathbf{p}^{(k)} = \mathbf{W}^{-1}\mathbf{L}\bar{\mathbf{\Gamma}}^{(k)}$.

However, the standard L2-metric on the density function may not be the most favorable metric for density fitting. Instead, energy is the directly concerned quantity. The kinetic energy is the most desired metric, since this would minimize the mismatch of the fitted density $\mathbf{p}^{(k)}$ to the kinetic energy label $T_\text{S}^{(k)}$. But there is no explicit expression to calculate the kinetic energy from density coefficient. We hence turn to using the Hartree energy and the external energy as the metric. (Using the XC energy requires an arbitrary choice of a functional approximation, and the calculation is more costly.)

For the Hartree energy as defined in Eq. (8), noting that it is quadratic in density, we fit $\mathbf{p}$ by minimizing the (2×) Hartree energy arising from the residual density:

$$2E_\text{H}[\rho_{\mathbf{p}^{(k)}} - \rho_{\mathbf{C}^{(k)}}] = \iint \frac{\left(\rho_{\mathbf{p}^{(k)}}(\mathbf{r}) - \rho_{\mathbf{C}^{(k)}}(\mathbf{r})\right)\left(\rho_{\mathbf{p}^{(k)}}(\mathbf{r}') - \rho_{\mathbf{C}^{(k)}}(\mathbf{r}')\right)}{\|\mathbf{r}-\mathbf{r}'\|} \mathrm{d}\mathbf{r}\mathrm{d}\mathbf{r}'$$
$$= {\mathbf{p}^{(k)}}^\top \tilde{\mathbf{W}}\mathbf{p}^{(k)} - 2{\mathbf{p}^{(k)}}^\top \tilde{\mathbf{L}}\bar{\mathbf{\Gamma}}^{(k)} + {\bar{\mathbf{\Gamma}}^{(k)}}^\top \tilde{\mathbf{D}}\bar{\mathbf{\Gamma}}^{(k)},$$

where symbols with tilde are the corresponding overlap matrices under integral kernel $\frac{1}{\|\mathbf{r}-\mathbf{r}'\|}$, which, using the symbol of Coulomb integral defined in Eq. (33), are $\tilde{\mathbf{W}}_{\mu\nu} := (\omega_\mu|\omega_\nu)$, $\tilde{\mathbf{L}}_{\mu,\alpha\beta} := (\omega_\mu|\eta_\alpha\eta_\beta)$, and $\tilde{\mathbf{D}}_{\alpha\beta,\gamma\delta} := (\eta_\alpha\eta_\beta|\eta_\gamma\eta_\delta)$ as already defined in Eq. (32). As a quadratic form, the solution is $\mathbf{p}^{(k)} = \tilde{\mathbf{W}}^{-1}\tilde{\mathbf{L}}\bar{\mathbf{\Gamma}}^{(k)}$. This result can be understood as if the Hartree energy (Coulomb integral) defines a metric on the space of density functions.

For the external energy as defined in Eq. (4), as it is linear in density, we fit $\mathbf{p}$ by directly minimizing the difference between the defined external energies:

$$\left(E_\text{ext}[\rho_{\mathbf{p}^{(k)}}] - E_\text{ext}[\rho_{\mathbf{C}^{(k)}}]\right)^2 = \left(E_\text{ext}(\mathbf{p}^{(k)}) - E_\text{ext}(\mathbf{C}^{(k)})\right)^2 \overset{\text{Eqs. (49, 38)}}{=} ({\mathbf{p}^{(k)}}^\top \mathbf{v}_\text{ext} - {\bar{\mathbf{\Gamma}}^{(k)}}^\top \bar{\mathbf{V}}_\text{ext})^2.$$

To combine the two metrics, the final optimization problem is a combined least squares problem:

$$\mathbf{p}^{(k)} = \operatorname*{argmin}_{\mathbf{p}} \mathbf{p}^\top \tilde{\mathbf{W}}\mathbf{p} - 2\mathbf{p}^\top \tilde{\mathbf{L}}\bar{\mathbf{\Gamma}}^{(k)} + {\bar{\mathbf{\Gamma}}^{(k)}}^\top \tilde{\mathbf{D}}\bar{\mathbf{\Gamma}}^{(k)} + (\mathbf{p}^\top \mathbf{v}_\text{ext} - {\bar{\mathbf{\Gamma}}^{(k)}}^\top \bar{\mathbf{V}}_\text{ext})^2,$$

which corresponds to the over-determined linear equations in matrix form:

$$\mathsf{W}\mathbf{p}^{(k)} = \mathsf{b}^{(k)}, \quad \text{where } \mathsf{W} := \begin{pmatrix} \tilde{\mathbf{W}} \\ \mathbf{v}_\text{ext}^\top \end{pmatrix}, \mathsf{b}^{(k)} := \begin{pmatrix} \tilde{\mathbf{L}}\bar{\mathbf{\Gamma}}^{(k)} \\ {\bar{\mathbf{\Gamma}}^{(k)}}^\top \bar{\mathbf{V}}_\text{ext} \end{pmatrix}.$$

This is directly solved using least-squares solvers. In this conversion, we did not explicitly consider the normalization constraint, ${\mathbf{p}^{(k)}}^\top \mathbf{w} = N$, since it is already satisfied with a high accuracy, due to the close fit to the original density.

Supplementary Section A.4.2 Value Label Calculation

The label for the value of KEDF can be calculated from Eq. (22) by leveraging the expression Eq. (35) under atomic basis:

$$T_\text{S}(\mathbf{C}^{(k)}) = \sum_{\alpha\beta} \mathbf{\Gamma}_{\alpha\beta}^{(k)} \mathbf{T}_{\alpha\beta} = {\bar{\mathbf{\Gamma}}^{(k)}}^\top \bar{\mathbf{T}},$$

where $\mathbf{\Gamma}^{(k)} = \mathbf{C}^{(k)}{\mathbf{C}^{(k)}}^\top$ from Eq. (29), and $\bar{\mathbf{\Gamma}}^{(k)}$ is the vector by flattening. The corresponding density coefficient $\mathbf{p}^{(k)}$ is calculated from $\mathbf{C}^{(k)}$ using density fitting as detailed above. A subtlety arises since the fitted density $\mathbf{p}^{(k)}$ may differ a little from the original density defined by $\mathbf{C}^{(k)}$ due to finite-basis incompleteness, so $T_\text{S}(\mathbf{C}^{(k)})$ may not be the best kinetic energy label for $\mathbf{p}^{(k)}$. Indeed, as mentioned, in density fitting, we do not have a way to directly find the density coefficient $\mathbf{p}^{(k)}$ that achieves the kinetic energy closest to $T_\text{S}(\mathbf{C}^{(k)})$. Instead, $\mathbf{p}^{(k)}$ is fitted to match the Hartree and external energy. We hence assume that the electronic energy is less affected by density-fitting error than the kinetic energy, and instead of taking $T_\text{S}(\mathbf{p}^{(k)}) \approx T_\text{S}(\mathbf{C}^{(k)})$, we take $E(\mathbf{p}^{(k)}) \approx E(\mathbf{C}^{(k)})$, which means $T_\text{S}(\mathbf{p}^{(k)}) + E_\text{eff}(\mathbf{p}^{(k)}) \approx T_\text{S}(\mathbf{C}^{(k)}) + E_\text{eff}(\mathbf{C}^{(k)})$. Hence the KEDF value label for $\mathbf{p}^{(k)}$ is taken as:

$$T_\text{S}^{(k)} := T_\text{S}(\mathbf{C}^{(k)}) + E_\text{eff}(\mathbf{C}^{(k)}) - E_\text{eff}(\mathbf{p}^{(k)}),$$

$$\text{where} \quad E_\text{eff}(\mathbf{C}^{(k)}) = \underbrace{\frac{1}{2}\bar{\mathbf{\Gamma}}^{(k)\top}\tilde{\mathbf{D}}\bar{\mathbf{\Gamma}}^{(k)}}_{E_\text{H}(\mathbf{C}^{(k)})} + E_\text{XC}(\mathbf{C}^{(k)}) + \underbrace{\bar{\mathbf{\Gamma}}^{(k)\top}\bar{\mathbf{V}}_\text{ext}}_{E_\text{ext}(\mathbf{C}^{(k)})},$$

$$E_\text{eff}(\mathbf{p}^{(k)}) = \underbrace{\frac{1}{2}\mathbf{p}^{(k)\top}\tilde{\mathbf{W}}\mathbf{p}^{(k)}}_{E_\text{H}(\mathbf{p}^{(k)})} + E_\text{XC}(\mathbf{p}^{(k)}) + \underbrace{\mathbf{p}^{(k)\top}\mathbf{v}_\text{ext}}_{E_\text{ext}(\mathbf{p}^{(k)})},$$

following definitions and expressions of Eqs. (34-38) and Eqs. (45-49). Labels for the two variants of functional models detailed in Supplementary Section B.4 can be calculated accordingly. For the residual version of KEDF $T_\text{S,res}$, the label is correspondingly modified using values at $\mathbf{p}^{(k)}$:

$$T_\text{S,res}^{(k)} := T_\text{S}^{(k)} - T_\text{APBE}(\mathbf{p}^{(k)}).$$

For the version E_TXC that learns the sum of KEDF and the XC energy, the corresponding label is: $T_\text{S}^{(k)} + E_\text{XC}(\mathbf{p}^{(k)}) = T_\text{S}(\mathbf{C}^{(k)}) + \big(E_\text{H}(\mathbf{C}^{(k)}) + E_\text{XC}(\mathbf{C}^{(k)}) + E_\text{ext}(\mathbf{C}^{(k)})\big) - \big(E_\text{H}(\mathbf{p}^{(k)}) + E_\text{XC}(\mathbf{p}^{(k)}) + E_\text{ext}(\mathbf{p}^{(k)})\big) + E_\text{XC}(\mathbf{p}^{(k)}) = T_\text{S}(\mathbf{C}^{(k)}) + E_\text{XC}(\mathbf{C}^{(k)}) + \big(E_\text{H}(\mathbf{C}^{(k)}) - E_\text{H}(\mathbf{p}^{(k)}) + E_\text{ext}(\mathbf{C}^{(k)}) - E_\text{ext}(\mathbf{p}^{(k)})\big)$, where the last term can be omitted since the Hartree energy difference and the external energy difference are minimized by $\mathbf{p}^{(k)}$ in density fitting. We therefore take the label as:

$$E_\text{TXC}^{(k)} := T_\text{S}(\mathbf{C}^{(k)}) + E_\text{XC}(\mathbf{C}^{(k)}).$$

Supplementary Section A.4.3 Gradient Label Calculation

Under an atomic basis, the kinetic energy functional of density is converted into a function of density coefficient $T_\text{S}(\mathbf{p}) := T_\text{S}[\rho_\mathbf{p}]$ following Eq. (46). For its gradient $\nabla_\mathbf{p} T_\text{S}(\mathbf{p})$, following the fact in Eq. (51), it is related to the variation of the functional $T_\text{S}[\rho]$ by integral with the basis:

$$\big(\nabla_\mathbf{p} T_\text{S}(\mathbf{p})\big)_\mu = \int \frac{\delta T_\text{S}[\rho_\mathbf{p}]}{\delta\rho}(\mathbf{r})\big(\nabla_\mathbf{p}\rho_\mathbf{p}(\mathbf{r})\big)_\mu \,\mathrm{d}\mathbf{r} = \int \frac{\delta T_\text{S}[\rho_\mathbf{p}]}{\delta\rho}(\mathbf{r})\omega_\mu(\mathbf{r})\,\mathrm{d}\mathbf{r}.$$

The variation corresponding to a known density is given by Eq. (23) above, which comes from the solution of orbitals in a KSDFT SCF iteration. If omitting the error in density fitting and approximating $\frac{\delta}{\delta\rho}T_\text{S}[\rho_{\mathbf{p}^{(k)}}]$ with $\frac{\delta}{\delta\rho}T_\text{S}[\rho_{\mathbf{C}^{(k)}}] = \frac{\delta}{\delta\rho}T_\text{S}[\rho_{[\Phi^{(k)}]}]$, then the gradient can be accessed by integrating both sides of Eq. (23) with the density basis:

$$\nabla_\mathbf{p} T_\text{S}(\mathbf{p}^{(k)}) + \mathbf{v}_\text{eff}^{(k)} = \mu^{(k)}\mathbf{w},$$

$$\text{where} \quad (\mathbf{v}_\text{eff}^{(k)})_\mu := \langle\omega_\mu|V_\text{eff}^{(k)}\rangle = \int V_\text{eff}^{(k)}(\mathbf{r})\omega_\mu(\mathbf{r})\,\mathrm{d}\mathbf{r}, \quad \mathbf{w}_\mu := \int \omega_\mu(\mathbf{r})\,\mathrm{d}\mathbf{r}. \tag{52}$$

In practice, the chemical potential $\mu^{(k)}$ is not needed, since the gradient matters in its projection on the tangent space of normalized densities in order to keep the density normalized in density optimization (see Eq. (2)). The space of normalized densities is a linear space $\{\mathbf{p} \mid \mathbf{p}^\top\mathbf{w} = N\}$ since $\int \rho_\mathbf{p}(\mathbf{r})\,\mathrm{d}\mathbf{r} = \sum_\mu \mathbf{p}_\mu \int \omega_\mu(\mathbf{r})\,\mathrm{d}\mathbf{r} = N$, so it coincides with its tangent space. The projection onto the tangent space is achieved by applying $\mathbf{I} - \frac{\mathbf{w}\mathbf{w}^\top}{\mathbf{w}^\top\mathbf{w}}$, which gives:

$$\Big(\mathbf{I} - \frac{\mathbf{w}\mathbf{w}^\top}{\mathbf{w}^\top\mathbf{w}}\Big)\nabla_\mathbf{p} T_\text{S}(\mathbf{p}^{(k)}) = -\Big(\mathbf{I} - \frac{\mathbf{w}\mathbf{w}^\top}{\mathbf{w}^\top\mathbf{w}}\Big)\mathbf{v}_\text{eff}^{(k)}. \tag{53}$$

Due to the same reason, the gradient loss function Eq. (4) also only matches the projected gradient of the model to the projected gradient label.

The remaining task is to evaluate $\mathbf{v}_\text{eff}^{(k)}$ in Eq. (52). Considering the complication that $V_\text{eff}^{(k)}(\mathbf{r})$ may not be taken as the explicit-form effective potential $V_{\text{eff}[\rho_{[\Phi^{(k-1)}]}]}(\mathbf{r}) = V_{\text{eff}[\rho_{\mathbf{C}^{(k-1)}}]}$ (Eq. (17) and Eq. (28)), such as when DIIS (Eq. (44)) is used in SCF iteration, evaluating $\mathbf{v}_{\text{eff}}^{(k)}$ can be done by leveraging the orbital-basis representation $\mathbf{V}_{\text{eff}}^{(k)}$ already available in the SCF problem (Eq. (26)), which is defined in Eq. (27) as the integral with paired orbital basis: $(\mathbf{V}_{\text{eff}}^{(k)})_{\alpha\beta} := \int V_{\text{eff}}^{(k)}(\mathbf{r})\eta_\alpha(\mathbf{r})\eta_\beta(\mathbf{r})\,\mathrm{d}\mathbf{r}$. Using the expansion coefficients $\mathbf{K}$ of the density basis onto the paired orbital basis, that is $\omega_\mu(\mathbf{r}) = \sum_{\alpha\beta} \mathbf{K}_{\mu,\alpha\beta}\eta_\alpha(\mathbf{r})\eta_\beta(\mathbf{r})$, we can conduct the conversion by $(\mathbf{v}_{\text{eff}}^{(k)})_\mu = \sum_{\alpha\beta} \mathbf{K}_{\mu,\alpha\beta}(\mathbf{V}_{\text{eff}}^{(k)})_{\alpha\beta}$.

However, solving for $\mathbf{K}$ is unaffordable: using least squares, this amounts to solving $\mathbf{D}\mathbf{K} = \mathbf{L}$ (or solving $\tilde{\mathbf{D}}\mathbf{K} = \tilde{\mathbf{L}}$). Since $\mathbf{D}$ (or $\tilde{\mathbf{D}}$) has shape $B^2 \times B^2$, the complexity is $O(MB^4) = O(N^5)$. Even it is only called once for one molecular structure, the cost is still intractable even for medium-sized molecules. Moreover, the approximation $\frac{\delta}{\delta\rho}T_{\text{S}}[\rho_{\mathbf{p}^{(k)}}] \approx \frac{\delta}{\delta\rho}T_{\text{S}}[\rho_{\mathbf{C}^{(k)}}]$ does not guarantee the optimality of density optimization using the learned KEDF model on the same molecular structure as explained later.

We hence turn to another approximation, and use a more direct way to calculate the gradient. The approximation is that Eq. (23) also holds for the fitted density $\rho_{\mathbf{p}^{(k)}}$:

$$\frac{\delta T_{\text{S}}[\rho_{\mathbf{p}^{(k)}}]}{\delta\rho} + V_{\text{eff}\{\mathbf{p}^{(k')}\}_{k'<k}} = \mu'^{(k)}, \tag{54}$$

where $V_{\text{eff}\{\mathbf{p}^{(k')}\}_{k'<k}}$ is the $V_{\text{eff}}^{(k)}$ constructed from fitted density coefficients in previous SCF iterations, instead of orbital coefficient solutions in previous SCF iterations that the $V_{\text{eff}}^{(k)}$ in Eq. (23) uses. The chemical potential $\mu'^{(k)}$ may be different, but due to the above argument for Eq. (53), it is not used. The approximation holds if density fitting error can be omitted, for example when using a large basis set. Following the procedure above, the corresponding kinetic-energy gradient after projection is given by:

$$\Big(\mathbf{I} - \frac{\mathbf{w}\mathbf{w}^\top}{\mathbf{w}^\top\mathbf{w}}\Big)\nabla_{\mathbf{p}}T_{\text{S}}(\mathbf{p}^{(k)}) = -\Big(\mathbf{I} - \frac{\mathbf{w}\mathbf{w}^\top}{\mathbf{w}^\top\mathbf{w}}\Big)\mathbf{v}_{\text{eff}\{\mathbf{p}^{(k')}\}_{k'<k}}, \tag{55}$$

$$\text{where} \quad \big(\mathbf{v}_{\text{eff}\{\mathbf{p}^{(k')}\}_{k'<k}}\big)_\mu := \langle \omega_\mu | V_{\text{eff}\{\mathbf{p}^{(k')}\}_{k'<k}} \rangle = \int V_{\text{eff}\{\mathbf{p}^{(k')}\}_{k'<k}}(\mathbf{r})\omega_\mu(\mathbf{r})\,\mathrm{d}\mathbf{r}, \tag{56}$$

correspondingly. Calculating $\mathbf{v}_{\text{eff}\{\mathbf{p}^{(k')}\}_{k'<k}}$ requires a direct approach. This can be done following the relation between the known functions $V_{\text{eff}[\rho_{[\Phi^{(k')}]}]}$ and the constructed $V_{\text{eff}}^{(k)}$ in the SCF iteration. In DIIS, this relation can be drawn from the construction in Eq. (44) by noting the definitions Eq. (27) and Eq. (30) of the matrices, which is a weighted average: $V_{\text{eff}}^{(k)} = \sum_{k'=0}^{k-1} \pi_{k'}^{(k)} V_{\text{eff}[\rho_{\mathbf{C}^{(k')}}]}$. Following this pattern, the required effective potential in Eq. (54) is constructed as: $V_{\text{eff}\{\mathbf{p}^{(k')}\}_{k'<k}} = \sum_{k'=0}^{k-1} \pi_{k'}^{(k)} V_{\text{eff}[\rho_{\mathbf{p}^{(k')}}]}$. The weights $\{\pi_{k'}^{(k)}\}_{k'=0}^{k-1}$ are taken as the same as those computed in the SCF iteration. Its vector form under the density basis is given by:

$$\mathbf{v}_{\text{eff}\{\mathbf{p}^{(k')}\}_{k'<k}} = \sum_{k'=0}^{k-1} \pi_{k'}^{(k)} \mathbf{v}_{\text{eff}_{\mathbf{p}^{(k')}}}, \quad \text{where } \big(\mathbf{v}_{\text{eff}_{\mathbf{p}}}\big)_\mu := \langle \omega_\mu | V_{\text{eff}[\rho_{\mathbf{p}}]} \rangle = \int V_{\text{eff}[\rho_{\mathbf{p}}]}(\mathbf{r})\omega_\mu(\mathbf{r})\,\mathrm{d}\mathbf{r}. \tag{57}$$

Calculation of each $\mathbf{v}_{\text{eff}_{\mathbf{p}^{(k)}}}$ can be carried out directly following Eq. (17) that gives $V_{\text{eff}[\rho_{\mathbf{p}}]}$ explicitly. In our implementation, each $\mathbf{v}_{\text{eff}_{\mathbf{p}^{(k)}}}$ is conveniently calculated using our automatic differentiation implementation mentioned in Supplementary Section A.3.2, since we notice the fact that:

$$\mathbf{v}_{\text{eff}_{\mathbf{p}}} = \nabla_{\mathbf{p}} E_{\text{eff}}(\mathbf{p}), \tag{58}$$

where $E_{\text{eff}}(\mathbf{p}) := E_{\text{eff}}[\rho_{\mathbf{p}}]$ is defined in Eq. (45) and its gradient $\nabla_{\mathbf{p}} E_{\text{eff}}(\mathbf{p})$ is given by Eq. (50). This fact is again due to the relation between gradient and variation revealed in Eq. (51) and noting that $V_{\text{eff}[\rho]}$ is defined as the variation $\frac{\delta E_{\text{eff}}[\rho]}{\delta\rho}$ of the effective energy functional in Eq. (17). As analyzed at the end of Supplementary Section A.3.2, evaluating the gradient has the same complexity as evaluating the energy $E_{\text{eff}}(\mathbf{p})$, which is $O(M^2) + O(MN_{\text{grid}}) = O(N^2)$, which is much lower than the $O(N^5)$ complexity above.

To sum up, $\{\mathbf{v}_{\text{eff}_{\mathbf{p}^{(k')}}}\}_{k'}$ are first calculated using automatic differentiation following Eq. (58), which are used to construct $\mathbf{v}_{\text{eff}\{\mathbf{p}^{(k')}\}_{k'<k}}$ following Eq. (57), then the gradient $\nabla_{\mathbf{p}} T_{\text{S}}(\mathbf{p}^{(k)})$ is given by Eq. (55) up to a projection. Since the loss function Eq. (4) for gradient supervision explicitly projects the gradient error, the gradient label itself does not have to be projected before evaluating the loss (that is, the loss is the same whether the gradient label is projected; since projection is idempotent). We hence take the gradient label $\nabla_{\mathbf{p}} T_{\text{S}}^{(k)}$ directly as the density-constructed DIIS effective potential vector:

$$\nabla_{\mathbf{p}} T_{\text{S}}^{(k)} := -\mathbf{v}_{\text{eff}\{\mathbf{p}^{(k')}\}_{k'<k}}.$$

For the residual version of KEDF $T_{\text{S,res}}$ and the version E_{TXC} that also includes XC energy as detailed in Supplementary Section B.4, the labels are produced accordingly:

$$\nabla_{\mathbf{p}} T_{\text{S,res}}^{(k)} := \nabla_{\mathbf{p}} T_{\text{S}}^{(k)} - \nabla_{\mathbf{p}} T_{\text{APBE}}(\mathbf{p}^{(k)}), \quad \nabla_{\mathbf{p}} E_{\text{TXC}}^{(k)} := \nabla_{\mathbf{p}} T_{\text{S}}^{(k)} + \nabla_{\mathbf{p}} E_{\text{XC}}(\mathbf{p}^{(k)}),$$

where $\nabla_{\mathbf{p}} T_{\text{APBE}}(\mathbf{p}^{(k)})$ and $\nabla_{\mathbf{p}} E_{\text{XC}}(\mathbf{p}^{(k)})$ are also calculated using automatic differentiation.

Apart from the convenient and efficient calculation using automatic differentiation, this choice of gradient label could also train a KEDF model that leads to the correct optimal density, since the labeling approach is compatible with the density optimization procedure of M-OFDFT shown in Eq. (2) and Eq. (50). More specifically, upon the convergence of SCF iteration for which we mark quantities with "$\star$", the DIIS effective potential $V_{\text{eff}}^{\star}$ is converged to the single-step, explicit-form effective potential $V_{\text{eff}[\rho_{\mathbf{C}^\star}]}$ (Eq. (17) and Eq. (28)) by design. Correspondingly, the density-constructed DIIS effective potential vector $\mathbf{v}_{\text{eff}\{\mathbf{p}^{(k')}\}_{k'<\star}}$ (Eq. (56)) at convergence coincides with $\mathbf{v}_{\text{eff}\mathbf{p}^\star}$ (Eq. (57)), which is $\nabla_{\mathbf{p}} E_{\text{eff}}(\mathbf{p}^\star)$ by Eq. (58). This gives a gradient label to the KEDF model through Eq. (55), which enforces the model to satisfy:

$$\left(\mathbf{I} - \frac{\mathbf{w}\mathbf{w}^\top}{\mathbf{w}^\top\mathbf{w}}\right)\left(\nabla_{\mathbf{p}} T_{\text{S},\theta}(\mathbf{p}^\star) + \nabla_{\mathbf{p}} E_{\text{eff}}(\mathbf{p}^\star)\right) \overset{\text{Eq. (50)}}{=} \left(\mathbf{I} - \frac{\mathbf{w}\mathbf{w}^\top}{\mathbf{w}^\top\mathbf{w}}\right)\nabla_{\mathbf{p}} E_\theta(\mathbf{p}^\star) = 0,$$

which in turn enforces the density optimization process Eq. (2) to converge to $\mathbf{p}^\star$, the true ground-state density coefficient. The optimality of density optimization using the learned KEDF model can then be expected.

Supplementary Section A.5 Force Calculation

The force experienced by atoms in a molecular structure is an important quantity as it is directly required for geometry optimization and molecular dynamics simulation. It is also used as a metric to evaluate the results of M-OFDFT (Results 2.2, Supplementary Section D.2.1). There are different ways to calculate the force in both KSDFT and OFDFT, and the results may differ. Here we describe two common ways for force calculation, which are the Hellmann-Feynman (HF) force [24, 25] and the analytical force [26]. Evaluation protocol using force for M-OFDFT and M-NNP/M-NNP-Den against KSDFT is detailed at the end.

Hellmann-Feynman Force Force is the negative gradient of the total energy of a molecule as a function $E_{\text{tot}}(\mathbf{X})$ of atom coordinates $\mathbf{X} = \{\mathbf{x}^{(a)}\}_{a=1}^{A}$ (molecular conformation). We omit the dependency on the molecular composition $\mathbf{Z}$ for brevity. The total energy $E_{\text{tot}}(\mathbf{X}) := E_{\mathbf{X}}^\star + E_{\text{nuc}}(\mathbf{X})$ comprises both the electronic energy $E_{\mathbf{X}}^\star$ in electronic ground state (including interaction with the nuclei), and also the energy from inter-nuclear interaction:

$$E_{\text{nuc}}(\mathbf{X}) := \frac{1}{2} \sum_{\substack{a,b=1,\cdots,A,\\ a\neq b}} \frac{Z^{(a)} Z^{(b)}}{\|\mathbf{x}^{(a)} - \mathbf{x}^{(b)}\|}, \tag{59}$$

which gives the inter-nuclear part of the force,

$$-\nabla_{\mathbf{x}^{(a)}} E_{\text{nuc}}(\mathbf{X}) = -Z^{(a)} \sum_{b:\neq a} Z^{(b)} \frac{\mathbf{x}^{(b)} - \mathbf{x}^{(a)}}{\|\mathbf{x}^{(b)} - \mathbf{x}^{(a)}\|^3}. \tag{60}$$

The electronic energy $E_{\mathbf{X}}^\star$ is the minimum after a variational optimization process for solving the electronic ground state of the molecule in conformation $\mathbf{X}$. In the most fundamental form, $E_{\mathbf{X}}^\star$ is determined by the variational problem on N-electron wavefunctions as shown in Eq. (1). The Hamiltonian operator therein $\hat{H}_{\mathbf{X}} = \hat{T} + \hat{V}_{\text{ee}} + \hat{V}_{\text{ext},\mathbf{X}}$ depends on the conformation $\mathbf{X}$ through $V_{\text{ext},\mathbf{X}}$, which is given in Eq. (2). The ground-state wavefunction $\psi_{\mathbf{X}}^\star$ and energy $E_{\mathbf{X}}^\star = \langle \psi_{\mathbf{X}}^\star | \hat{H}_{\mathbf{X}} | \psi_{\mathbf{X}}^\star \rangle$ hence also depend on $\mathbf{X}$. The gradient of $E_{\mathbf{X}}^\star$ can then be reformed as: $\nabla_{\mathbf{X}} E_{\mathbf{X}}^\star =$

$$\begin{aligned}
\nabla_{\mathbf{X}} \langle \psi_{\mathbf{X}}^\star | \hat{H}_{\mathbf{X}} | \psi_{\mathbf{X}}^\star \rangle &= \langle \nabla_{\mathbf{X}} \psi_{\mathbf{X}}^\star | \hat{H}_{\mathbf{X}} | \psi_{\mathbf{X}}^\star \rangle + \langle \psi_{\mathbf{X}}^\star | \nabla_{\mathbf{X}} \hat{H}_{\mathbf{X}} | \psi_{\mathbf{X}}^\star \rangle + \langle \psi_{\mathbf{X}}^\star | \hat{H}_{\mathbf{X}} | \nabla_{\mathbf{X}} \psi_{\mathbf{X}}^\star \rangle \\
&\overset{(*)}{=} E_{\mathbf{X}}^\star \langle \nabla_{\mathbf{X}} \psi_{\mathbf{X}}^\star | \psi_{\mathbf{X}}^\star \rangle + \langle \psi_{\mathbf{X}}^\star | \nabla_{\mathbf{X}} \hat{H}_{\mathbf{X}} | \psi_{\mathbf{X}}^\star \rangle + E_{\mathbf{X}}^\star \langle \psi_{\mathbf{X}}^\star | \nabla_{\mathbf{X}} \psi_{\mathbf{X}}^\star \rangle \\
&= \langle \psi_{\mathbf{X}}^\star | \nabla_{\mathbf{X}} \hat{H}_{\mathbf{X}} | \psi_{\mathbf{X}}^\star \rangle + E_{\mathbf{X}}^\star \nabla_{\mathbf{X}} \langle \psi_{\mathbf{X}}^\star | \psi_{\mathbf{X}}^\star \rangle \\
&\overset{(\#)}{=} \langle \psi_{\mathbf{X}}^\star | \nabla_{\mathbf{X}} \hat{H}_{\mathbf{X}} | \psi_{\mathbf{X}}^\star \rangle,
\end{aligned} \tag{61}$$

where the equality (*) is due to that $\psi_{\mathbf{X}}^\star$ is an eigenstate of the Hermitian operator $\hat{H}_{\mathbf{X}}$ with real eigenvalue $E_{\mathbf{X}}^\star$, and the equality (#) is due to that the wavefunction is normalized $\langle \psi_{\mathbf{X}}^\star | \psi_{\mathbf{X}}^\star \rangle = 1$ for all $\mathbf{X}$.

Eq. (61) is the Hellmann-Feynman (HF) theorem [24, 25]. To continue the calculation, the gradient of the Hamiltonian operator in the expression can be derived as $\nabla_{\mathbf{x}^{(a)}} \hat{H}_{\mathbf{X}} = \nabla_{\mathbf{x}^{(a)}} \hat{V}_{\text{ext},\mathbf{X}}$, and by noting that $\hat{V}_{\text{ext},\mathbf{X}}$ is multiplicative and one-body as shown in Eq. (2), we have $\nabla_{\mathbf{x}^{(a)}} V_{\text{ext},\mathbf{X}}(\mathbf{r}) = -Z^{(a)} \frac{\mathbf{r}-\mathbf{x}^{(a)}}{\|\mathbf{r}-\mathbf{x}^{(a)}\|^3}$, and subsequently, the electronic force can be calculated as:

$$-\nabla_{\mathbf{x}^{(a)}} E^\star_{\mathbf{X}} = -\langle \psi^\star_{\mathbf{X}} | \nabla_{\mathbf{x}^{(a)}} \hat{H}_{\mathbf{X}} | \psi^\star_{\mathbf{X}} \rangle = Z^{(a)} \int \rho^\star_{\mathbf{X}}(\mathbf{r}) \frac{\mathbf{r}-\mathbf{x}^{(a)}}{\|\mathbf{r}-\mathbf{x}^{(a)}\|^3} \,\mathrm{d}\mathbf{r} =: \mathbf{f}^{(a)}_{\mathbf{X}}, \tag{62}$$

where $\rho^\star_{\mathbf{X}}(\mathbf{r}) := \rho_{[\psi^\star_{\mathbf{X}}]}(\mathbf{r})$ defined in Eq. (3). This is the *Hellmann-Feynman (HF) force* $\mathbf{f}_{\mathbf{X}}$. This expression coincides with the electrostatic force under a classical view, indicating "there are no 'mysterious quantum-mechanical forces' acting in molecules" [9]. From the expression, evaluating the HF force only requires a good approximation to the ground-state electron density $\rho^\star_{\mathbf{X}}(\mathbf{r})$, which is available in various quantum chemistry methods, including $\rho_{\mathbf{C}^\star_{\mathbf{X}}}$ in KSDFT given by Eq. (28) and $\rho_{\mathbf{p}^\star_{\mathbf{X}}}$ in M-OFDFT given by Eq. (25). The total force on the nuclei is the sum with the inter-nuclear force in Eq. (60).

Analytical Force However, when using the atomic basis, there emerges approximation error in the function representation, since the basis is incomplete. Consequently, the conditions of the HF theorem do not hold exactly. This makes the HF force in Eq. (62) only an approximation to the true electronic force $-\nabla_{\mathbf{x}^{(a)}} E^\star_{\mathbf{X}}$, and other approximations are possible. For example, $-\nabla_{\mathbf{x}^{(a)}} E^\star_{\mathbf{X}}$ can also be estimated by directly taking the analytical gradient of the electronic energy $E^\star_{\mathbf{X}}$ expressed under the atomic basis with the optimal coefficients [26]. This way of estimating the electronic force is hence called *analytical force* $\mathbf{f}_{\text{ana},\mathbf{X}}$. For KSDFT, $E^\star_{\mathbf{X}} = E_{\mathbf{X}}(\mathbf{C}^\star_{\mathbf{X}})$, where $E_{\mathbf{X}}(\mathbf{C})$ is given by Eqs. (34-38) (note that the basis functions η_α, η_β and matrices $\mathbf{T}$, $\tilde{\mathbf{D}}$ and $\mathbf{V}_{\text{ext}}$ all depend on $\mathbf{X}$), and $\mathbf{C}^\star_{\mathbf{X}}$ is the optimal orbital coefficients. The corresponding analytical force on an atom a can be expanded as:

$$\mathbf{f}^{(a)}_{\text{ana},\mathbf{X}} := -\nabla_{\mathbf{x}^{(a)}} E_{\mathbf{X}}(\mathbf{C}^\star_{\mathbf{X}}) = -(\nabla_{\mathbf{x}^{(a)}} E_{\mathbf{X}})(\mathbf{C}^\star_{\mathbf{X}}) - \operatorname{tr}\left((\nabla_{\mathbf{x}^{(a)}} \mathbf{C}^\star_{\mathbf{X}})^\top \nabla_{\mathbf{C}} E_{\mathbf{X}}(\mathbf{C}) \Big|_{\mathbf{C}=\mathbf{C}^\star_{\mathbf{X}}} \right), \tag{63}$$

where $(\nabla_{\mathbf{x}^{(a)}} E_{\mathbf{X}})$ takes the gradient with a fixed $\mathbf{C}$, and $\nabla_{\mathbf{x}^{(a)}} \mathbf{C}^\star_{\mathbf{X}}$ is the Jacobian, $(\nabla_{\mathbf{x}^{(a)}} \mathbf{C}^\star_{\mathbf{X}})_{\alpha i,\xi} := \partial (\mathbf{C}^\star_{\mathbf{X}})_{\alpha i} / \partial \mathbf{x}^{(a)}_\xi$ in which $\xi \in \{1,2,3\}$ indices the three spacial components, and matrix operations, including transpose, matrix multiplication and trace, act on indices α and i. When $\mathbf{C}^\star_{\mathbf{X}}$ is indeed accurately optimized, self-consistency in Eq. (43) and orthonormality in Eq. (39) are satisfied. By also noting Eq. (42), the second term in Eq. (63) vanishes: $\operatorname{tr}\left((\nabla_{\mathbf{x}^{(a)}} \mathbf{C}^\star_{\mathbf{X}})^\top \nabla_{\mathbf{C}} E_{\mathbf{X}}(\mathbf{C}) \big|_{\mathbf{C}=\mathbf{C}^\star_{\mathbf{X}}} \right) \overset{\text{Eq. (42)}}{=} 2 \operatorname{tr}\left((\nabla_{\mathbf{x}^{(a)}} \mathbf{C}^\star_{\mathbf{X}})^\top \mathbf{F}_{\mathbf{C}^\star_{\mathbf{X}}} \mathbf{C}^\star_{\mathbf{X}} \right) \overset{\text{Eq. (43)}}{=} 2 \operatorname{tr}\left((\nabla_{\mathbf{x}^{(a)}} \mathbf{C}^\star_{\mathbf{X}})^\top \mathbf{S}_{\mathbf{X}} \mathbf{C}^\star_{\mathbf{X}} \boldsymbol{\varepsilon}^\star_{\mathbf{X}} \right) = \operatorname{tr}\left((\nabla_{\mathbf{x}^{(a)}} \mathbf{C}^\star_{\mathbf{X}})^\top \nabla_{\mathbf{C}^\star_{\mathbf{X}}} \operatorname{tr}\left((\mathbf{C}^{\star\top}_{\mathbf{X}} \mathbf{S}_{\mathbf{X}} \mathbf{C}^\star_{\mathbf{X}} - \mathbf{I}) \boldsymbol{\varepsilon}^\star_{\mathbf{X}} \right) \right) = \nabla_{\mathbf{x}^{(a)}} \operatorname{tr}\left((\mathbf{C}^{\star\top}_{\mathbf{X}} \mathbf{S}_{\mathbf{X}} \mathbf{C}^\star_{\mathbf{X}} - \mathbf{I}) \boldsymbol{\varepsilon}^\star_{\mathbf{X}} \right) \overset{\text{Eq. (39)}}{=} 0$. If in practice $\mathbf{C}^\star_{\mathbf{X}}$ is not exactly optimized, the contribution from $\nabla_{\mathbf{x}^{(a)}} \mathbf{C}^\star_{\mathbf{X}}$ should also be considered; see Pulay [26] for detailed treatments.

Note that only the $-(\nabla_{\mathbf{x}^{(a)}} \bar{\mathbf{V}}_{\text{ext},\mathbf{X}})^\top \bar{\mathbf{\Gamma}}^\star_{\mathbf{X}}$ term ($\bar{\mathbf{\Gamma}}^\star_{\mathbf{X}}$ is the vector of flattened optimal density matrix $\mathbf{\Gamma}^\star_{\mathbf{X}} := \mathbf{C}^\star_{\mathbf{X}} \mathbf{C}^{\star\top}_{\mathbf{X}}$), as a part of $-(\nabla_{\mathbf{x}^{(a)}} E_{\mathbf{X}})(\mathbf{C}^\star_{\mathbf{X}})$, corresponds to the HF force (see Eq. (38)). Other terms in the analytical force $\mathbf{f}^{(a)}_{\text{ana},\mathbf{X}}$ in Eq. (63) are collectively called the Pulay force after [26], that is, $\mathbf{f}_{\text{ana},\mathbf{X}} - \mathbf{f}_{\mathbf{X}}$. They come in the form of the gradient of the basis functions with respect to atom coordinates, hence are non-zero when using atomic basis and differentiate the analytical force from the HF force.

Implementation Although the analytical force is regarded as a more accurate estimation when using atomic basis, we still take the HF force to evaluate the results, since the way to calculate the analytical force is different for different methods: KSDFT and M-OFDFT require different types of Coulomb integrals under different basis sets, and M-NNP and M-NNP-Den only require back-propagating the gradient through the deep-learning model. Moreover, the HF force in Eq. (62) only depends on the density from the ground-state solution, so it can also be seen as a relevant metric to evaluate the solved density.

For each molecular system, we take the true value of HF force as that given by the KSDFT solution, where the density is taken after density fitting (see Supplementary Section A.4.1). The HF force by M-OFDFT is calculated directly from the optimized density. For M-NNP and M-NNP-Den, as they cannot provide the density, only the corresponding analytical forces, $-\nabla_{\mathbf{X}} E_{\text{tot},\theta}(\mathbf{X})$ and $-\nabla_{\mathbf{X}} E_{\text{tot},\theta}(\mathbf{X}, \mathbf{p}^{\text{init}})$, are available. Note that since they predict the total energy, the inter-nuclear force $-\nabla_{\mathbf{X}} E_{\text{nuc}}(\mathbf{X})$ (Eq. (60)) is included in the gradients. To convert them into HF forces, we extract from the gradients with the inter-nuclear force as well as the Pulay force: $\mathbf{f}^{(a)}_{\text{M-NNP},\mathbf{X}} = -\nabla_{\mathbf{x}^{(a)}} E_{\text{tot},\theta}(\mathbf{X}) - (-\nabla_{\mathbf{x}^{(a)}} E_{\text{nuc}}(\mathbf{X})) - (\mathbf{f}^{(a)}_{\text{ana},\mathbf{X}} - \mathbf{f}^{(a)}_{\mathbf{X}})$, and similarly for $\mathbf{f}^{(a)}_{\text{M-NNP-Den},\mathbf{X}}$, where the analytical force $\mathbf{f}^{(a)}_{\text{ana},\mathbf{X}}$ and the HF force $\mathbf{f}^{(a)}_{\mathbf{X}}$ are calculated from KSDFT. Following previous works [27, 28], the error in predicted force is measured by the mean absolute error (MAE) over each of the three spacial components of the force on each atom in each molecule in the test set.

Supplementary Section A.6 Scaling Property under Atomic Basis

The KEDF has an exact scaling property which describes its change after uniformly scaling (stretching or squeezing) a density. Under a scaling (squeezing) rate λ, the uniformly scaled density is given by the rule of change of variables: $\hat{\lambda}\rho(\mathbf{r}) := \lambda^3 \rho(\lambda \mathbf{r})$. The scaling property is stated as the following [29]:

$$T_{\mathrm{S}}[\hat{\lambda}\rho] = \lambda^2 T_{\mathrm{S}}[\rho]. \tag{64}$$

Designing the model to exactly satisfy this condition not only guarantees reasonable results in some physical sense, but would also reduce the functional space where the model needs to search by learning, hence improving accuracy and the generalization and extrapolation ability. Some prior investigations [30, 31] for machine-learning KEDF indeed observed accuracy improvement in some cases (though not as substantial in some other cases).

Now we consider leveraging this property for the KEDF model $T_{\mathrm{S}}(\mathbf{p}, \mathcal{M})$ under atomic basis. As input density is represented under an atomic basis $\{\omega_\mu\}_{\mu=1}^M$ using coefficient $\mathbf{p}$ as $\rho_{\mathbf{p}}$ given by Eq. (25), we need to first represent the scaled density $\hat{\lambda}\rho_{\mathbf{p}}$ under the same basis with coefficient $\hat{\lambda}(\mathbf{p})$:

$$\hat{\lambda}\rho_{\mathbf{p}} := \sum_\mu \mathbf{p}_\mu \hat{\lambda}\omega_\mu(\mathbf{r}) = \sum_\mu \hat{\lambda}(\mathbf{p})_\mu \omega_\mu(\mathbf{r}). \tag{65}$$

Solving for $\hat{\lambda}(\mathbf{p})$ using least squares gives $\hat{\lambda}(\mathbf{p}) = \mathbf{W}^{-1}\mathbf{W}^{(\lambda)}\mathbf{p}$, where $\mathbf{W}^{(\lambda)}_{\mu\nu} := \langle \omega_\mu | \hat{\lambda}\omega_\nu \rangle$ here, and $\mathbf{W}_{\mu\nu} := \langle \omega_\mu | \omega_\nu \rangle$ as the same as above. The scaling property Eq. (64) is then transformed as:

$$T_{\mathrm{S}}(\mathbf{W}^{-1}\mathbf{W}^{(\lambda)}\mathbf{p}, \mathcal{M}) = \lambda^2 T_{\mathrm{S}}(\mathbf{p}, \mathcal{M}). \tag{66}$$

However, to preserve Eq. (64) exactly, the expansion Eq. (65) must hold exactly. But this is not the case: the finite basis set $\{\omega_\mu\}_{\mu=1}^M$ is incomplete, and the scaled basis functions $\{\hat{\lambda}\omega_\mu\}_{\mu=1}^M$ are not linearly dependent on the original ones. Hence, the transformed scaling property Eq. (66) under the atomic basis is *not exact*, thus may not provide much benefit to the KEDF model $T_{\mathrm{S},\theta}(\mathbf{p}, \mathcal{M})$.

There seems to be a possibility when using the even-tempered atomic basis set, which comes in the following form:

$$\begin{aligned} &\omega_{\mu=(a,\tau,\boldsymbol{\xi})}(\mathbf{r}) = w_{a,\tau,\boldsymbol{\xi}}(\mathbf{r} - \mathbf{x}^{(a)}), \\ \text{where} \quad &w_{a,\tau,\boldsymbol{\xi}}(\mathbf{r} = (x,y,z)) := x^{\boldsymbol{\xi}_1} y^{\boldsymbol{\xi}_2} z^{\boldsymbol{\xi}_3} \exp(-\alpha_{a,|\boldsymbol{\xi}|}\beta^\tau \|\mathbf{r}\|^2), \end{aligned} \tag{67}$$

with tempering ratio $\beta > 1$, monomial-exponent parameter $\boldsymbol{\xi} = (\boldsymbol{\xi}_1, \boldsymbol{\xi}_2, \boldsymbol{\xi}_3)$, and exponent parameter $\alpha_{a,|\boldsymbol{\xi}|}$ shared across $\boldsymbol{\xi}$ values with the same $|\boldsymbol{\xi}| := \boldsymbol{\xi}_1 + \boldsymbol{\xi}_2 + \boldsymbol{\xi}_3$. The index τ runs from 0 to $\mathcal{T}_{a,|\boldsymbol{\xi}|}$. Therefore, under the scaling with $\beta^{-\frac{1}{2}}$, we have: $\widehat{\beta^{-\frac{1}{2}}}\omega_{\mu=(a,\tau,\boldsymbol{\xi})}(\mathbf{r}) = \widehat{\beta^{-\frac{1}{2}}} w_{a,\tau,\boldsymbol{\xi}}(\mathbf{r} - \mathbf{x}^{(a)}) = \beta^{-\frac{3}{2}} w_{a,\tau,\boldsymbol{\xi}}(\beta^{-\frac{1}{2}}\mathbf{r} - \mathbf{x}^{(a)}) = (\beta^{-\frac{1}{2}})^{3+|\boldsymbol{\xi}|} w_{a,\tau-1,\boldsymbol{\xi}}(\mathbf{r} - \mathbf{x}^{(a)}/\beta^{-\frac{1}{2}})$, which is in the same even-tempered basis set but on a scaled molecular structure $\mathcal{M}/\beta^{-\frac{1}{2}} := \{\mathbf{Z}, \mathbf{X}/\beta^{-\frac{1}{2}}\}$, as long as $\tau > 0$. For $\tau = 0$, it corresponds to the most flat basis function, and its coefficient $\mathbf{p}_{\mu=(a,\tau=0,\boldsymbol{\xi})}$ is close to zero hence can be omitted for density representation. Therefore, under the scaling with $\beta^{-\frac{1}{2}}$, the scaled density can be expressed as:

$$\begin{aligned} \widehat{\beta^{-\frac{1}{2}}}\rho_{\mathbf{p},\mathcal{M}}(\mathbf{r}) &= \sum_{a,\boldsymbol{\xi}} \sum_{\tau=0}^{\mathcal{T}_{a,|\boldsymbol{\xi}|}} \mathbf{p}_{a,\tau,\boldsymbol{\xi}} \widehat{\beta^{-\frac{1}{2}}} \omega_{a,\tau,\boldsymbol{\xi}}(\mathbf{r}) \\ &= \sum_{a,\boldsymbol{\xi}} \sum_{\tau=1}^{\mathcal{T}_{a,|\boldsymbol{\xi}|}} \mathbf{p}_{a,\tau,\boldsymbol{\xi}} (\beta^{-\frac{1}{2}})^{3+|\boldsymbol{\xi}|} w_{a,\tau-1,\boldsymbol{\xi}}(\mathbf{r} - \mathbf{x}^{(a)}/\beta^{-\frac{1}{2}}) + \sum_{a,\boldsymbol{\xi}} \mathbf{p}_{a,0,\boldsymbol{\xi}} \widehat{\beta^{-\frac{1}{2}}} \omega_{a,0,\boldsymbol{\xi}}(\mathbf{r}) \\ &\approx \sum_{a,\boldsymbol{\xi}} \sum_{\tau=1}^{\mathcal{T}_{a,|\boldsymbol{\xi}|}} \mathbf{p}_{a,\tau,\boldsymbol{\xi}} (\beta^{-\frac{1}{2}})^{3+|\boldsymbol{\xi}|} w_{a,\tau-1,\boldsymbol{\xi}}(\mathbf{r} - \mathbf{x}^{(a)}/\beta^{-\frac{1}{2}}) \\ &=: \sum_{a,\boldsymbol{\xi}} \sum_{\tau=0}^{\mathcal{T}_{a,|\boldsymbol{\xi}|}} \mathbf{p}^{(\beta^{-\frac{1}{2}})}_{a,\tau,\boldsymbol{\xi}} w_{a,\tau,\boldsymbol{\xi}}(\mathbf{r} - \mathbf{x}^{(a)}/\beta^{-\frac{1}{2}}) = \rho_{\mathbf{p}^{(\beta^{-\frac{1}{2}})}, \mathcal{M}/\beta^{-\frac{1}{2}}}(\mathbf{r}), \end{aligned}$$

where the new coefficients are defined as:

$$\mathbf{p}^{(\beta^{-\frac{1}{2}})}_{a,\tau,\boldsymbol{\xi}} := \begin{cases} (\beta^{-\frac{1}{2}})^{3+|\boldsymbol{\xi}|} \mathbf{p}_{a,\tau+1,\boldsymbol{\xi}}, & \tau < \mathcal{T}_{a,|\boldsymbol{\xi}|}, \\ 0, & \tau = \mathcal{T}_{a,|\boldsymbol{\xi}|}. \end{cases}$$

The scaling property Eq. (64) can then be written as:

$$T_S(\mathbf{p}^{(\beta^{-\frac{1}{2}})}, \mathcal{M}/\beta^{-\frac{1}{2}}) = (\beta^{-\frac{1}{2}})^2 T_S(\mathbf{p}, \mathcal{M}).$$

However, this holds only for one value of scaling depending on the basis set (also for $(\beta^{-\frac{1}{2}})^n$ for integer n as long as the contributions from the first n coefficients can be omitted). Moreover, this constraint connects the original input to a *scaled conformation* $\mathcal{M}/\beta^{-\frac{1}{2}}$, which can be far from an equilibrium structure. The behavior of the $T_{S,\theta}$ model for these scaled conformations may be less relevant for real applications. Hence still, it does not seem definite to gain benefits from the scaling property.

Another possibility to exactly expressing the scaling property is to replace the molecular structure $\mathcal{M}$ with the basis overlap matrix $\mathbf{W}$ to characterize the atomic basis. In this way, the rescaling of atomic basis can be described by the change of the overlap matrix:

$$\hat{\lambda}\mathbf{W}_{\mu\nu} := \int \hat{\lambda}\omega_\mu(\mathbf{r})\, \hat{\lambda}\omega_\nu(\mathbf{r})\, \mathrm{d}\mathbf{r} = \lambda^3 \int \omega_\mu(\mathbf{r})\omega_\nu(\mathbf{r})\, \mathrm{d}(\lambda\mathbf{r}) = \lambda^3 \mathbf{W}_{\mu\nu}.$$

Hence, we do not need to expand the rescaled basis onto the original one, and the scaling property in Eq. (64) becomes:

$$T_S(\mathbf{p}, \lambda^3\mathbf{W}) = \lambda^2 T_S(\mathbf{p}, \mathbf{W}).$$

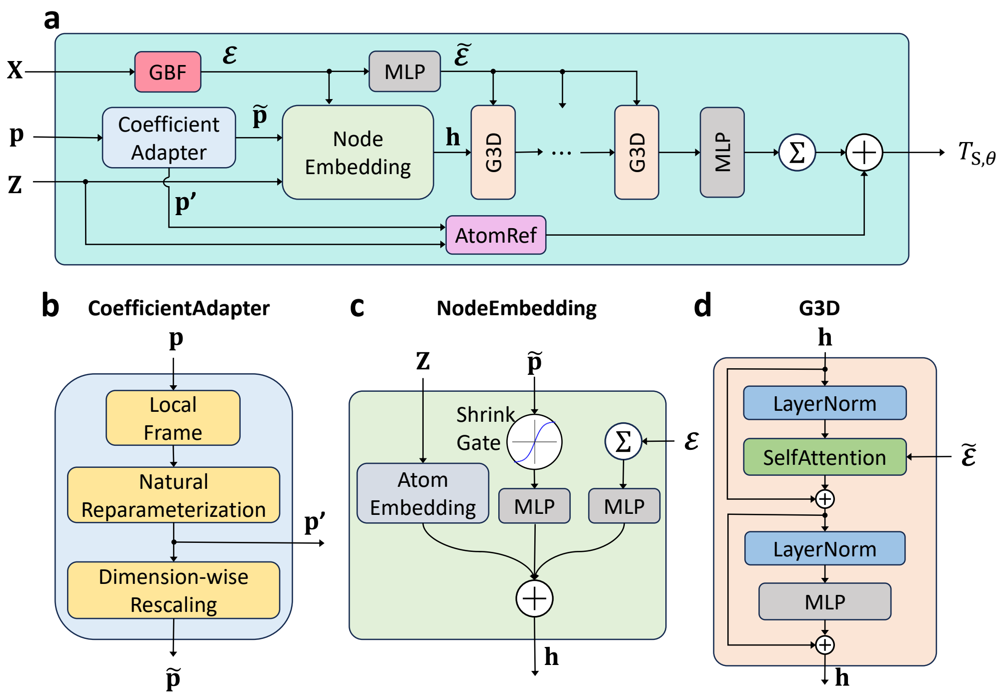
It is promising future work to investigate whether the overlap matrix is sufficient to specify the spacial and type configurations of the basis functions, and to design proper model architecture for effectively processing such pairwise feature and ensuring the above scaling property.

Supplementary Section B M-OFDFT Technical Details

In this section, we provide more technical details of the proposed M-OFDFT framework. We first provide an overall description of the nonlocal neural network model for KEDF and detail the architecture of neural network components in Supplementary Section B.1. We then elaborate on other components in the model that handle the unconventional challenges for learning a neural network model, including the use of local frame to guarantee geometric invariance of the model in Supplementary Section B.2, and enhancement modules for expressing a large gradient range in Supplementary Section B.3. Furthermore, we discuss explorations on learning different density functional variants in Supplementary Section B.4. Finally, we detail the density optimization techniques in Supplementary Section B.5, which is the usage of M-OFDFT to solve a given molecular structure.

Supplementary Section B.1 Model Specification

We start with more explanations on the design idea of the model formulation. As motivated in the main paper, M-OFDFT utilizes the expansion coefficients $\mathbf{p}$ of the density under an atomic basis set as the direct density feature input into the functional model. Compared to the grid-based representation, a common alternative for density representation, using the atomic-basis representation saves thousands of times the representation dimension. This is critical for implementing a nonlocal calculation in the model, which is well-known vital for approximating KEDF [32–36], but would otherwise be prohibited by the millions times more cost. The aggregation to atoms further reduces the required number of interacting nodes (from grids to atoms) in the nonlocal calculation. Noting that atomic basis are also used in material systems, so this way of density representation hence M-OFDFT is not restricted to non-periodic molecular systems in formulation. We note that there are also works in the context of learning the XC functional that develop machine-learning models also using features under atomic basis [37, 38], but the models do not take the molecular structure in input and focus on processing the features hosted on each atom individually, hence are essentially not nonlocal models.



Supplementary Figure 1: The KEDF model architecture. **(a)** (See also Alg. 1.) Overview of the model architecture. The model calculates the non-interacting kinetic energy (or a variant of it) from the given density, specified by the density coefficient vector $\mathbf{p}$ on an atomic basis set (Supplementary Section B.1.1), as well as the atomic numbers $\mathbf{Z}$ and coordinates $\mathbf{X}$ of all atoms in the target molecule for characterizing the basis functions. The coefficient vector $\mathbf{p}$ is first processed by the *CoefficientAdapter* module to make an SE(3)-invariant density features $\tilde{\mathbf{p}}$ and reduce the gradient scale for the rest of the model to fit. The $\tilde{\mathbf{p}}$ features are then used to construct initial node features $\mathbf{h}$ by the *NodeEmbedding* module (Supplementary Section B.1.3), for which the types $\mathbf{Z}$ and positions $\mathbf{X}$ of the atoms are also incorporated to provide information to interpret the coefficient features. The positional input $\mathbf{X}$ is perceived by the model only in terms of pairwise distance features $\mathcal{E}$ produced by the Gaussian basis function (*GBF*) module (Supplementary Section B.1.2). The node features $\mathbf{h}$ are subsequently updated by several Graphormer-3D (*G3D*) modules (Supplementary Section B.1.4). They calculate the interaction of features on every pair of nodes hence cover the nonlocal effect, in which relative position features $\tilde{\mathcal{E}}$ between each pair of nodes are considered, which are processed from $\mathcal{E}$ by a multi-layer perceptron (*MLP*) module (Eq. (69)). The final node features are aggregated by another MLP module and summed over the nodes to produce a scalar, and the final output is its addition with the output of the *AtomRef* enhancement module (Methods 4.3, Supplementary Section B.3.3) which shares the burden to model a large-scale gradient. **(b)** (See also Alg. 2.) Structure of the *CoefficientAdapter* module. It consists of the *LocalFrame* module (Methods 4.2; Supplementary Section B.2) to convert SE(3)-equivariant features into invariant features, followed by two enhancement modules *NaturalReparameterization* and *DimensionwiseRescaling* (Methods 4.3; Supplementary Section B.3.2 and B.3.1) which reduce the gradient scale for subsequent modules to fit. **(c)** Structure of the *NodeEmbedding* module (Supplementary Section B.1.3). It integrates information from three sources for each node: atom-type (for basis-set type) $\mathbf{Z}$ which is mapped to a feature vector by atom embedding, positional feature in terms of pairwise distance feature $\mathcal{E}$ which is aggregated over nodes and processed by an MLP module, and density features $\tilde{\mathbf{p}}$ which is mapped to a mild numerical range by a shrink gate and processed by another MLP module. **(d)** Structure of the *G3D* module (Supplementary Section B.1.4). The *SelfAttention* module updates node features by the nonlocal cross-node interaction of features $\mathbf{h}$ based on spacial relation features $\tilde{\mathcal{E}}$ between nodes. An MLP module further processes the updated node features. The *LayerNorm* module is applied before each of the two modules for numerical convenience. Note that the data on each streamline is the concatenated features across the nodes. Only the *SelfAttention* module models the interaction across nodes, while other modules are applied to the features of each node independently.

Since the atomic basis depends on the molecular structure $\mathcal{M} := \{\mathbf{Z}, \mathbf{X}\}$, the model also needs to include $\mathcal{M}$ into its input for a complete specification of the input density, where the atom types $\mathbf{Z} := \{Z^{(a)}\}_{a=1}^{A}$ are used to specify the types of basis functions (since atoms of different elements are assigned with dif-

Algorithm 1 Evaluation of the KEDF model $T_{\mathrm{S},\theta}(\mathbf{p}, \mathcal{M})$ (or the kinetic residual model $T_{\mathrm{S,res},\theta}(\mathbf{p}, \mathcal{M})$ or the TXC model $E_{\mathrm{TXC},\theta}(\mathbf{p}, \mathcal{M})$; see also Supplementary Figure 1)

```
Require: Input molecular structure $\mathcal{M} = \{\mathbf{X}, \mathbf{Z}\}$ comprising positions $\mathbf{X} := \{\mathbf{x}^{(a)}\}_{a=1}^{A}$ and atomic
    numbers $\mathbf{Z} := \{Z^{(a)}\}_{a=1}^{A}$ of all atoms in the molecule, input density coefficients $\mathbf{p}$ (see Supplemen-
    tary Section B.1.1).
1: Construct pairwise distance features $\boldsymbol{\mathcal{E}} \leftarrow \texttt{GBF}(\mathbf{X})$ and $\tilde{\boldsymbol{\mathcal{E}}} \leftarrow \texttt{MLP}(\boldsymbol{\mathcal{E}})$ (Eq. (68), Eq. (72), Eq. (69));
2: Process coefficient features: $(\tilde{\mathbf{p}}, \mathbf{p}') \leftarrow \texttt{CoefficientAdapter}(\mathbf{p})$ (Alg. 2);
3: Construct initial atomic representations: $\mathbf{h} \leftarrow \texttt{NodeEmbedding}(\mathbf{Z}, \boldsymbol{\mathcal{E}}, \tilde{\mathbf{p}})$ (Eq. (70));
4: for $i$ in $1 \cdots L$ do
5:     Update atomic representations using the $i$-th G3D module: $\mathbf{h} \leftarrow \texttt{G3D}^{(i)}(\mathbf{h}, \tilde{\boldsymbol{\mathcal{E}}})$ (Eq. (74), Eq. (71),
       Eq. (73));
6: end for
7: Compute the output of the atomic reference module: $T' \leftarrow T_{\mathrm{AtomRef}}(\mathbf{p}', \mathcal{M})$ (Eq. (8));
8: Compute the kinetic energy: $T_\mathrm{S} \leftarrow \sum_{a=1}^{A} \texttt{MLP}(\mathbf{h}_a) + T'$ (Eq. (75));
9: return $T_\mathrm{S}$
```

ferent types (that is, parameters) of basis functions) and the conformation $\mathbf{X} := \{\mathbf{x}^{(a)}\}_{a=1}^{A}$ to specify the centers of the basis functions. Although including $\mathcal{M}$ into the model input sacrifices formal universality, explicit dependency on $\mathcal{M}$ is arguably inevitable for an efficient density representation, which requires the structure of the problem, that is, the pattern of the density ("inductive bias" in machine learning), to reduce the representation dimension. Even the irregular grid representation inherits the pattern of molecular structure, hence a nonlocal model using grid input feature still requires generalization across molecular systems. On the other hand, the nonlocal Graphormer model architecture, based on which our neural network model is designed, has shown attractive capability to generalize across conformations and chemicals for a range of molecular tasks in previous studies [39–41]. Our extrapolation study in Results 2.3 directly validates the superior generalization capability for OFDFT. Particularly, Fig. 3(c) shows that M-OFDFT outperforms classical KEDFs which only use raw electron density input by a large margin even in an extrapolation setup, indicating the benefit of accuracy of nonlocal calculation enabled by the concise density representation based on the molecular structure $\mathcal{M}$ outweighs the sacrifice of formal universality. As for the generalization to molecules with unseen elements, since the atom type Z here as perceived by the model only represents the type of basis functions that is used to hold the electron density near the atom but does not represent the physics of the actual nucleus or its interaction with electrons, we can assign the atom of unseen element with a seen atom type, and use the basis functions of the seen element to hold the electron density around that atom. Nevertheless, due to a different electron structure, the model has not seen the pattern of density coefficients for the unseen element, so there exists a generalization challenge. This could potentially be mitigated by using a common basis set for all elements or including more elements in training, which will be investigated in future work.

The KEDF model takes atomic numbers $\mathbf{Z}$, positions $\mathbf{X}$, and density coefficients $\mathbf{p}$ of all atoms in the molecule as input. Note that given $\mathbf{p}$ and $\mathcal{M}$, there is no need of explicit density gradient or Laplacian features since they are transmitted to the features of the basis functions and thus already embodied in $\mathcal{M}$. The input variables are first used to construct node features encoding information about the electron density surrounding each atom. These features are further transformed and employed to predict the non-interacting kinetic energy. The gradient of the KEDF with respect to density coefficients for density optimization is obtained through auto-differentiation [18]. Considering the fact that nonlocal calculation is indispensable for KEDF, we build the nonlocal model based on the Graphormer architecture [39, 40]. Notably, Graphormer can handle varying-length input feature (as needed since different molecules have different numbers of atoms) in a permutation-invariant manner, which is not straightforward using other popular architectures such as multi-layer perceptions alone. The architecture has shown attractive performance in processing molecular structure to predict various properties, for example, ground-state energy and HOMO-LUMO gap of molecular systems [39, 40], and also structure sampling from a thermodynamical ensemble [41]. The nonlocal architecture has an $O(N^2)$ complexity, which does not increase the complexity of OFDFT. Our empirical results in Supplementary Section D.4.2 indicate the nonlocal formulation is crucial; restricting atomic interactions with distance cutoffs leads to a performance decline on considered molecular systems.

An overview of the KEDF model is sketched in Supplementary Figure 1(a) and summarized in Alg. 1. The process can be narrated in four stages.

(i) To adapt the Graphormer architecture for learning a physical functional, a few modifications are needed. To address the additional learning challenges mentioned in Methods 4.2 and 4.3, the input density coefficients, formulated following the details in Supplementary Section B.1.1, are first processed by the CoefficientAdapter module (Supplementary Figure 1(b)), in which the local frame module (Methods 4.2; Supplementary Section B.2) first converts the rotational equivariant coefficients to rotational invariant coefficients, and two enhancement modules NaturalReparameterization and DimensionwiseRescaling (Methods 4.3; Supplementary Section B.3.2 and B.3.1) follow in order so as to reduce the gradient scale.

(ii) Before leveraging the Graphormer architecture, initial node (that is, atom) features need to be prepared, which, in addition to the node type ($\mathbf{Z}$) and geometry ($\boldsymbol{\mathcal{E}}$, processed from $\mathbf{X}$) information in the original version of Graphormer, density coefficients ($\tilde{\mathbf{p}}$, processed from $\mathbf{p}$) should also be blended in so that the resulted node features hold the information of electron density around the respective atoms. This process is handled by the NodeEmbedding module (Supplementary Figure 1(c); Supplementary Section B.1.3). Note that the conformation input $\mathbf{X}$ is perceived by the model only in the form of pairwise distance features $\boldsymbol{\mathcal{E}}$, which is produced by the Gaussian Basis Function (GBF) module (Supplementary Section B.1.2) from pairwise distances of all atom pairs. These features are consumed by the NodeEmbedding module and also by the G3D module next. In this way, the model is naturally invariant with respect to the translation and rotation of atom coordinates $\mathbf{X}$.

(iii) Several concatenated Graphormer modules then process the node features. Since in processing the interaction between node features, the spacial relation between the two nodes needs to be considered, we use the Graphormer-3D (G3D) version [40], where the conformation information is input as pairwise distance features processed by the GBF module and a multi-layer perceptron (MLP) module. In the G3D module (Supplementary Figure 1(d); Supplementary Section B.1.4), the SelfAttention module carries out calculation on any pair of node features, which covers nonlocal interaction or correlation of density features at distance. The LayerNorm module in the G3D module is adopted following successful experiences of such models, which shifts and scales the feature distribution to make the following layer easier to process numerically.

(iv) Finally, the last-layer node features are processed by MLP and then got aggregated into one scalar, which is added with the output of the AtomRef enhancement module $T_{\text{AtomRef}}(\mathbf{p}', \mathcal{M})$ (Methods 4.3; Supplementary Section B.3.3) to construct the final energy output (Supplementary Section B.1.5). The AtomRef module shares the burden to express large gradient thus reduces the difficulty to fit large gradient for the neural-network G3D branch.

We next detail the neural network components in the model, including GBF, NodeEmbedding, and G3D modules. Other rule-based or pre-arranged modules fall in standalone topics, so we provide their details in subsequent subsections, including local frame in Supplementary Section B.2, and enhancement modules of DimensionwiseRescaling, NaturalReparameterization and AtomRef in Supplementary Section B.3. We start with handling the formatting of density coefficient features.

Supplementary Section B.1.1 Density Basis and Coefficient Specification

As mentioned in Methods 4, M-OFDFT adopts atomic basis as an efficient density representation for molecules. Each basis function ω_μ is specified by the position of the center atom and the basis function index, and we hence use $\mu = (a, \tau)$ to index the τ-th basis function centered at atom a. The basis coefficients therefore can be correspondingly attributed to each atom, which naturally serve as node-wise density features for the atom point cloud.

Specifically, we choose an even-tempered basis set [42] (Eq. (67)) as the density basis set, with its β parameter taken as 2.5. Each atom type Z (that is, atomic number) has its own set of basis functions and the size of each set $\mathcal{T}_Z$ varies from different atom types. The detailed composition of each atom type is summarized in Supplementary Table 2.

The difference of basis functions on different types of atoms makes the coefficient features hold different meanings and even come with different dimensions. This is unconventional and challenging for machine learning models to process. To make the coefficient vector homogeneous over all the atoms, the basis function sets on different atom types are joined together, and this united basis set is broadcast to all atoms, making a unified $\mathcal{T}$-dimensional density coefficient vector $\mathbf{p}_a$ on any atom a, where $\mathcal{T} := \sum_Z \mathcal{T}_Z$ is the sum of the number of basis functions over all considered atom types. The final density coefficient vector for the entire system is thus the concatenation of these $\mathcal{T}$-dimensional vectors: $\mathbf{p} := \operatorname{concat}(\{\mathbf{p}_a\}_{a=1}^{A})$. In more detail, in the QM9 dataset, the 477-dimensional concatenated coefficient vector consists of 20, 109, 116, 116 and 116 basis functions which correspond to H, C, N, O and F, respectively. For example, given the coefficient of a hydrogen (H) atom, we place them at the first 20 dimensions and use zero-valued vectors to mask other positions. A graphic illustration is shown in Supplementary Figure 2. The zero-valued mask is also employed to mask the predicted gradient during density optimization, avoiding introducing irrelevant gradient information from masked positions.

Supplementary Table 2: Orbital composition of the basis set associated with each atom type. Note that even though the compositions for N, O and F are the same, their basis sets can still be different from each other because they can have different exponents and contraction coefficients.

Atom Type	Orbitals
H	6s3p1d
C	11s8p7d3f2g
N	11s8p7d4f2g
O	11s8p7d4f2g
F	11s8p7d4f2g

Supplementary Section B.1.2 Gaussian Basis Function (GBF) Module

As shown in Supplementary Figure 1(a), the model converts the conformation (atom coordinates) input $\mathbf{X}$ into pairwise distance features before sent to the rest part of the model, so that geometric invariance with respect to $\mathbf{X}$ is guaranteed. These features are produced by the GBF module. It first converts atom coordinates $\mathbf{X} = \{\mathbf{x}^{(a)}\}_{a=1}^{A}$ into pairwise distances, and then expand each distance value $\|\mathbf{x}_a - \mathbf{x}_b\|$ into a feature vector by evaluation under a series of Gaussian basis functions:

$$\text{define } \boldsymbol{\mathcal{E}} := \texttt{GBF}(\mathbf{X}) : \quad \boldsymbol{\mathcal{E}}_{ab}^{k} := \frac{1}{\sqrt{2\pi}\sigma_k} e^{-\frac{(\|\mathbf{x}_a - \mathbf{x}_b\| - \mu_k)^2}{2\sigma_k^2}}, \tag{68}$$

where μ_k and σ_k are learnable scalar parameters representing the center and scale of the k-th Gaussian basis function.

The pairwise distance features are used by the SelfAttention module in the G3D module (Supplementary Figure 1(d); Supplementary Section B.1.4), which help identify the strength and ways of interactions between node features. They are also used to construct the initial node features in the NodeEmbedding module introduced next (Supplementary Figure 1(c); Supplementary Section B.1.3).

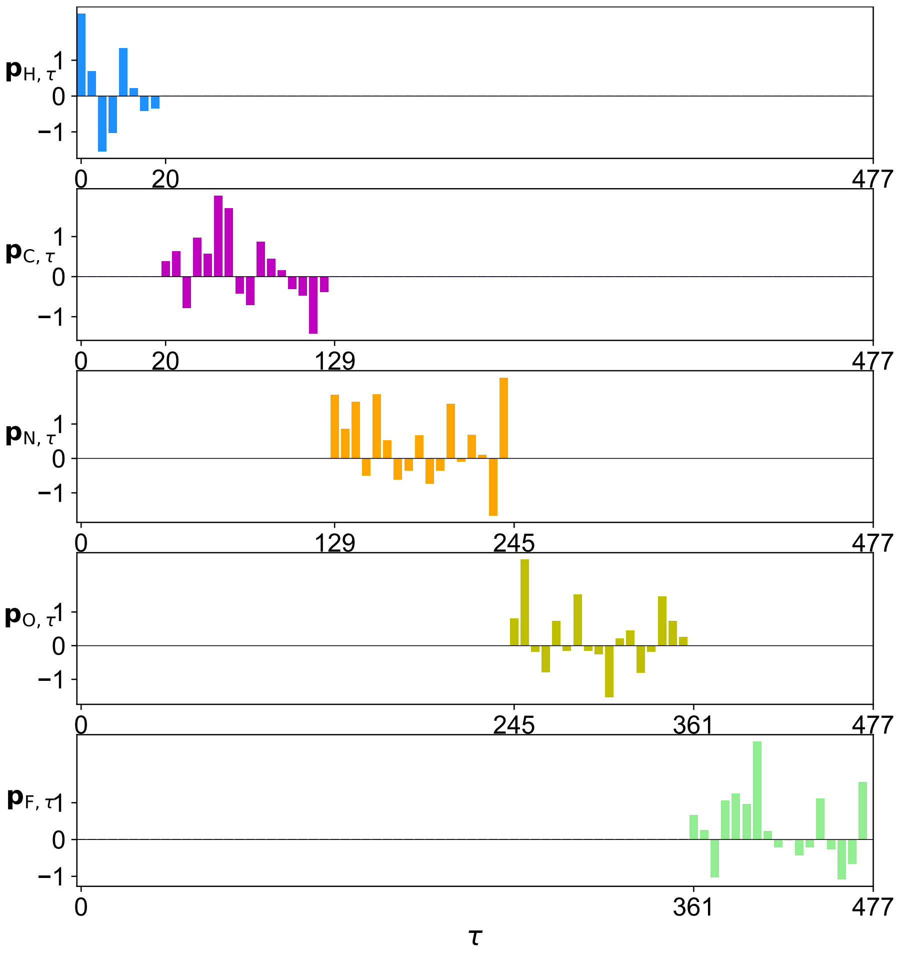
Supplementary Section B.1.3 NodeEmbedding Module

As shown in Supplementary Figure 1(c), we design a NodeEmbedding module to integrate various input features into initial node features $\mathbf{h}$, which, for each atom a, embody the atom type $Z^{(a)}$ (for specifying the type of basis functions centered at that location), the spacial position relation with other atoms (basis centered at other locations) specified by $\mathbf{X}$, and the density coefficient vector $\mathbf{p}_a$ on that atom representing the electron density around the atom (see Supplementary Section B.1.1 above). Specifically, **(i)** for the atom type $Z^{(a)}$, an AtomEmbedding module assigns a learnable feature vector $\mathbf{c}^{Z^{(a)}}$ to the atom according to its type $Z^{(a)}$. **(ii)** For positional features encoding the spacial relation of the atom a with respect to other atoms, distance features with respect to all other atoms are summed: $\sum_{b=1}^{A} \boldsymbol{\mathcal{E}}_{ab}$. **(iii)** For the density coefficient $\mathbf{p}_a$ on the atom a, it is first processed by the CoefficientAdapter module (Supplementary Figure 1(b)) detailed later in Supplementary Section B.2 and Supplementary Section B.3. The processed coefficient vector $\tilde{\mathbf{p}}_a$ becomes translation and rotation (that is, SE(3)) invariant, and is properly transformed to exhibit a mild scale range of itself and of the corresponding gradient. Due to the trade-off between the scale range of $\tilde{\mathbf{p}}_a$ and the corresponding gradient, the scale of $\tilde{\mathbf{p}}_a$ is still large for a neural network to process according to our trials. Therefore, we introduce a ShrinkGate module:

$$\texttt{ShrinkGate}(\tilde{\mathbf{p}}_a) := \lambda_{\text{co}} \tanh(\lambda_{\text{mul}} \tilde{\mathbf{p}}_a),$$

where λ_{co} and λ_{mul} are learnable scalar parameters. The $\tanh$ function is applied element-wise, which maps to a bounded space hence suppresses extreme values to avoid numerical challenges. Due to the SE(3)-invariant property of $\tilde{\mathbf{p}}_a$, applying such a nonlinear operation on it still preserves this geometry invariance.

Before aggregating features from the three sources, positional features and density coefficient features are processed by multi-layer perceptron (MLP) modules. MLP (also called feed-forward network in some context) are the most classical neural network architecture, which has been proven to be able to approximate any continuous function under certain limit [43], and are indispensable components in modern



Supplementary Figure 2: Concatenation of atomic basis for different atom types. τ is the index of density coefficient dimension and $\mathbf{p}_{Z,\tau}$ denotes the coefficient vector for atom type Z. Note that this is a schematic illustration, so the coefficient dimensions may not correspond to the actual dimensions precisely.

neural network architectures. Specifically, all the MLP modules in our model shown in Supplementary Figure 1 follow the general expression:

$$\mathtt{MLP}(\mathbf{x}) := \mathbf{U}^{(2)}\mathtt{gelu}(\mathbf{U}^{(1)}\mathbf{x} + \mathbf{b}^{(1)}) + \mathbf{b}^{(2)}, \tag{69}$$

where $\mathbf{x}$ here represents a general feature vector of a unit (node or atom pair), $\mathbf{U}^{(1)}$ and $\mathbf{U}^{(2)}$ are learnable weight matrix parameters, $\mathbf{b}^{(1)}$ and $\mathbf{b}^{(2)}$ are learnable bias vector parameters, and $\mathtt{gelu}(x) := x\Phi(x)$ for any scalar input $x \in \mathbb{R}$ is the Gaussian error linear unit activation function [44] that introduces nonlinearity to the module, where $\Phi(x)$ is the cumulative distribution function of the standard Gaussian distribution. When applied to a feature vector, gelu operates element-wise: $\mathtt{gelu}(\mathbf{x})_k := \mathtt{gelu}(\mathbf{x}_k)$.

To sum up, the whole NodeEmbedding module can be formulated as:

$$\text{define } \mathbf{h} := \mathtt{NodeEmbedding}(\mathbf{Z}, \boldsymbol{\mathcal{E}}, \tilde{\mathbf{p}}) :$$
$$\mathbf{h}_a := \mathbf{c}^{Z^{(a)}} + \mathtt{MLP}^{(\boldsymbol{\mathcal{E}})}\left(\sum_{b=1}^{A} \boldsymbol{\mathcal{E}}_{ab}\right) + \mathtt{MLP}^{(\mathbf{p})}\big(\mathtt{ShrinkGate}(\tilde{\mathbf{p}}_a)\big), \tag{70}$$

where $\mathtt{MLP}^{(\boldsymbol{\mathcal{E}})}$ and $\mathtt{MLP}^{(\mathbf{p})}$ are two MLP instances with independent parameters.

Supplementary Section B.1.4 Graphormer-3D (G3D) Module

The Graphormer-3D (G3D) module (Supplementary Figure 1(d)) is the main neural network module in the KEDF model (Supplementary Figure 1(a)). It is a Transformer-based [22] graph neural network, which performs nonlocal calculation between every pair of node features even located at distance, while is tailored to taking into account the spacial relations between the two locations. Each G3D module contains two layer normalization (LayerNorm) modules, a SelfAttention module, and an MLP module, assembled in the way shown in Supplementary Figure 1(d).

LayerNorm The Layer Normalization module [45] is a widely adopted technique in Transformer-based architectures. It shifts and scales the running feature vector in a neural network so that the values over the vector components distribute with zero mean and unit variance, which is the numerical range over which neural network modules are the most sensitive, thus facilitates stable and faster training and a better fit to data. This module is applied before sending node features to the SelfAttention module and the MLP module. It normalizes the node feature vector on each node independently:

$$\begin{aligned}\texttt{LayerNorm}(\mathbf{h}_a) &:= \mathbf{s} \odot \frac{\mathbf{h}_a - \mu_a}{\sigma_a} + \mathbf{b}, \\ \text{where } \mu_a := \frac{1}{D_{\text{hid}}} \sum_{k=1}^{D_{\text{hid}}} \mathbf{h}_a^k, &\quad \sigma_a := \sqrt{\frac{1}{D_{\text{hid}}} \sum_{k=1}^{D_{\text{hid}}} (\mathbf{h}_a^k - \mu_a)^2},\end{aligned} \tag{71}$$

D_{hid} is the dimension of the input feature vector $\mathbf{h}_a$ for node a, $\mathbf{s} \in \mathbb{R}^{D_{\text{hid}}}$ and $\mathbf{b} \in \mathbb{R}^{D_{\text{hid}}}$ are learnable vector parameters, and $\odot$ denotes element-wise multiplication.

SelfAttention The SelfAttention module is the processor responsible for nonlocal calculation that produces the result of interaction between any two node feature vectors. For updating feature vector on node (that is, atom) a, the vanilla SelfAttention module [22] considers interaction of the current feature vector $\mathbf{h}_a$ of node a with the feature vector $\mathbf{h}_b$ of each of the other nodes as well as itself. This is done by constructing a "query" feature vector $\mathbf{Q}_a := \mathbf{U}^{(\text{query})}\mathbf{h}_a$ for node a to interact with other nodes (including itself), and each of the other nodes, say, node b (could be a), constructs a "key" feature vector $\mathbf{K}_b := \mathbf{U}^{(\text{key})}\mathbf{h}_b$ to respond to the "query", and a "value" feature vector $\mathbf{V}_b := \mathbf{U}^{(\text{value})}\mathbf{h}_b$ to convey its contribution. Here, $\mathbf{U}^{(\text{query})}$, $\mathbf{U}^{(\text{key})}$ and $\mathbf{U}^{(\text{value})}$ are learnable weight matrix parameters in $\mathbb{R}^{D'_{\text{hid}} \times D_{\text{hid}}}$, where D_{hid} is the dimension of each node feature vector $\mathbf{h}_a$ or $\mathbf{h}_b$, and D'_{hid} is another hyperparameter determining the dimension of the key, query and value feature vectors. The respond of the "key" to the "query" is modeled by $\frac{\mathbf{Q}_a^\top \mathbf{K}_b}{\sqrt{D'_{\text{hid}}}}$ which is treated as the unnormalized log-probability, or logit, to determine the portion that node b contributes to the new node feature vector of node a. The probability, or the portion of contribution, is recovered by applying the softmax function to the logits:

$$\texttt{softmax}(\boldsymbol{\ell})_b := \frac{e^{\boldsymbol{\ell}_b}}{\sum_{b'} e^{\boldsymbol{\ell}_{b'}}}.$$

In matrix form, the updated node feature vectors $\mathbf{h}' := [\mathbf{h}'_1, \cdots, \mathbf{h}'_A] \in \mathbb{R}^{D'_{\text{hid}} \times A}$ as a stacked array over all the nodes is:

$$\begin{aligned}\mathbf{h}' &:= \mathbf{V}\,\texttt{softmax}\Big(\frac{\mathbf{Q}^\top \mathbf{K}}{\sqrt{D'_{\text{hid}}}}\Big)^\top \in \mathbb{R}^{D'_{\text{hid}} \times A}, \\ \text{where } \mathbf{Q} &:= \mathbf{U}^{(\text{query})}\mathbf{h},\ \mathbf{K} := \mathbf{U}^{(\text{key})}\mathbf{h},\ \mathbf{V} := \mathbf{U}^{(\text{value})}\mathbf{h},\ \mathbf{h} := [\mathbf{h}_1, \cdots, \mathbf{h}_A] \in \mathbb{R}^{D_{\text{hid}} \times A}.\end{aligned}$$

To enlarge expressiveness, it is common practice to introduce "multi-head attention", where the above self attention calculation is repeated D_{head} times, and the D_{head}-fold results are concatenated. More explicitly, the updated node feature vector array is:

$$\begin{aligned}\mathbf{h}' &:= [\mathbf{h}'_1, \cdots, \mathbf{h}'_A] \in \mathbb{R}^{(D_{\text{head}} D'_{\text{hid}}) \times A}, \\ \text{where } \mathbf{h}'_a &:= \texttt{concatenate}(\{\mathbf{h}'^{(1)}_a, \cdots, \mathbf{h}'^{(D_{\text{head}})}_a\}) \in \mathbb{R}^{D_{\text{head}} D'_{\text{hid}}}, \forall a = 1 \cdots A,\end{aligned}$$

are reshaped from:

$$\begin{aligned}[\mathbf{h}'^{(d)}_1, \cdots, \mathbf{h}'^{(d)}_A] &:= \mathbf{V}^{(d)}\,\texttt{softmax}\Big(\frac{\mathbf{Q}^{(d)\top} \mathbf{K}^{(d)}}{\sqrt{D'_{\text{hid}}}}\Big)^\top \in \mathbb{R}^{D'_{\text{hid}} \times A}, \forall d = 1 \cdots D_{\text{head}}, \\ \text{where } \mathbf{Q}^{(d)} &:= \mathbf{U}^{(\text{query},d)}\mathbf{h},\ \mathbf{K}^{(d)} := \mathbf{U}^{(\text{key},d)}\mathbf{h},\ \mathbf{V}^{(d)} := \mathbf{U}^{(\text{value},d)}\mathbf{h},\ \mathbf{h} := [\mathbf{h}_1, \cdots, \mathbf{h}_A] \in \mathbb{R}^{D_{\text{hid}} \times A}.\end{aligned}$$

To let the updated feature vector array $\mathbf{h}'$ have the same shape as the original $\mathbf{h}$, hyperparameters D_{head} and D'_{hid} are chosen such that $D_{\text{head}} D'_{\text{hid}} = D_{\text{hid}}$.

The vanilla SelfAttention module handles general featured point (node) cloud input, but for a set of featured atoms, there is a spacial or positional relation between a pair of atoms. Distance is a natural way to describe such relation. For example, a shorter distance generally indicates a stronger interaction between the two node feature vectors. To inform the attention mechanism of this characteristic, Graphormer-3D (G3D) [40] introduce pairwise distance features into the attention mechanism. To accommodate for the different usage of pairwise distance features from that in the NodeEmbedding module, a learnable MLP layer is applied to the original pairwise distance features, pair by pair, which maps each distance feature vector to dimension D_{head}:

$$\tilde{\boldsymbol{\mathcal{E}}} := \texttt{MLP}(\boldsymbol{\mathcal{E}}) \in \mathbb{R}^{A \times A \times D_{\text{head}}} : \quad \tilde{\boldsymbol{\mathcal{E}}}_{ab} := \texttt{MLP}(\boldsymbol{\mathcal{E}}_{ab}) \in \mathbb{R}^{D_{\text{head}}}, \tag{72}$$

where the MLP in the latter expression follows the general formulation in Eq. (69). This new pairwise distance feature array is incorporated into the vanilla self attention as a bias to the contribution logits. For an explicit expression, let $\tilde{\boldsymbol{\mathcal{E}}}^{(d)} := [\tilde{\boldsymbol{\mathcal{E}}}^{(d)}_{ab}]_{ab}$ denote the $A \times A$ matrix combining the d-th distance feature for all the pairs. The expression for the SelfAttention module is:

$$\begin{aligned}
&\text{define } \mathbf{h}' = [\mathbf{h}'_1, \cdots, \mathbf{h}'_A] := \texttt{SelfAttention}(\mathbf{h}, \tilde{\boldsymbol{\mathcal{E}}}) \in \mathbb{R}^{D_{\text{hid}} \times A} \text{ for } \mathbf{h} := [\mathbf{h}_1, \cdots, \mathbf{h}_A] \in \mathbb{R}^{D_{\text{hid}} \times A} : \\
&\quad \mathbf{h}'_a := \texttt{concatenate}(\{\mathbf{h}'^{(1)}_a, \cdots, \mathbf{h}'^{(D_{\text{head}})}_a\}) \in \mathbb{R}^{D_{\text{head}} D'_{\text{hid}} = D_{\text{hid}}}, \forall a = 1 \cdots A, \\
&\text{where } [\mathbf{h}'^{(d)}_1, \cdots, \mathbf{h}'^{(d)}_A] := \mathbf{V}^{(d)} \, \texttt{softmax}\Big(\frac{\mathbf{Q}^{(d)\top} \mathbf{K}^{(d)}}{\sqrt{D'_{\text{hid}}}} + \tilde{\boldsymbol{\mathcal{E}}}^{(d)} \Big)^{\top} \in \mathbb{R}^{D'_{\text{hid}} \times A}, \forall d = 1 \cdots D_{\text{head}}, \\
&\quad \mathbf{Q}^{(d)} := \mathbf{U}^{(\text{query},d)} \mathbf{h}, \ \mathbf{K}^{(d)} := \mathbf{U}^{(\text{key},d)} \mathbf{h}, \ \mathbf{V}^{(d)} := \mathbf{U}^{(\text{value},d)} \mathbf{h}.
\end{aligned} \tag{73}$$

Assembly The third component in the G3D module is an MLP module, which follows the same form as given in Eq. (69), and is applied to the node feature vector of each node independently (that is, the nodes share the same MLP module to process their feature vectors). These modules are combined to make the G3D module following the illustration in Supplementary Figure 1(d). Explicitly in equation,

$$\begin{aligned}
&\text{define } \mathbf{h}' := \texttt{G3D}(\mathbf{h}, \tilde{\boldsymbol{\mathcal{E}}}) : \\
&\quad \mathbf{h}' := \texttt{MLP}(\texttt{LayerNorm}(\mathbf{h}'')) + \mathbf{h}'', \\
&\quad \mathbf{h}'' := \texttt{SelfAttention}(\texttt{LayerNorm}(\mathbf{h}), \tilde{\boldsymbol{\mathcal{E}}}) + \mathbf{h}.
\end{aligned} \tag{74}$$

Supplementary Section B.1.5 Output Process

To produce a scalar output, each node feature vector $\mathbf{h}_a$ is processed to produce a scalar by an MLP module following the form of Eq. (69), whose output dimension (that is, number of rows of $\mathbf{U}^{(2)}$) is 1. The scalars from all the nodes are summed up, and the summed value is added with the output of the AtomRef module (Methods 4.3; Supplementary Section B.3.3) to produce the final energy output:

$$T_{\text{S}} := \sum_{a=1}^{A} \texttt{MLP}(\mathbf{h}_a) + T_{\text{AtomRef}}(\mathbf{p}', \mathcal{M}). \tag{75}$$

Supplementary Section B.1.6 Model Configuration

In all experimental settings, we employ the same backbone architecture, Graphormer, specifically utilizing the Graphormer-3D (*G3D*) encoder module. To ensure a fair comparison, most hyperparameters in all models are maintained consistently (for example, model depth and hidden dimension). A summary of all hyperparameter choices can be found in Supplementary Table 3. It is worth mentioning that in the shrink gate of *NodeEmbedding*, the initial value of learnable parameters ν_{co} is set to 10, while the initial value of ν_{mul} is set to 0.02 for the ethanol dataset and 0.05 for all other datasets. We did not conduct an extensive hyperparameter search, and most hyperparameters were chosen following Graphormer [39]. All models are implemented using the PyTorch deep learning framework [18].

Supplementary Section B.1.7 Training Hyperparameters

Following Graphormer [39], all models are trained using the Adam optimizer and a linear decay learning schedule. The peak learning rate is set to 1×10^{-4} for our functional models and 3×10^{-4} for baseline models (that is, M-NNP and M-NNP-Den). Other optimizer hyperparameters can be found in Supplementary Table 3. A warmup stage featuring a linearly increasing learning rate is introduced to stabilize training during the initial stage. The number of warmup steps is set to 30k for M-OFDFT and 60k for baseline models. The batch size is set to 256 for the ethanol dataset and 128 for all other datasets. All models are trained on Nvidia Tesla V100 GPUs.

Supplementary Table 3: Hyperparameters of the Graphormer model. These hyperparameters are adopted in all methods in the present work (M-OFDFT, M-NNP, M-NNP-Den).

Hyperparameter	M-OFDFT
G3D Modules	12
Hidden Dimension	768
MLP Hidden Dimension	768
Number of Heads	32
GBF Dimension	128
Dropout	0.1
Attention Dropout	0.1
Optimizer	Adam
Learning Rate Schedule	Linear decay
Adam (β_1, β_2)	(0.95, 0.99)
Adam ϵ	1×10^{-8}

For different molecule datasets, the number of training epochs is determined by examining the loss curve. Training is halted once the validation loss fails to decrease for 20 epochs. Specifically, our functional models are trained for approximately 600 and 700 epochs on the ethanol and QM9 datasets, respectively. For the QMugs dataset, our model is trained for approximately 700 epochs, while the M-NNP and M-NNP-Den models are trained for 2,300 epochs. Notably, we propose a series of QMugs datasets with increasing molecular size in Results 2.3, where the number of SCF datapoints will slightly increase as the average molecular size increases (larger molecules generally require more SCF iterations to converge). To ensure a fair comparison, we maintain approximately the same number of training epochs for different datasets. In the Chignolin experiment, there are four Chignolin datasets with increasing peptide length and data size. Our functional models are trained for 1,400, 1,100, 800, and 750 epochs, respectively, while the M-NNP and M-NNP-Den models are trained for 3,000, 3,000, 1,000, and 900 epochs, respectively.

In practice, we employ a weighted loss function to optimize the KEDF model:

$$L = \xi_{\text{eng}} L_{\text{eng}} + \xi_{\text{grad}} L_{\text{grad}} + \xi_{\text{den}} L_{\text{den}},$$

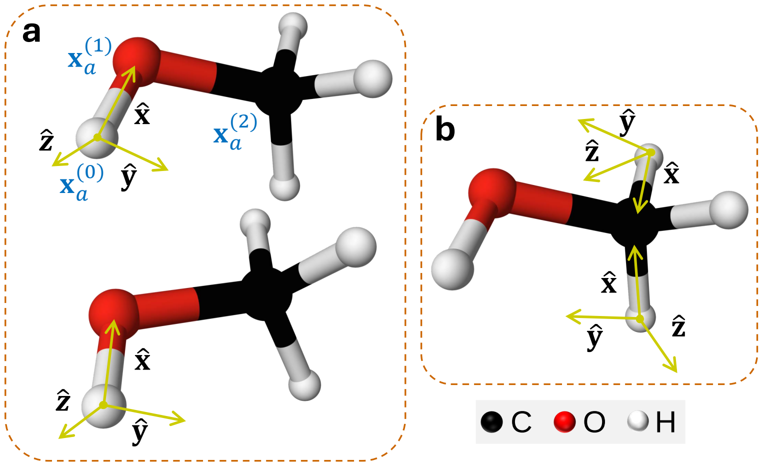
where L_{eng}, L_{grad}, and L_{den} represent the KEDF energy loss (Eq. (3)), gradient loss (Eq. (4)), and projected density loss (Eq. (80)), respectively. ξ_{eng}, ξ_{grad}, and ξ_{den} are the corresponding loss weights. The selection of loss weights is determined through grid search on the validation set for various datasets independently, and the utilized loss weights are provided in Supplementary Table 4.

Supplementary Table 4: Loss weights for various datasets and learning targets. The loss weights are tuned on each setting by conducting a grid search on the validation set.

Dataset	ξ_{eng}	ξ_{grad}	ξ_{den}
Ethanol-$T_{\text{S,res}}$	1	0.1	0.08
Ethanol-E_{TXC}	1	0.03	1
QM9-$T_{\text{S,res}}$	1	0.12	0.005
QM9-E_{TXC}	1	0.12	0.05
QMugs	1	0.1	1
Chignolin	1	0.1	1

Supplementary Section B.2 Local Frame Module for Geometric Invariance

As mentioned in Methods 4.2, by expanding the electron density on a set of atomic basis, the expansion coefficients $\mathbf{p}$ mathematically comprise geometric tensors of various orders equivariant to rotations and translations. A local frame simultaneously rotating with the structure is adopted to decouple the density feature from the change of the coordinate system and unnecessary geometric variability. As shown in Supplementary Figure 3(a), to construct the local frame at an atom at $\mathbf{x}_a^{(0)}$, we choose $\hat{\mathbf{x}}$ pointing to its nearest atom $\mathbf{x}_a^{(1)}$. The $\hat{\mathbf{z}}$ axis lies in the line of the cross-product of $\hat{\mathbf{x}}$ with the direction to the second-nearest not-on-$\hat{\mathbf{x}}$ atom $\mathbf{x}_a^{(2)}$ and the $\hat{\mathbf{y}}$ axis is then given by $\hat{\mathbf{y}} = \hat{\mathbf{z}} \times \hat{\mathbf{x}}$. Note that we exclude the hydrogen atoms in the neighborhood of the center atom following [46], making the obtained local frame depend more on heavy atoms, whose positions are more stable than those of hydrogen atoms and reflect more



Supplementary Figure 3: Illustration of the local frame. **(a)** The local frame is constructed for each atom according to its local environment, which is equivariant with respect to. the transformation of the global frame. **(b)** Local frames of similar substructures (for example, the three H-C bonds in the methyl group) rotate with substructures accordingly, resulting in invariant coefficient features for locally similar density patterns.

reliable local structures. Denote the local frame associated with atom a as:

$$\mathcal{R}_a := (\hat{\mathbf{x}}, \hat{\mathbf{y}}, \hat{\mathbf{z}}),$$

the density coefficient vector and the gradient vector under the local frame are calculated by:

$$\mathbf{p}_a'^{l} = \mathbf{D}^l(\mathcal{R}_a)\mathbf{p}_a^l, \quad \nabla_{\mathbf{p}_a'^{l}} T_{\mathrm{S}} = \mathbf{D}^l(\mathcal{R}_a)\nabla_{\mathbf{p}_a^l} T_{\mathrm{S}}. \tag{76}$$

where l is the degree of tensors or azimuthal quantum number. $\mathbf{p}^l$ corresponds to the coefficient of basis functions of the degree l (or type-l tensors in mathematics). $\mathbf{D}^l(\mathcal{R}_a)$ is the Wigner-D matrix of degree l.

Notably, an additional benefit of the local frame is that it makes a stable feature for locally similar density patterns. For example, consider the coefficients corresponding to the atomic basis on the three hydrogen atoms in a methyl group (Supplementary Figure 3(b)). As the three H-C bonds are highly indistinguishable, the densities on the bonds are very close and contribute almost identically to the kinetic energy. But as the three bonds have different orientations relative to a common global coordinate system, the coefficients on the hydrogen atoms are vastly different if the basis functions on different hydrogen atoms are aligned to the same directions. In contrast, basis functions under local frames are oriented equivariantly with the orientation of the local substructures, leaving the coefficients largely invariant, which makes more physical meanings to predict the energy. For example, coefficients on basis functions aligning with the $\hat{\mathbf{x}}$ direction always represent the density on the shortest bond with the atom. As a result, the local frame makes almost the same density coefficients on the three hydrogen atoms in the methyl group *even* though the H-C bonds have different orientations. We also provide a numerical visualization of the ethanol dataset, where the target functional is chosen as the residual energy of the non-interacting kinetic energy after deducted by the value given by the base kinetic energy functional APBE [21]. As shown in Supplementary Figure 4, the local frame always attains a noticeable scale (standard deviation) reduction across various atom types, demonstrating its capability to eliminate unnecessary geometric variability caused by various orientations of the chemical bonds. More importantly, as shown in Supplementary Figure 5, this approach also brings a substantial scale reduction for gradient labels across various atom types, especially for H atoms, where almost all coefficient dimensions exhibit a $> 60\%$ gradient scale reduction. This is crucial to alleviate the large gradient range and make the subsequent dimension-wise rescaling module easier (See more discussions in Supplementary Section B.3.1). These two advantages of the local frame are beneficial for the efficient optimization of the KEDF model. The empirical results in Supplementary Section D.4.3 suggest that the KEDF model achieves a 2-fold lower training and test error by using this technique.

Supplementary Section B.3 Enhancement Modules for Expressing Vast Gradient Range

As mentioned in Methods 4.3, the raw gradient range of the input is vast, and conventional data normalization techniques are not applicable in our task since minimizing the gradient scale is conflicting with minimizing the coefficient scale. For the same reason we neither can take the logarithm of the density feature, as a number of works have adopted [47], since the gradient would be correspondingly scaled up. Another way to decrease the gradient scale is to downscale the energy value (that is, using a larger energy unit). However, this does not bring further improvement in our trials. As mentioned in the main context, this is due to the trade-off between easy learning and model resolution: an error in the normalized (small) scale will be enlarged in the original scale. It neither works to use a separate model to learn the gradient directly, as we have empirically observed that the energy value model easily overfits the training data, and during density optimization the gradient model unnecessarily decreases the energy constructed from the value model and even appears non-conservative as the optimization diverges.

Having both large coefficients and gradients implies a function with a large Lipschitz coefficient, which leads to a great challenge for the optimization of conventional neural networks. According to these observations, we turn to elaborating on a series of enhancement modules to express the vast gradient range and enable effective training.

Supplementary Section B.3.1 Dimension-wise Rescaling

Dimension-wise rescaling of coefficients and gradients is applied after the process of local frame and natural reparameterization. The two modules made the rescaling easier, but there is still a scale trade-off between coefficients and gradients. We hence choose moderate values for the scale parameters in Eq. (5): the target gradient scale s_{grad} is set to 0.05 and the maximal coefficient scale s_{coeff} is set to 50. Compared to the gradient scale, handling input coefficients with a larger range is more manageable. For example, incorporating a ShrinkGate module to further compress the coefficients into a bounded space, as discussed in Supplementary Section B.1.3. We found this technique yields an admirable performance empirically. Consequently, we prefer allowing a larger coefficient scale in the trade-off. In the context of deep learning, the Lipschitz coefficient (that is, the maximum absolute gradient that the neural network model could express) is usually used to indicate the capability of a model fitting gradient label. Following this convention, we take the maximum absolute gradient value to measure the label scale.

To illustrate the importance of dimension-wise rescaling, we visualize in Supplementary Figure 6 the scales of gradients and density coefficients over the dimensions before and after the processing of dimension-wise rescaling in the setting of learning residual KEDF with APBE base KEDF on the QM9 dataset. The shown gradient scales are estimated after centralizing the gradient values by subtracting the gradient mean of each dimension with the atomic reference module. As plotted in Supplementary Figure 6(a), many gradient dimensions have an extremely large scale. We found this leads to great difficulty in the practical optimization of deep learning models. After dimension-wise rescaling, most dimensions are rescaled to have the desired gradient scale (Supplementary Figure 6(b)). As illustrated in Supplementary Figure 6(c), the density coefficients are rescaled to a larger scale, and a ShrinkGate module is introduced to normalize the coefficients into a friendly space (Supplementary Section B.1.3). As a result, we found that the neural network model without the dimension-wise rescaling technique hardly converges and the loss curve is particularly volatile, due to the smoothness of common neural network architectures which restricts the range of the output gradient of the model. Applying the dimension-wise rescaling technique effectively mitigates this dilemma and enables efficient optimization.

Supplementary Section B.3.2 Natural Reparameterization

Although dimension-wise rescaling allows the trade-off between numerical scales of coefficients and gradients, the trade-off often has a difficult frontier that the two sides cannot be simultaneously made in a mild scale. A way to improve the trade-off is to balance the difficulties over the dimensions. The dimensions exhibit different scales since they have different sensitivity to the density hence the energy. Different basis functions spread over different regions in space, so a same amount of coefficient change on different basis functions influences the density function differently. To understand and balance the sensitivities, we derive how the change of coefficients affect the density function. Consider a small change $\Delta\mathbf{p}$ to the coefficients $\mathbf{p}$, which leads to a change in the density function $\Delta\rho(\mathbf{r}) := \rho_{\mathbf{p}+\Delta\mathbf{p}}(\mathbf{r}) - \rho_{\mathbf{p}}(\mathbf{r})$. This difference is typically measured by the L2-metric in the function space: $\int |\rho_{\mathbf{p}+\Delta\mathbf{p}}(\mathbf{r}) - \rho_{\mathbf{p}}(\mathbf{r})|^2 \,\mathrm{d}\mathbf{r} = \int |\Delta\rho(\mathbf{r})|^2 \,\mathrm{d}\mathbf{r} = \int |\rho_{\Delta\mathbf{p}}(\mathbf{r})|^2 \,\mathrm{d}\mathbf{r} = \int \sum_\mu \Delta\mathbf{p}_\mu \omega_\mu(\mathbf{r}) \sum_\nu \Delta\mathbf{p}_\nu \omega_\nu(\mathbf{r}) \,\mathrm{d}\mathbf{r} = \Delta\mathbf{p}^\top \mathbf{W} \Delta\mathbf{p}$, where $\mathbf{W}_{\mu\nu} := \int \omega_\mu(\mathbf{r}) \omega_\nu(\mathbf{r}) \,\mathrm{d}\mathbf{r}$ is the overlap matrix of the atomic basis for density. As the atomic basis is not orthonormal, $\mathbf{W}$ is not isotropic (that is, cannot be turned proportional to the identity matrix by orthogonal transformations), so the same amount of coefficient change in different dimensions has different effects on the density function.

The proposed natural parameterization $\tilde{\mathbf{p}} := \mathbf{M}^\top \mathbf{p}$, where $\mathbf{M}$ is a square matrix satisfying $\mathbf{M}\mathbf{M}^\top = \mathbf{W}$, fulfills the desired balance. To see this, noting that $\mathbf{W}$ is non-singular hence is $\mathbf{M}$, we have $\mathbf{p} = \mathbf{M}^{-\top}\tilde{\mathbf{p}}$, so the density change is $\int |\Delta\rho(\mathbf{r})|^2 \,\mathrm{d}\mathbf{r} = \Delta\mathbf{p}^\top \mathbf{W} \Delta\mathbf{p} = \Delta\tilde{\mathbf{p}}^\top \mathbf{M}^{-1}\mathbf{W}\mathbf{M}^{-\top}\Delta\tilde{\mathbf{p}} = \Delta\tilde{\mathbf{p}}^\top \mathbf{M}^{-1}\mathbf{M}\mathbf{M}^\top\mathbf{M}^{-\top}\Delta\tilde{\mathbf{p}} = \Delta\tilde{\mathbf{p}}^\top\Delta\tilde{\mathbf{p}}$, hence the change of coefficient in any dimension contributes equally to the density change. The gradient is reparameterized accordingly, following $\nabla_{\tilde{\mathbf{p}}} T_\mathrm{S} = (\nabla_{\tilde{\mathbf{p}}}\mathbf{p})^\top \nabla_{\mathbf{p}} T_\mathrm{S} = \mathbf{M}^{-1}\nabla_{\mathbf{p}} T_\mathrm{S}$.

Choosing the matrix $\mathbf{M}$ still faces a degree of freedom of an orthogonal transformation. We choose:

$$\mathbf{M} = \mathbf{Q}\sqrt{\mathbf{\Lambda}}, \tag{77}$$

where $\sqrt{\mathbf{\Lambda}}$ denotes element-wise square-root operation, and the diagonal matrix $\mathbf{\Lambda}$ and orthogonal matrix $\mathbf{Q}$ come from the eigenvalue decomposition of $\mathbf{W} = \mathbf{Q}\mathbf{\Lambda}\mathbf{Q}^\top$ (note $\mathbf{W}$ is symmetric positive definite so such decomposition exists). We found it achieves more balanced sensitivity than using Cholesky decomposition in terms of the resulted largest gradient scale after dimension-wise rescaling. Although the eigenvalue decomposition requires a cubic complexity, it is only needed once per molecular structure, hence does not introduce a large overhead compared to the cost in density optimization. Empirically, natural reparameterization considerably stabilizes the optimization of the functional model and brings lower training and validation loss, as presented in Supplementary Section D.4.3, underscoring the necessity of balancing the density coefficient with a physical metric.

Supplementary Section B.3.3 Atomic Reference Module

As mentioned in Methods 4.3, the per-type bias statistics $\{\bar{T}_Z\}_Z$ and the global bias $\bar{T}_\mathrm{global}$ on the dataset are solved from the following over-determined linear equations using least squares method:

$$\bar{T}_{\mathcal{M}^{(d)}} := \sum_Z A_Z^{(d)} \bar{T}_Z + \bar{T}_\mathrm{global} = \mathrm{mean}\{T_\mathrm{S}^{(d,k)} - \bar{\mathbf{g}}_{\mathcal{M}^{(d)}}{}^\top \mathbf{p}^{(d,k)}\}_k, \quad \forall d,$$

where $A_Z^{(d)} := \#\{a \in \mathcal{M}^{(d)} : Z^{(a)} = Z\}$ is the number of atoms of type Z in molecule $\mathcal{M}^{(d)}$. Note that on the ethanol and Chignolin datasets where all structures share the same constitution, we only use the global bias, that is, $\bar{T}_\mathcal{M} := \bar{T}_\mathrm{global}$. A similar treatment for energy bias is also adopted in [48].

To demonstrate the benefit of the atomic reference module, we visualize the gradient scale reduction by the module on the QM9 dataset in Supplementary Figure 7. We find that the atomic reference module substantially reduces the scale (variance) of gradient labels, especially for dimensions corresponding to 's' atomic orbital functions. Since 's' orbitals usually have large gradient mean values but small variance values, subtracting the gradient mean from these dimensions greatly simplifies the learning of gradient labels.

CoefficientAdapter Module These enhancement modules and the neural network model are combined in the way shown in the model architecture Supplementary Figure 1(a). In the architecture, the natural reparameterization module and the dimension-wise rescaling module are combined with the local frame technique into the CoefficientAdapter module (Supplementary Figure 1(b)). It transforms the vanilla density coefficient $\mathbf{p}$ to a neural-network-friendly density feature $\tilde{\mathbf{p}}$ as the direct input to the neural network model. It not only guarantees geometric invariance of density features, but also facilitates efficient learning with vast gradient labels. We outline the implementation in Alg. 2.

Supplementary Section B.4 Functional Variants

In principle, besides the unknown KEDF, any energy density functionals in the decomposition formulation of ground-state energy (Eq. (10)) can be taken as the training objective of the proposed M-OFDFT framework. Here we first describe the two versions we mainly employed in the implementation of M-OFDFT, and then describe our exploration in other functional variants.

Supplementary Section B.4.1 Residual KEDF

In the implementation of M-OFDFT, the first functional version we introduced aims to reduce the difficulty of modeling KEDF directly by employing an existing KEDF as the base functional and learning the residual:

$$T_{\mathrm{S},\theta}(\mathbf{p}, \mathcal{M}) := T_{\mathrm{S,res},\theta}(\mathbf{p}) + T_\mathrm{S,base}(\mathbf{p}, \mathcal{M}). \tag{78}$$

Algorithm 2 Evaluation of the CoefficientAdapter module

```
Require: Input density coefficients $\mathbf{p}$.
Require: Pre-computed dimension-wise rescaling factors $\{\lambda_{Z,\tau}\}_{Z,\tau}$ (see Eq. (5) and Supplementary Section B.3.1); pre-computed quantities for the target molecular structure $\mathcal{M}$: Wigner-D matrices $\{\{\mathbf{D}^l_a := \mathbf{D}^l(\mathcal{R}_a)\}_{l=0}^{l_{\max}}\}_{a=1}^{A}$ (see Eq. (76)) for transforming coefficients on each atom onto the local frame of the atom, and the square-root matrix $\mathbf{M}$ of the density-basis overlap matrix $\mathbf{W}$ (see Eq. (77)).
1: for $a$ in $1 \cdots A$ do
2:     for $l$ in $0 \cdots l_{\max}$ do
3:         Transform density coefficients using the Wigner-D matrix: $\mathbf{p}^l_a \leftarrow \mathbf{D}^l_a \mathbf{p}^l_a$ (Eq. (76));
4:     end for
5: end for
6: Conduct natural reparameterization: $\mathbf{p}' \leftarrow \mathbf{M}^\top \mathbf{p}$ (Eq. (7));
7: for $a$ in $1 \cdots A$ do
8:     for $\tau$ in $1 \cdots \mathcal{T}$ do
9:         Rescale density coefficients dimension-wise: $\tilde{\mathbf{p}}_{a,\tau} \leftarrow \lambda_{Z^{(a)},\tau}\, \mathbf{p}'_{a,\tau}$ (Eq. (6));
10:    end for
11: end for
12: return $(\tilde{\mathbf{p}}, \mathbf{p}')$
```

Specifically, we chose the APBE KEDF [21] as the base KEDF since it best fits the training data on the QM9 dataset among four functionals mentioned in Methods 4.7.

The residual KEDF formulation also allows leveraging a lower bound of KEDF which introduces non-negativity to the residual model, in hope for better extrapolation performance by encoding this analytical property into the model. Nevertheless, we find that it is not straightforward to gain benefits from this property, as it introduces additional training challenge which outweighs its merit. Specifically, we take the von Weizsäcker (vW) KEDF [49] as the base KEDF, which is a lower bound of the true KEDF [6, Thm. 1.1]. The training challenge is that it renders the gradient for the residual model to learn in vast range. The visualization of processed gradient and coefficient scales presented in Supplementary Figure 8. Even after the processing of local frame, natural reparameterization, atomic reference model, and dimension-wise rescaling, the processed gradient scale, in terms of the maximum absolute value across all dimensions and all datapoints, is 2.08×10^7 on the QM9 dataset (Supplementary Figure 8(a)), which is orders larger than the scale 2.74 when using the APBE as the base KEDF (Supplementary Figure 6(b)), and the scale 4.82 for the TXC version below (that is, learning the sum of the KEDF and XC functional), even allowing a larger processed density coefficient scale of 1113.87 (Supplementary Figure 8(b)) vs. 507.44 for APBE base KEDF (Supplementary Figure 6(c)) and 451.59 for the TXC version. This large gradient scale impedes any effective training of the neural network model. We even tried dropping out datapoints with particularly large gradient labels that exceed a chosen threshold for training, but observed a performance degradation, due to reduced information on a broader range of densities. We suspect that this difficulty may be due to the divergence between learning an easier rule and learning a numerically more friendly target. The vW functional leaves the neural network model to learn a non-negative residual, which can be regarded as an easier rule. But since the vW functional is a lower bound, it may not approximate the KEDF closely, hence could leave the residual and its gradient in a large scale.

Nevertheless, in Supplementary Section D.1.1 we empirically verified the learned KEDF model satisfies this lower bound.

Supplementary Section B.4.2 TXC Functional

The residual KEDF version has achieved a simpler learning target with a tractable gradient range, but it has a computational bottleneck of evaluating the value and gradient of the APBE base KEDF as well as the PBE XC functional from the density coefficient, which is conducted on a grid. As discussed in Supplementary Section A.3.2, the time complexity of calculating the value is $O(MN_{\text{grid}})$ (recall M is the number of basis functions). Evaluating the gradient by automatic differentiation requires the same time complexity as evaluating the value, and moreover, it also requires $O(MN_{\text{grid}})$ memory occupation to store intermediate values for back-propagation. Considering the large prefactor of N_{grid} (commonly $\sim 10^3 N$), the computational cost for residual KEDF to conduct density optimization becomes unaffordable for large-scale systems. Such cost was observed on the QMugs dataset, for which Supplementary Section D.3 presents more detailed results.

Algorithm 3 Usage of M-OFDFT to solve a given molecular system (density optimization process)

```
Require: A trained kinetic residual model $T_{\mathrm{S,res},\theta}(\mathbf{p}, \mathcal{M})$ or TXC model $E_{\mathrm{TXC},\theta}(\mathbf{p}, \mathcal{M})$, molecular structure $\mathcal{M} = \{\mathbf{X}, \mathbf{Z}\}$ of the given system comprising atomic numbers $\mathbf{Z}$ and positions $\mathbf{X}$ of all atoms, gradient descent step size $\varepsilon$, initialization method init_method, a trained density coefficient projection branch $\Delta\mathbf{p}_\theta$ if init_method == 'ProjMINAO' (Supplementary Section B.5.2).
 1: Compute constants for this molecular structure $\mathcal{M}$: density basis normalization vector $\mathbf{w}$, overlap matrix $\mathbf{W}$ and its square-root matrix $\mathbf{M}$ for natural reparameterization, 2-center-2-electron Coulomb integral $\tilde{\mathbf{W}}$ for evaluating $E_{\mathrm{H}}$ (Eq. (47)), external potential vector $\mathbf{v}_{\mathrm{ext}}$ for evaluating $E_{\mathrm{ext}}$ (Eq. (49)), rotation matrices and Wigner-D matrices for local frame, build grid and calculate density basis values on grid points for evaluating $E_{\mathrm{XC}}$ and $T_{\mathrm{S,base}}$ if using $T_{\mathrm{S,res},\theta}$ model;
 2: if init_method == 'Hückel' then
 3:     $\mathbf{p}^{(1)} \leftarrow$ Hückel_init($\mathcal{M}$);
 4: else if init_method == 'ProjMINAO' then
 5:     $\mathbf{p} \leftarrow$ MINAO_init($\mathcal{M}$);
 6:     $\mathbf{p}^{(1)} \leftarrow \mathbf{p} - \Delta\mathbf{p}_\theta(\mathbf{p}, \mathcal{M})$;
 7: end if
 8: $k \leftarrow 1$
 9: while stopping criterion (described in Supplementary Section B.5.3) is not met do
10:     Compute electronic energy using the PyTorch implementation (see Alg. 1 for evaluating $T_{\mathrm{S,res},\theta}$ or $E_{\mathrm{TXC},\theta}$):
11:     if using the kinetic residual model then
12:         $E^{(k)} \leftarrow T_{\mathrm{S,res},\theta}(\mathbf{p}^{(k)}, \mathcal{M}) + T_{\mathrm{S,base}}(\mathbf{p}^{(k)}, \mathcal{M}) + E_{\mathrm{H}}(\mathbf{p}^{(k)}, \tilde{\mathbf{W}}) + E_{\mathrm{XC}}(\mathbf{p}^{(k)}, \mathcal{M}) + E_{\mathrm{ext}}(\mathbf{p}^{(k)}, \mathbf{v}_{\mathrm{ext}})$;
13:     else if using the TXC model then
14:         $E^{(k)} \leftarrow E_{\mathrm{TXC},\theta}(\mathbf{p}^{(k)}, \mathcal{M}) + E_{\mathrm{H}}(\mathbf{p}^{(k)}, \tilde{\mathbf{W}}) + E_{\mathrm{ext}}(\mathbf{p}^{(k)}, \mathbf{v}_{\mathrm{ext}})$;
15:     end if
16:     Calculate the gradient of the electronic energy with respect to the density coefficients:
        $\nabla_{\mathbf{p}} E^{(k)} \leftarrow$ auto_grad($E^{(k)}, \mathbf{p}^{(k)}$);
17:     Update the density coefficients using projected gradient:
        $\mathbf{p}^{(k+1)} \leftarrow \mathbf{p}^{(k)} - \varepsilon\left(\mathbf{I} - \frac{\mathbf{w}\mathbf{w}^\top}{\mathbf{w}^\top\mathbf{w}}\right)\nabla_{\mathbf{p}} E^{(k)}$;
18:     $k \leftarrow k + 1$
19: end while
20: return $(E, \mathbf{p})$ at the iteration determined by the stopping criterion.
```

To get rid of calculation on grid, the second version of learning target for the deep-learning model is introduced as the sum of the KEDF and the XC functional, which we call the TXC functional, denoted as $E_{\mathrm{TXC},\theta}(\mathbf{p}, \mathcal{M})$.

Supplementary Section B.4.3 Learning Other Functionals

Besides the two versions above, we also experimented other direct targets for the deep-learning model to learn, including directly learning the KEDF $T_{\mathrm{S}}[\rho]$ (that is, without a base KEDF), the universal functional $U[\rho]$ (Eq. (7)), and the total energy functional $E_{\mathrm{tot},\mathcal{M}}[\rho]$, that is, electronic energy $E[\rho]$ in Eq. (10) plus inter-nuclear energy $E_{\mathrm{nuc}}(\mathcal{M})$ in Eq. (59) later. We found that both directly learning the KEDF and the universal functional are hard to optimize due to their large gradient range, which is even larger than that for learning the residual KEDF model and the TXC model, and cannot be effectively handled even using the proposed techniques. Interestingly, learning the total energy functional $E_{\mathrm{tot},\mathcal{M}}[\rho]$, that is, adding the external energy E_{ext} and inter-nuclear energy $E_{\mathrm{nuc}}(\mathcal{M})$ to the universal functional as the target, substantially reduces the gradient range and enables stable optimization, especially on protein systems we considered. Specifically, on the 50 test Chignolin structures considered in Results 2.3, learning the total energy model $E_{\mathrm{tot},\theta}(\mathbf{p}, \mathcal{M})$ using data of all peptides gives a better density optimization result (per-atom energy MAE $0.071\,\mathrm{kcal/mol}$ as in Fig. 3(e) 'M-OFDFT/Pretrain', per-atom *relative* energy MAE $0.097\,\mathrm{kcal/mol}$ as in Fig. 3(c) 'M-OFDFT') than learning the TXC model $E_{\mathrm{TXC},\theta}$ (per-atom energy MAE $0.102\,\mathrm{kcal/mol}$, per-atom *relative* energy MAE $0.148\,\mathrm{kcal/mol}$). Nevertheless, these results by the $E_{\mathrm{TXC},\theta}$ model are still reasonable and effective, and are substantially better than the classical OFDFT using the APBE KEDF (per-atom *relative* energy MAE $0.684\,\mathrm{kcal/mol}$ as in Fig. 3(c) 'APBE').

Supplementary Section B.5 Density Optimization Details

After a functional model is learned, M-OFDFT solves a given molecular structure by density optimization, as shown in Alg. 3. Here we provide additional results and details for density optimization.

Supplementary Section B.5.1 Additional Density Optimization Results

As mentioned in Methods 4.4, M-OFDFT can produce reasonable optimization curves for a given molecular structure from either of the two initialization methods Hückel and ProjMINAO, and gives accurate energy and HF force results. Here we illustrate the density optimization behaviors of various initialization strategies using a QM9 molecule. As shown Fig. 9, the optimization process starting from the MINAO density leads to an obvious gap from the target KSDFT energy. In contrast, the optimization curve starting from Hückel density converges closely to the target energy, although it shows a larger energy error than MINAO density at initialization. Furthermore, ProjMINAO initialization also converges closely to the target energy, and finally achieves a better performance than Hückel initialization. These findings demonstrate that both Hückel and ProjMINAO initializations have a better alignment with the training-density manifold, and thus lead to better generalization performance during density optimization. More attractively, Fig. 9 implies that M-OFDFT only requires an on-manifold initialization but does not need projection in each density optimization step, implying its better robustness than previous methods.

To further show the advantage of the density optimization results, we compare M-OFDFT with classical OFDFT using well-established KEDFs, following the same setting as Fig. 9 except on a different QM9 molecule. First, we compare the optimization curves in energy by M-OFDFT and OFDFT with classical KEDFs including TF [2, 3], TF+$\frac{1}{9}$vW [50], TF+vW [51], and APBE [21], from the conventional MINAO [16] initialization. As illustrated in Supplementary Figure 10(a-b), M-OFDFT converges closely to the true ground-state energy using both initialization methods, while all classical KEDFs lead to optimization curves falling below the true ground-state energy. We did not plot the curve for TF+$\frac{1}{9}$vW since it diverges so vastly that the curve soon runs out of the shown range.

We further plot the density error along with the optimization process, measured in the L2-metric $\int |\rho(\mathbf{r}) - \rho^\star(\mathbf{r})|^2 \,\mathrm{d}\mathbf{r}$ from the KSDFT ground-state density $\rho^\star(\mathbf{r})$ (see Supplementary Section B.3.2 for calculation details). As shown in Supplementary Figure 10(c-d), M-OFDFT from either initialization continuously drives the density towards the true ground-state density and results in a small density error, even though the optimization process is not driven by minimizing the density error (but by minimizing electronic energy). This indicates that the energy objective of M-OFDFT constructed from the learned functional model is not artificial in the sense of only producing the correct energy after optimization, but it also leads to the correct electron density, so the functional holds the desired physical meaning. In contrast, density error curves by classical KEDFs diverge quickly and present an ascending trend, revealing a problem for applying these functionals to molecular systems. Again, we did not plot the curve for TF+$\frac{1}{9}$vW since it soon runs out of the shown range.

A subtlety in the density optimization process is the non-negativity of the density value everywhere in space. As the basis functions follow the form of the multiplication of a Gaussian radial function which is always non-negative, and a spherical harmonic function or a monomial (as is the case of the even-tempered basis in Eq. (67)) that accounts for the angular anisotropicity which can take negative values. The coefficients hence need to be within a certain region to guarantee that the represented density function is non-negative everywhere. Due to the complexity of the basis functions, an explicit expression for such a constraint is not obvious. We hence implemented a numerical guarantee that enforces the non-negativity of density value on each grid point, by adding the following artificial energy penalty term to the minimization objective Eq. (45) of density optimization:

$$E_{\text{nonneg}}(\mathbf{p}, \mathcal{M}) := \sum_{g=1}^{N_{\text{grid}}} \max\{-\rho_{\mathbf{p}}(\mathbf{r}^{(g)}), 0\}, \tag{79}$$

where $\rho_{\mathbf{p}}(\mathbf{r}) := \sum_\mu \mathbf{p}_\mu \omega_\mu(\mathbf{r})$ is the electron density function represented by coefficient vector $\mathbf{p}$ (see Eq. (25)), and $\{\mathbf{r}^{(g)}\}_{g=1}^{N_{\text{grid}}}$ is a set of grid points for the molecular structure $\mathcal{M}$. Nevertheless, in our empirical trials, we found that this additional term is seldom activated during density optimization, and density optimization without this term already leads to an electron density that is non-negative on almost all grid points. The number of exceptional grid points is even smaller than that due to density fitting error. Considering the cost of evaluating density values on grid points, we hence omitted this step.

For ensuring density non-negativity, it is possible to represent the density as the square of linear combination of the atomic orbital basis functions, as adopted in many existing OFDFT implementations (for example, [33, 52–54]). But this would revert the computational complexity to quartic ($O(N^4)$) due to the Hartree term, and sacrifice the advantage over KSDFT. Future explorations could be expanding the density onto a set of non-negative-valued basis functions.

Supplementary Section B.5.2 Density Initialization Details

Recall that we introduced two initialization methods in Methods 4.4, Hückel and ProjMINAO, for stable density optimization in M-OFDFT. Initialization is worthy of the attention since it should stay in the training-data manifold of the functional model for reliable prediction. Training data come from eigensolutions of effective one-electron Hamiltonian matrix, which motivates the first choice of the Hückel initialization [55, 56] which follows the same mechanism. The second choice is to project the initial density onto the manifold, inspired by previous methods, for example, using local PCA [57, 58] or kernel PCA [59]. But these existing projection techniques is not easily applicable for M-OFDFT since the density optimization domain is the coefficient space generated by the given molecular structure, which differs for different molecular structures, so it is not straightforward to determine the training-data manifold within the coefficient space for an unseen molecular structure. We hence use an additional deep-learning model to conduct the projection of the MINAO initialization, which we call ProjMINAO.

To implement the ProjMINAO initialization, we construct an additional output branch $\Delta\mathbf{p}_\theta(\mathbf{p}, \mathcal{M})$ to predict the required correction to project the given density coefficient $\mathbf{p}$ towards the ground-state density coefficient $\mathbf{p}^\star$ for the molecular structure $\mathcal{M}$. Due to the formulation of KSDFT, the ground-state density corresponds to the eigensolution of an effective Hamiltonian (Supplementary Section A.2), hence is on the training-data manifold for sure. Projecting towards the ground state could also accelerate convergence in density optimization. The branch is built on top of the last *G3D* module shown in Supplementary Figure 1(a) of the KEDF model. It processes the hidden representations of each atom through an MLP module to predict the corresponding coefficient difference. The branch is trained to project the density coefficient data along the SCF iteration of KSDFT onto the corresponding ground-state coefficient:

$$\sum_d \sum_k \left\| \Delta\mathbf{p}_\theta(\mathbf{p}^{(d,k)}, \mathcal{M}^{(d)}) - (\mathbf{p}^{(d,k)} - \mathbf{p}^{(d,\star)}) \right\|. \tag{80}$$

Note that even though the projection is aimed at the ground state, we do not rely on it for accuracy, as the functional model could continue optimizing the density (Fig. 9).

Since the MINAO or Hückel initialization primarily produces orbital coefficients, density fitting (Supplementary Section A.4.1) is required to convert them to density coefficients to complete the density initialization. Note that this procedure is performed only *once* for each molecular structure, so the cubic computational cost of density fitting (and generating Hückel initialized orbitals) does not dominate over the quadratic scaling of density optimization on regular workloads.

For very large molecules, we also propose an alternative initialization to directly generate the density coefficients by superposition of isolated-atom densities, in the same spirit of MINAO which uses the superposition of isolated-atom orbitals. This amounts to concatenating the fitted MINAO density coefficients of each atom as if in isolation, which only has a linear computational cost. This constructed density coefficients are then fed to the projection branch for the correction, which is the final initialized density. In practice, this technique drastically reduces the time needed to construct the initial coefficients, further reducing the overall time consumption of M-OFDFT, and renders only a minor increase of error (approximately 0.02 $\mathrm{kcal/mol}$ in per-atom energy error on the interested protein systems in Results 2.4).

The effectiveness of this approach to handle off-the-training-data-manifold issue in density optimization is demonstrated qualitatively by the convergent curves of energy and density in Supplementary Figure 9 and Supplementary Figure 10, and quantitatively by the results in Results 2.2 and Results 2.3. In contrast to some other deep-learning OFDFT methods [57, 59, 58] which require the on-manifold projection in each density optimization step, M-OFDFT only requires an on-manifold initialization, meaning at most one projection. This makes the implementation much more efficient especially for large systems, and also indicates that the learned functional generalizes better since it is more robust to unseen densities.

Supplementary Section B.5.3 Stopping Criterion

Another required specification for density optimization is stopping criterion. Ideally, the solution from M-OFDFT is expected at the stationary point with zero projected gradient of the electronic energy. But in practice, the exact zero-gradient point may be missed due to discretization error. Therefore, we propose a set of stopping criteria to determine the practical stationary point over a chosen number of optimization steps.

- For molecules in a similar scale as those in training, the stopping criterion is chosen as the step with minimal single-step energy update, that is, at the global (over the chosen number of optimization steps) minimum of single-step energy update. This is the closest step to the stationary point in practice.
- For larger-scale molecules than those in training, however, the above criterion is not as effective as on in-scale systems. The reason is that the reliable coefficient region around the ground-state density for the model is likely smaller and anisotropic due to the extrapolation risk, so the coefficients may find a mistaken direction to "sneak off" the ground state in an extensive optimization. Note that it is hard to project the confronted density during optimization onto the training-data manifold precisely, since the manifold of coefficient is unknown for a molecular structure not present in training. Therefore, we mitigate this problem by controlling the extent of density optimization to balance optimization effectiveness and reliability. Specifically, we consider two additional criteria: **(1)** the step where the projected (onto the tangent space of normalized coefficients; not the training-data manifold) gradient norm *first* stops decreasing, that is, the first local minimum of the projected gradient norm. **(2)** the step where the single-step energy *first* stops decreasing, that is, the first local minimum of single-step energy update. These two criteria characterize a local minimum on the optimization process, hence represent a certain level of optimization effectiveness. Meanwhile, they are chosen to be the first encountered point with such characterization, where is likely still in the reliable region for the model. Based on empirical results on the validation set, we prefer first invoking criterion **(1)**. If this criterion is not met within the chosen optimization steps, we seek for criterion **(2)**. If this is still not met, we resort to the original criterion of the global minimum of single-step energy update, which always exists.

Supplementary Section B.5.4 Density Optimization Hyperparameters

During the deployment stage, we use the ProjMINAO as the initialization choice for all settings in Results 2. The adopted gradient descent steps and step size ε are chosen according to the performance of evaluation datasets. The gradient descent steps and step size are set to 1000 and 1×10^{-3} for all molecular datasets and functional variants (including traditional KEDF baselines), with the exception of implementing the learned KEDF models $T_{\mathrm{S,res},\theta}$ and $E_{\mathrm{TXC},\theta}$ on the ethanol dataset, where the step size ε is set to 5×10^{-4}. The vanilla Stochastic Gradient Descent (SGD) optimizer is adopted in the density optimization process.

Supplementary Section C Related Work

In this section, we give a more detailed discussion on prior works that leverage machine learning to approximate the kinetic energy density functional (KEDF) for OFDFT. Kernel ridge regression is employed in pioneering works [57, 59, 60, 58, 30, 31], including the extension that leverages kernel gradients [61]. These works have proven the success of the idea of machine-learning OFDFT. Such models can be seen as nonlocal, but the costly calculation on grid restricts the applications to 1-dimensional systems. Some other works fit a linear combination of classical KEDF approximations with explicit expression [62], including nonlocal ones [63], but the demonstrated systems are still effectively 1-dimensional. Deep neural networks have also been explored recently, including multi-layer perceptrons (a.k.a feed-forward neural networks) [64–66, 52, 53, 67] and convolutional neural networks [68, 61, 69, 54]. Many of them learn the kinetic energy density at each grid point from semi-local density features on that point [53], and to compensate for nonlocal effects, third-order [64, 65, 52] and fourth-order [66] density derivative features are leveraged. Others consider the interaction of density features at different locations hence are nonlocal [68, 61, 69, 67, 54]. Many of such works enable the calculation on 3-dimensional systems [68, 64–66, 53, 69, 54], and the lower computational complexity than KSDFT has been shown empirically [53]. Nevertheless, the demonstrated systems are still limited to tiny molecules of dozen atoms, with exceptions of [68] with about 30 atoms but restricted to alkanes, and [53] with a few thousands atoms but on material systems and without accuracy evaluation. The contribution of our work M-OFDFT is the applicability to diverse and larger molecules (for example, the Chignolin polypeptide containing 168 atoms) with extrapolation capability while maintaining the low complexity of OFDFT. Technical contributions are summarized in the beginning of Methods 4.

Supplementary Section D Additional Empirical Results

Supplementary Section D.1 In-Scale Results

Supplementary Section D.1.1 Quantitative Results

Supplementary Table 5: Performance of M-OFDFT with different learning targets and initialization configurations, as an extension to the results narrated in Results 2.2. The mean absolute errors (MAEs) in energy and Hellmann-Feynman (HF) force are listed in $\mathrm{kcal/mol}$ and $\mathrm{kcal/mol/Å}$, respectively. The results are calculated on ethanol test structures (n=10,000) and QM9 test molecules (n=13,388).

Method	Ethanol		QM9	
	Energy	HF force	Energy	HF force
$T_{\text{S,res}}$-Hückel	5.27	27.73	9.88	12.73
$T_{\text{S,res}}$-ProjMINAO	**0.18**	**1.18**	0.93	**2.91**
E_{TXC}-Hückel	3.95	126.65	9.69	123.51
E_{TXC}-ProjMINAO	**0.18**	2.07	**0.73**	4.73
M-NNP	0.10	25.24	0.25	7.72

Functional variants and initialization As mentioned in Results 2.2, M-OFDFT achieves chemical accuracy on unseen conformations (ethanol) and chemicals (QM9) in a similar scale as seen during training, using a deep-learning model targeting the residual KEDF on top of a base KEDF. Here we provide results under more technical choices in Supplementary Table 5. Firstly, we let the model learn the sum of the kinetic and XC energy E_{TXC} to get rid of the large prefactor of grid computation for calculation on large-scale molecules (Supplementary Section B.4.2). This is adopted in time complexity (Results 2.4) and extrapolation (Results 2.3) studies, and here we list its performance in this in-scale case. Secondly, as presented in Methods 4.4, two initialization strategies are proposed to address the out-of-distribution issue of initialized density. The results reported in Results 2.2 are produced by ProjMINAO initialization, and here we also list the results produced by the Hückel initialization method, which is one of the standard initialization methods in common DFT calculation [16, 56], and does not require an additional deep-learning model (or module). We also provide the results of M-NNP, that is, a deep-learning model with the same architecture that directly predicts the ground-state energy of the given molecular structure in an end-to-end way.

From Supplementary Table 5, we can see that although the E_{TXC} model combines with an additional unknown density functional, it can still be effectively trained, and does not have obvious negative effects on the accuracy. The energy accuracy is even improved on both datasets given the same initialization method. For the initialization methods, even without the deep-learning shortcut, M-OFDFT still achieves a reasonable accuracy using the Hückel initialization. With an additional deep-learning module for predicting a correction, the ProjMINAO initialized density stays close to the ground-state density, so it can yield better results. Finally, M-OFDFT achieves a comparable energy accuracy with the end-to-end method M-NNP, even though the process to give the energy by M-OFDFT (density optimization using the KEDF model) is not directly supervised. Notably, on tasks that do not have a direct supervision, M-OFDFT is superior over M-NNP. As the table shows, M-OFDFT achieves a substantially more accurate HF force estimation than M-NNP, even though neither has seen HF force label in training. Abundant evidence in Results 2.3 also shows the dominating advantage of M-OFDFT in extrapolation to molecules larger than seen during training.

Details on the potential energy surface (PES) study As shown in Fig. 2(c) of Results 2.2, M-OFDFT can accurately reproduce the PESs of ethanol. Here we provide more implementation specifics of the evaluation process. To benchmark the PES, we generate a series of ethanol structures by varying either the $\mathrm{H{-}C{-}C{-}O}$ torsion angle or the $\mathrm{O{-}H}$ bond length, starting from the equilibrium ethanol conformation (optimized by classical molecular dynamics simulation). The torsion angles are taken uniformly on $[-180°, 180°)$ with 15° increment, where the 0° angle is defined when the four atoms are on the same plane and H and O are on the same side. The bond lengths are taken uniformly on $[0.856\,\text{Å}, 0.144\,\text{Å}]$ with 0.015 Å increment. The interval is taken as the range of the $\mathrm{O{-}H}$ bond length in the training dataset. The energy curves in Fig. 2(c) are produced by the residual KEDF $T_{\text{S,res},\theta}$ version (Supplementary Section B.4.1) of M-OFDFT with ProjMINAO density initialization. This setting is consistent with other results shown in Results 2.2. The curves are *not* evaluated on densities solved by KSDFT [68, 65].

Geometry optimization study To investigate the utility of M-OFDFT for geometry optimization, we integrate the M-OFDFT implementation with the geometry optimization framework in PySCF [16], wherein the HF force (Supplementary Section A.5) by M-OFDFT is used. We generate a set of initial ethanol structures by varying the H−C−C−O torsion angle from the equilibrium ethanol conformation. The torsion angles are taken uniformly on $[-180°, 60°]$ with 30° increment. For each initial structure, we relax the structure using both KSDFT and M-OFDFT for at most 100 steps. For M-OFDFT, the residual KEDF $T_{\mathrm{S,res},\theta}$ version (Supplementary Section B.4.1) with ProjMINAO density initialization is used, which is consistent with other results shown in Results 2.2.

To evaluate the optimized structures by M-OFDFT, we first calculate the rooted mean square deviation (RMSD) between the optimized structure by M-OFDFT and the optimized structure by KSDFT for each initial ethanol structure. The mean RMSD value across all the initial structures is 0.07 Å, indicating a good consistency of M-OFDFT with KSDFT. To further evaluate the optimized structures by M-OFDFT, we compare the bond lengths and angles against those of optimized structures by KSDFT. For a reference to assess the error, for each bond or angle type, we plot the distribution in the form of violin plot of the bond length or angle value in the ethanol training dataset. As the dataset is from the MD17 dataset [27, 70], the plot represents the distribution in thermodynamic equilibrium. As depicted in Supplementary Figure 12, the majority of the bond lengths and angles of the optimized structures by M-OFDFT exhibit good agreement with the results of KSDFT, and align closely with the high-density region of the corresponding thermodynamic equilibrium distributions. The difference from KSDFT results is also substantially smaller than the span of the corresponding thermodynamic equilibrium distribution. This result underscores the practical efficacy of M-OFDFT for geometry optimization.

Comparison with vW as a lower bound As mentioned in Supplementary Section B.4.1, since the von Weizsäcker (vW) KEDF [49] is a lower bound of the true KEDF [6, Thm. 1.1], taking it as the base KEDF could inform the $T_{\mathrm{S,res}}$ model to be non-negative, but unfortunately introduces more training challenges. Hence it remains to be explored to leverage the lower-bound property of the vW KEDF. Nevertheless, we can empirically verify that our learned KEDF models already satisfy this lower bound. For this, we present a scatter plot in Supplementary Figure 13, where each point represents the vW KEDF value (x-axis coordinate) and the kinetic energy value by our learned KEDF model (y-axis coordinate) of each electron density. The densities for this evaluation are taken as the ground-state densities optimized by our KEDF model on unseen ethanol test structures (Supplementary Figure 13(a-b)) or unseen QM9 test structures (Supplementary Figure 13(c-d)), following the setting in Results 2.2. Our KEDF model takes either the residual KEDF version (Supplementary Section B.4.1) with APBE base KEDF (Supplementary Figure 13(a,c)), or the TXC functional version (Supplementary Section B.4.2) which gives the kinetic energy value by subtracting the E_{XC} value from the model-predicted value E_{TXC} (Supplementary Figure 13(b,d)). The figure clearly shows that in all cases, the kinetic energy values by our KEDF model are larger than the corresponding vW values, hence our learned KEDF models satisfy this lower bound property.

Supplementary Section D.1.2 Visualization of Optimized Densities

Radial density plot For more qualitative investigation of M-OFDFT, we provide more visualization results of the optimized density by M-OFDFT, in addition to Fig. 2(b). Supplementary Figure 14 shows the radial density by spherical integral around the other two heavy atoms in the ethanol molecule, that is, the α-carbon atom (the one connected to the oxygen atom) and the β-carbon atom. The results demonstrate that the density curves of M-OFDFT again align almost identically with the KSDFT density curves. In contrast, the classical KEDF using APBE for KEDF again leads to a substantial deviation around bonding regions between peaks of core electrons.

Partial charge and dipole moment visualization As reported in Results 2.2, the optimized density of M-OFDFT can accurately reproduce Hirshfeld partial charges [71] and dipole moments of molecules. To further illustrate the results, we provide a representative example in Supplementary Figure 15 of atomic partial charge on each atom in an unseen test ethanol structure as well as the dipole moment of the structure, based on the optimized density by M-OFDFT. The results show that both the Hirshfeld partial charges for each atom and the dipole moment from M-OFDFT are in close agreement with those obtained from KSDFT in the ethanol molecule.

Supplementary Section D.2 Extrapolation Results

Supplementary Section D.2.1 Results on QMugs and Chignolin

Validation error in the extrapolation experiments As shown in Results 2.3, M-OFDFT has shown a superior extrapolation performance in energy than end-to-end deep-learning counterparts M-NNP and M-NNP-Den that predict the total energy directly. We emphasize here that this is not due to that M-NNP and M-NNP-Den are not well trained or overfit to the training dataset. Supplementary Table 6 shows that M-NNP and M-NNP-Den achieve a better validation error than M-OFDFT, but their extrapolation errors increase much more vastly than that of M-OFDFT.

Supplementary Table 6: Training, validation and extrapolation error (per-atom energy MAE in $\mathrm{kcal/mol}$) of our M-OFDFT and reference methods M-NNP and M-NNP-Den on the QMugs and Chignolin extrapolation setups. For the QMugs case, all the methods are trained and validated on two disjoint splits of QM9 molecules and QMugs molecules with no more than 15 heavy atoms, and are tested for extrapolation on QMugs molecules (n=50) with 56-60 heavy atoms. For the Chignolin case, all the methods are trained and validated on two disjoint splits of fragment structures of all peptide lengths (2-5), and are tested for extrapolation on Chignolin structures (n=1,000). Note that for M-OFDFT, the training and validation energy errors are from end-to-end model output, while the extrapolation error is from the result after density optimization using the model, consistent with Fig. 3.

Method	QMugs			Chignolin		
	Train	Validation	Extrapolation	Train	Validation	Extrapolation
M-NNP	0.008	**0.017**	1.768	0.003	0.003	0.084
M-NNP-Den	**0.004**	0.019	1.824	**0.002**	**0.002**	0.079
M-OFDFT	0.044	0.055	**0.112**	0.015	0.016	**0.071**

Extrapolation performance in HF force We then show the better extrapolation performance of M-OFDFT in HF force. As shown in Supplementary Figure 16(a), M-OFDFT with functional model trained on molecules with fewer than 15 heavy atoms consistently maintains a substantially lower HF force MAE than M-NNP and M-NNP-Den on larger-scale molecules. M-OFDFT also outperforms the two end-to-end counterparts across various fragment datasets in the Chignolin experiment (Supplementary Figure 16(b)).

In the finetuning setting of Chignolin, Supplementary Figure 16(c) shows that M-OFDFT always achieves much better HF force results than M-NNP and M-NNP-Den in each of the three setups. M-OFDFT attains a substantial 40.8% HF force error reduction in finetuning over training from scratch, showing the ability of capturing transferable knowledge from accessible-scale data, and more efficiently leveraging limited large-scale data. In contrast, the end-to-end deep-learning methods exhibit even worse HF force results when incorporating accessible-scale data, manifesting an extrapolation difficulty to a larger scale.

Extrapolation comparison to NNP with other architectures Results 2.3 have demonstrated the qualitatively better extrapolation performance of M-OFDFT than direct energy prediction, that is, the neural network potential (NNP) formulation, using the same model architecture as M-OFDFT, denoted as M-NNP. We further verify that this conclusion still holds even using more advanced and recent architectures for NNP. We consider Equivariant Transformer (ET) [72] and Equiformer [73], which are recently proposed NNP architectures that have shown remarkable performance and competitiveness in the field. Notably, ET is one of state-of-the-art equivariant NNP architectures that use Cartesian vector features to maintain $\mathrm{SE}(3)$-equivariance, and Equiformer is a cutting-edge approach amongst NNP architectures that leverage high-order spherical harmonics tensors to encode molecular features.

Following the setting of Fig. 3(a) introduced in Results 2.3, we train the NNP models on QM9 and QMugs molecules with no more than 15 heavy atoms, and test them on increasingly larger molecules from the QMugs dataset. Results in Supplementary Figure 17 demonstrate that although using the more advanced architectures improves the performance over M-NNP, the error in the extrapolation cases is still larger than that of M-OFDFT, and the error still increases with molecule size, while the error of M-OFDFT keeps constant or even decreasing.

Evaluation of electron density in the Chignolin experiment For a thorough assessment of the extrapolation capability of M-OFDFT, we evaluate the electron density solved by M-OFDFT on peptide structures in various lengths. As peptides are relatively large molecules, it is inconvenient to directly visualize the densities. We hence calculate the Hirshfeld partial charge [71] and dipole moment from the solved density.

This evaluation is conducted in parallel with the setting in Results 2.3. To construct the evaluation benchmark, we prepare a test set encompassing a range of short-peptide structures, from dipeptides to pentapeptides, as well as Chignolin structures (of length 10). We sample 50 structures for each category of peptides. More details about the peptide structures are described in Methods 4.5.4. For solving test peptide structures of lengths 2 to 5, we apply the total energy functional model $E_{\text{tot},\theta}$ trained on all training peptide structures (of lengths 2-5), following the same setting as Fig. 3(c). For solving test Chignolin structures, we further finetune the model on 800 training Chignolin structures. We take KSDFT results to evaluate error, and compare the results with the classical OFDFT using the TF+$\frac{1}{9}$vW KEDF. Results in Supplementary Table 7 demonstrate that M-OFDFT consistently outperforms the classical OFDFT over peptides of all lengths, in terms of the accuracy on these density-related quantities. The partial charge MAE of M-OFDFT is substantially lower, by two orders of magnitude. This substantial improvement further underscores the power of M-OFDFT.

Supplementary Table 7: Hirshfeld partial charge and dipole moment results in mean absolute error (MAE) from KSDFT. The units for partial charge and dipole moment are e and D, respectively. Each dataset contains 50 structures.

Test dataset	Quantity	M-OFDFT	TF+$\frac{1}{9}$vW
Dipeptide	Partial charges	**2.62**$\times 10^{-3}$	0.147
	Dipole moment	**0.217**	2.970
Tripeptide	Partial charges	**2.68**$\times 10^{-3}$	0.153
	Dipole moment	**0.283**	3.556
Tetrapeptide	Partial charges	**2.68**$\times 10^{-3}$	0.139
	Dipole moment	**0.390**	3.420
Pentapeptide	Partial charges	**2.85**$\times 10^{-3}$	0.141
	Dipole moment	**0.543**	3.474
Chignolin	Partial charges	**3.32**$\times 10^{-3}$	0.132
	Dipole moment	**1.077**	12.049

Supplementary Section D.2.2 Additional Extrapolation Study from QM9

Results on charged molecules As clarified in Supplementary Section B.1, the input atom types $\mathbf{Z}$ and positions $\mathbf{X}$ are only used to inform the KEDF model of the types and centers of the atomic basis functions under which the expansion coefficient input is defined, but not for the physics of the actual atoms. This formulation makes M-OFDFT inherently general for handling input densities from either neutral or charged molecular systems, just by feeding the KEDF model with expansion coefficients of the corresponding electron density onto the atomic basis functions whose types and positions are specified by $\mathbf{Z}$ and $\mathbf{X}$, even if the KEDF model is trained on data only from neutral molecules.

Here, we demonstrate the efficacy of M-OFDFT in handling charged molecules. To construct an evaluation benchmark, we randomly select five unseen carboxylic acid molecules from the QM9 test set, and deprotonate the hydrogen cation from the carboxyl group ($-$C(O)OH) of each molecule, thereby generating five carboxylate anions ($-$C(O)O$^-$). We employ the TXC functional model $E_{\text{TXC},\theta}$ trained on the QM9 training set, which comprises neutral molecules only, to solve these charged systems. We initialize the electron density using ProjMINAO (Methods 4.4), which projects the electron density from the MINAO initialization onto the training-data manifold by a deep-learning model. Since MINAO gives the initial electron density by treating the system as neutral, we rescale the density coefficients produced by ProjMINAO to normalize to the correct number of electrons of the charged system. Note that in the density optimization process (Eq. (2)), the number of electrons is kept throughout, so the ground-state density solution also respects the correct charge of the system. We evaluate the performance of M-OFDFT in terms of the mean absolute error (MAE) from KSDFT results over the five systems in the energy difference between the neutral and the corresponding charged system.

The result is that M-OFDFT achieves a $3.80\,\mathrm{kcal/mol}$ MAE of energy difference. In comparison, the result of the classical OFDFT using the TF+$\frac{1}{9}$vW KEDF is $30692.16\,\mathrm{kcal/mol}$, an error of five orders larger. This result indicates that M-OFDFT trained on neutral systems is still effective in handling charged molecules, an extrapolation capability of a new kind. The capability for charged molecules can be further improved if data from charged molecules are included.

Results on unseen chemical environments Extrapolating a trained machine-learning model to unseen chemical environments, for example, bond types not present in training data, is another challenging extrapolation evaluation. To investigate this type of extrapolation capability of M-OFDFT, we apply M-OFDFT to the carbon monoxide molecule CO, which contains a triple bond C≡O that is not encountered in training the KEDF model. The initial CO structure is generated using the RDKit software [74] which gives the bond length of $1.118\,\text{Å}$. We then augment four additional CO structures by adjusting bond lengths to $1.102\,\text{Å}$, $1.112\,\text{Å}$, $1.122\,\text{Å}$ and $1.132\,\text{Å}$, containing both squeezed and stretched bond lengths. The residual KEDF $T_{\text{S},\theta}$ model trained on QM9 training set is used to solve these systems. The Hückel method is chosen for density initialization, which exhibits better robustness to various bond lengths than the ProjMINAO initialization in our trials.

We evaluate the results in mean absolute error (MAE) with respect to KSDFT results. The Hirshfeld partial charge MAE and dipole moment MAE of optimized densities by M-OFDFT over the five CO structures are $0.102\,\mathrm{e}$ and $0.150\,\mathrm{D}$, respectively. As a reference, these MAE numbers are $0.296\,\mathrm{e}$ and $0.496\,\mathrm{D}$ when using the classical OFDFT with the TF+$\frac{1}{9}$vW KEDF. Hence M-OFDFT still achieves a substantial improvement over classical OFDFT even in this extrapolation scenario.

It should be noted that neither charged molecular systems nor the triple bond C≡O has been encountered in our training data, thus they are indeed challenging extrapolation tasks. While the formulation of M-OFDFT is designed to be universally applicable to all densities and molecular systems, its performance as a neural network model is hard to completely avoid extrapolation error. This is reflected in the (absolute) energy MAE of M-OFDFT on charged QM9 molecules, which stands at $3.79\,\mathrm{kcal/mol}$, higher than $0.73\,\mathrm{kcal/mol}$ on neutral molecules from the same dataset. Despite this, the extrapolation performance of M-OFDFT is still reasonable, and is still substantially better than classical OFDFT methods, showcasing the potential of M-OFDFT in these more challenging scenarios. The performance of M-OFDFT on these systems can be further improved by enriching the train data with charged molecules and new bond patterns such as triple bond C≡O, which will be investigated in future work.

Supplementary Section D.3 Empirical Time Cost Results

Settings As mentioned in Results 2.4, we aim to demonstrate the lower time complexity of M-OFDFT compared to KSDFT by empirically benchmarking the two methods on two types of large molecular systems. For KSDFT calculations on all systems, we adopt the same settings as introduced in Methods 4.6. M-OFDFT calculations are benchmarked on a 32-core CPU server with 216 GiB memory, which is equipped with one Nvidia A100 GPU with 80 GiB memory. Fig. 4 is produced by running M-OFDFT and KSDFT on QMugs molecules. For QMugs molecules, since they are generally much larger than ethanols and QM9 molecules to the extent that the residual KEDF $T_{\text{S,res},\theta}$ formulation becomes substantially costly due to grid-based computation for $T_{\text{S,base}}$ and E_{XC}, we apply the learned TXC functional model $E_{\text{TXC},\theta}$ in the same setting as is used to produce Fig. 3(a).

Time cost of each computational component To better understand the structure of the time cost in the density optimization process (that is, the process to use M-OFDFT to solve a queried molecular system), we split the time cost into various computational components in M-OFDFT. Both the residual KEDF $T_{\text{S,res},\theta}$ formulation (Supplementary Section B.4.1) and the TXC functional $E_{\text{TXC},\theta}$ formulation (Supplementary Section B.4.2) of M-OFDFT are considered. As shown in Supplementary Figure 18(a), in the $T_{\text{S,res},\theta}$ formulation, the three major parts of the time cost are the evaluation of the XC functional (denoted as "EXC"), the evaluation of the base KEDF (denoted as "Ts-Base"), and the evaluation of the $T_{\text{S,res},\theta}$ model (denoted as "ML-Pred"). Noting that the first two components are evaluated on grid, we conclude that grid-based computation is the main restriction to running M-OFDFT on large molecules, conforming to Supplementary Section B.4.2, hence the TXC functional $E_{\text{TXC},\theta}$ formulation is motivated, which does not require any grid-based computation. We also note that grid-based computation also occupies a substantial amount of GPU memory (Supplementary Section B.4.2). Using the hardware specified above, we can only afford systems of up to 230 electrons under the $T_{\text{S,res},\theta}$ formulation, which is where the plot ends. To compare the component-wise time cost of the $T_{\text{S,res},\theta}$ formulation and the $E_{\text{TXC},\theta}$ formulation, we conduct the same analysis with the $E_{\text{TXC},\theta}$ KEDF model. As shown in Supplementary Figure 18(b), due to the removal of grid-based computations, the total running time is substantially reduced. We note that in this case, the evaluation of the $E_{\text{TXC},\theta}$ model takes the largest computational cost. This part has an $O(N^2)$ computational complexity due to the need for nonlocal calculation. As the molecular size increases, this could lead to considerable computational demands. Despite the importance of the nonlocal calculation (Supplementary Section D.4.2), its influence presumably does not extend infinitely, thus allowing us to reduce the complexity by using a distance cutoff for large molecular systems. Specifically, with a distance cutoff r_c, the Transformer-based model, Graphormer, can be modified to capture nonlocal interactions between one atom and its neighboring atoms within the cutoff. The complexity is then

$O(AA_{\text{rc}})$, where A_{rc} is the average number of neighboring atoms within the distance cutoff r_c. As r_c is taken as a constant, A_{rc} is a constant. Hence the modification reduces the complexity of the model to linear: $O(AA_{\text{rc}}) = O(A) = O(N)$. We also note that analogous approaches to trim the neighborhood based on distance cutoffs have been utilized to achieve linear cost scaling using the Transformer architecture in the realm of machine learning [75, 76], which pave the way for further improvement of M-OFDFT.

Details on the time evaluation on proteins To further highlight the scaling advantage of M-OFDFT, we evaluate it on two large-scale protein molecular systems containing 709 and 738 atoms. The all-atom conformations of these systems are obtained from Lindorff-Larsen et al. [77] and neutralized following the same pipeline for processing Chignolin structures (detailed in Methods 4.5.4). Such a scale is already seldom encountered in the context of KSDFT calculations. Considering the substantial scale difference between the protein systems evaluated and the molecules used during training, we employ the total energy functional model $E_{\text{tot},\theta}(\mathbf{p}, \mathcal{M})$ trained on all Chignolin fragments and finetuned on 800 Chignolin structures. This model is expected to provide the best extrapolation capacity within our available resources.

The results indicate that M-OFDFT achieves a 25.6-fold and 27.4-fold speedup over KSDFT on the two protein systems, respectively. The per-atom energy MAE of M-OFDFT on the two test systems is 0.23 kcal/mol and 0.31 kcal/mol, respectively. While higher than the error on the Chignolin case, the result still demonstrates a substantial advantage over M-NNP, which gives per-atom energy MAE of 0.36 kcal/mol and 0.63 kcal/mol, respectively.

Supplementary Section D.4 Ablation Study

Supplementary Section D.4.1 Multi-Step Data and Gradient Label

To capture the energy landscape in the density coefficient space, we generate multiple $\mathbf{p}$ samples for each molecular structure $\mathcal{M}$ and compute the gradient label $\nabla_{\mathbf{p}} T_{\text{S}}$ for additional supervision information. An ablation experiment is conducted on the ethanol dataset to investigate the importance of each type of supervision by excluding the supervision label during the training process. The ablation results are presented in Supplementary Table 8. We observe that removing the gradient label $\nabla_{\mathbf{p}} T_{\text{S}}$ leads to a considerable decrease in energy and HF force accuracy for both density initialization strategies, emphasizing the importance of maintaining the optimization of M-OFDFT on a physical track. Incorporating multi-step $\mathbf{p}$ samples is also crucial for enhancing the performance of M-OFDFT, particularly for the ProjMINAO initialization, as the accuracy of the density corrector model depends on the size of training densities.

Supplementary Table 8: Ablation study for various data augmentation strategies. All results are evaluated on test ethanol molecules with the learned $T_{\text{S,res},\theta}$ model. The MAEs in energy and HF force are listed in kcal/mol and kcal/mol/Å, respectively.The results are calculated on ethanol test structures (n=10,000).

Multi-step $\mathbf{p}$	Gradient $\nabla_{\mathbf{p}} T_{\text{S}}$	Density Initialization	Energy	HF Force
✓	✓	Hückel	5.27	27.73
		ProjMINAO	**0.18**	**1.18**
✓		Hückel	166.84	144.78
		ProjMINAO	170.83	143.67
	✓	Hückel	6.43	32.83
		ProjMINAO	20.61	38.56

Supplementary Section D.4.2 Nonlocality

Considering the nonlocal dependency of KEDF on electron density, it is essential to incorporate nonlocal calculations in OFDFT, which motivates us to use Graphormer as our backbone architecture. To demonstrate the importance of nonlocality, we conduct an ablation study by gradually decreasing the receptive field (that is, the distance cutoff of atom neighborhood) of the functional model (specifically, the TXC model (Supplementary Section B.4.2)) of M-OFDFT. The experiment follows the extrapolation setting in parallel with Supplementary Table 6, that is, the functional model is trained on molecules with no more than 15 heavy atoms and the resulting M-OFDFT is tested on 50 QMugs molecular structures with 56-60 heavy atoms. As illustrated in Supplementary Figure 19, both the energy and HF force prediction error tend to worsen as the distance cutoff decreases, providing empirical evidence in our setting that a nonlocal model is indeed indispensable for well approximating a functional involving the kinetic energy density functional (KEDF).

Supplementary Section D.4.3 Local Frame and Enhancement Modules

As discussed in Methods 4.2, the input coefficients are tensors equivariant to rotation, but the output energy is a scalar hence invariant, which requires the model to have such invariance built-in. In addition, the vast range of gradient data makes optimizing the KEDF model challenging. A CoefficientAdapter module is proposed to guarantee the geometric invariance of the model and express the vast gradient range. We conducted an ablation experiment to study the importance of each component in this module. As shown in Supplementary Table 9, utilizing the local frame results in a considerable performance improvement compared to the baseline model using the standard global frame (for example, derived by PCA on the atom coordinates), demonstrating the effectiveness of the local frame in reducing the geometric variability of input data.

Furthermore, since different basis functions have varying importance in representing a density, some coefficient dimensions can substantially influence the density function and the electronic energy. This is difficult for M-OFDFT to handle, as the model treats each dimension equally. During density optimization, this influence is amplified when the input density is far from the ground state. We address this issue by introducing the natural reparameterization technique to balance the impact of each coefficient dimension. The results in Supplementary Table 9 show that implementing natural reparameterization can substantially improve the performance of the model, especially for the Hückel initialization, which is far from the ground-state density. Here, the energy MAE is reduced by one or two orders of magnitude, highlighting the powerful capability of natural reparameterization in balancing sensitivities across coefficient dimensions.

We also attempted to examine the importance of the other two enhancement modules, the atomic reference module and the dimension-wise rescaling module, but found it difficult to optimize the learning objective when removing either module from the model. Consequently, the unreasonable evaluation results are omitted in Supplementary Table 9. Moreover, the substantial impact on expressing a vast gradient range and the quantitative results for reducing geometric variability, as clarified in Supplementary Section B.3.1, also strongly support the necessity of these two modules.

Supplementary Table 9: Ablation study for components in the CoefficientAdapter module. All results are evaluated on test ethanol molecules with the learned $T_{\text{S,res},\theta}$ model. The MAEs in energy and HF force are listed in kcal/mol and kcal/mol/Å, respectively.The results are calculated on ethanol test structures (n=10,000).

Local Frame	Nat. Reparam.	Density Initialization	Energy	HF Force
		Hückel	567.69	73.17
		ProjMINAO	0.57	2.02
✓		Hückel	514.56	36.55
		ProjMINAO	0.30	1.29
	✓	Hückel	25.41	37.18
		ProjMINAO	**0.18**	1.81
✓	✓	Hückel	5.77	27.73
		ProjMINAO	**0.18**	**1.18**

Supplementary Section D.4.4 Results Using Other Training Strategies

As stressed in the main paper, one of the additional challenges for learning a functional model beyond conventional machine learning tasks is that the model is used to construct an objective function for optimizing the input, hence in addition to accurate end-to-end prediction on a discrete set of input queries, the model also needs to capture the output landscape (tendency of change in each locality) to properly guide the optimization. To address this challenge, we introduced techniques to generate multiple density datapoints each also with a gradient label for each molecular structure (Methods 4.1, Supplementary Section A.4), which provides a more holistic depiction of the output landscape. While there are alternative methods to regularize the optimization behavior in the context of learning the XC functional model [78–80, 38], our experiments indicate that they are not as effective for our task.

Kirkpatrick et al. [78] design an SCF loss based on second-order perturbation theory to regularize the functional stationary at provided self-consistent solutions. This effectively acts as a gradient supervision at the ground state, that is, the given self-consistent orbitals, for each molecular structure. The SCF loss itself is not directly suitable for learning the KEDF model as it is for supervising the gradient with respect to orbitals but not density (also explained by del Mazo-Sevillano and Hermann [67]). Approaches under the same spirit to supervise the gradient at the electronic ground state of each molecular structure are also explored in learning a KEDF model [61, 52, 53, 67]. In contrast, M-OFDFT introduces gradient labels on *multiple* electron density states, instead of only the ground-state density, for each molecular structure, which provides richer and broader landscape information, leading to a substantial performance improvement as shown in our ablation study results in Supplementary Table 8 (row 1 and row 3).

Other studies shape the output landscape and regularize the optimization behavior by directly supervising the model-optimized energy and density (or orbitals). This approach supervises the end goal, but unfortunately complicates the learning process of the model, since the dependency of the model-optimized results on model parameters is nested iteratively following an optimization iteration. Some works resort to gradient-free optimization methods (for example, evolutionary-style algorithms) to optimize the model parameters [79], which are not as efficient as gradient-based optimization. Li et al. [80] choose to directly invoke automatic differentiation on the loss with respect to model parameters through the optimization process of orbitals, which is, in the context of learning the XC functional for KSDFT, typically conducted through the self-consistency field (SCF) iteration that solves the Kohn-Sham equations, hence called the Kohn-Sham regularizer. We tried its counterpart for learning the KEDF for OFDFT, by supervising the model-optimized energy and density after the density optimization process using gradient descent (note SCF iteration is unnatural here), and then applying automatic differentiation through the gradient-descent iteration. In our experiments on QM9, we found this incurs substantially more computational cost for computing the gradient for model parameters. Even only using the E_{TXC} version of functional model (Supplementary Section B.4.2) to get rid of the costly grid-based computation, only a few number of density optimization steps up to 8 is affordable. With this maximally affordable cost, this strategy could work when applied to an end-to-end pretrained model on the energy and gradient labels, as we observed improved energy accuracy on held-out test molecules using 8 steps of density optimization. However, an undesired observation indicates that the model learned in this way still does not meet the goal: the density optimization process often does not converge, and even when it does, the accuracy is much worse than that at the 8-th step. This makes the method, instead of learning a density functional that holds a physical meaning, more like an end-to-end ground-state energy predictor constructed by unrolling the machine-learning model 8 times through the gradient descent process.

Under the same idea to supervise model-optimized energy, Chen et al. [38] find it is possible to avoid automatic differentiation through the iterative orbital optimization process if the resulting energy $E_\theta(\mathbf{C}^\star_\theta)$ is indeed optimal in the view of the current functional model, since $\nabla_\theta E_\theta(\mathbf{C}^\star_\theta) = (\nabla_\theta E_\theta)\big|_{\mathbf{C}^\star_\theta} + (\nabla_\theta \bar{\mathbf{C}}^\star_\theta)^\top (\nabla_{\bar{\mathbf{C}}} E_\theta)\big|_{\bar{\mathbf{C}}^\star_\theta} = (\nabla_\theta E_\theta)\big|_{\mathbf{C}^\star_\theta}$ ($\bar{\mathbf{C}}$ denotes the vector of flattened $\mathbf{C}$ matrix, and $\nabla_\theta \bar{\mathbf{C}}^\star_\theta$ is the Jacobian matrix) as the optimality condition indicates $(\nabla_{\bar{\mathbf{C}}} E_\theta)\big|_{\bar{\mathbf{C}}^\star_\theta} = 0$ on the admissible space of orbital coefficients, so $\nabla_\theta \bar{\mathbf{C}}^\star_\theta$ is avoided. The same argument also applies to the optimization process of density coefficients. Since finding the optimal orbitals after every model update is expensive, the method opts to optimize the model and density alternately. However, this modification biases the orbital/density optimality thus the optimization process may also deviate from being effective. In our experiment on Chignolin, we iterated the alternate optimization for four steps (and even carefully optimized the density in each step with a large cost), but observed only marginal performance improvement. Specifically, the performance gains from each step are 0.5%, -0.06%, 1.7%, and -0.8%, respectively, where the negative signs indicate that some steps even worsen the results.

References

[1] Mel Levy. 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. A.1

[2] Llewellyn H Thomas. The calculation of atomic fields. In *Mathematical proceedings of the Cambridge philosophical society*, volume 23, pages 542–548. Cambridge University Press, 1927. A.1, B.5.1

[3] Enrico Fermi. Eine statistische methode zur bestimmung einiger eigenschaften des atoms und ihre anwendung auf die theorie des periodischen systems der elemente. *Zeitschrift für Physik*, 48(1): 73–79, 1928. B.5.1

[4] John C Slater. A simplification of the Hartree-Fock method. *Physical review*, 81(3):385, 1951. A.1

[5] Pierre Hohenberg and Walter Kohn. Inhomogeneous electron gas. *Physical review*, 136(3B):B864, 1964. A.1

[6] Elliott H Lieb. Density functionals for Coulomb systems. *International Journal of Quantum Chemistry*, 24(3):243–277, 1983. A.1, A.1, A.1, B.4.1, D.1.1

[7] Walter Kohn and Lu Jeu Sham. Self-consistent equations including exchange and correlation effects. *Physical review*, 140(4A):A1133, 1965. A.1

[8] Frank W. Bobrowicz and William A. Goddard. *The Self-Consistent Field Equations for Generalized Valence Bond and Open-Shell Hartree-Fock Wave Functions*, pages 79–127. Springer US, Boston, MA, 1977. ISBN 978-1-4757-0887-5. doi: 10.1007/978-1-4757-0887-5_4. URL https://doi.org/10.1007/978-1-4757-0887-5_4. A.1

[9] Ira N Levine, Daryle H Busch, and Harrison Shull. *Quantum chemistry*, volume 6. Pearson Prentice Hall Upper Saddle River, NJ, 2009. A.1, A.5

[10] SM Blinder. Basic concepts of self-consistent-field theory. *American journal of physics*, 33(6): 431–443, 1965. A.2

[11] P. Pulay. 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. A.2, A.3.1

[12] Konstantin N. Kudin, Gustavo E. Scuseria, and Eric Cancès. 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. A.2, A.3.1

[13] Michel Dupuis, John Rys, and Harry F. King. Evaluation of molecular integrals over Gaussian basis functions. *The Journal of Chemical Physics*, 65(1):111–116, 07 1976. ISSN 0021-9606. doi: 10.1063/1.432807. URL https://doi.org/10.1063/1.432807. A.3.2

[14] J. Rys, M. Dupuis, and H. F. King. Computation of electron repulsion integrals using the rys quadrature method. *Journal of Computational Chemistry*, 4(2):154–157, 1983. doi: https://doi.org/10.1002/jcc.540040206. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/jcc.540040206. A.3.2

[15] Qiming Sun. Libcint: An efficient general integral library for Gaussian basis functions. *Journal of computational chemistry*, 36(22):1664–1671, 2015. A.3.2

[16] Qiming Sun, Timothy C Berkelbach, Nick S Blunt, George H Booth, Sheng Guo, Zhendong Li, Junzi Liu, James D McClain, Elvira R Sayfutyarova, Sandeep Sharma, et al. PySCF: the Python-based simulations of chemistry framework. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 8(1):e1340, 2018. A.3.2, B.5.1, D.1.1

[17] Atilim Gunes Baydin, Barak A Pearlmutter, Alexey Andreyevich Radul, and Jeffrey Mark Siskind. Automatic differentiation in machine learning: a survey. *Journal of Marchine Learning Research*, 18:1–43, 2018. A.3.2

[18] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. *Advances in neural information processing systems*, 32, 2019. A.3.2, B.1, B.1.6

[19] John P Perdew, Kieron Burke, and Matthias Ernzerhof. Generalized gradient approximation made simple. *Physical review letters*, 77(18):3865, 1996. A.3.2

[20] Chuin Wei Tan, Chris J. Pickard, and William C. Witt. Automatic differentiation for orbital-free density functional theory. *The Journal of Chemical Physics*, 158(12):124801, 03 2023. ISSN 0021-9606. doi: 10.1063/5.0138429. URL https://doi.org/10.1063/5.0138429. A.3.2

[21] Lucian A Constantin, E Fabiano, S Laricchia, and F Della Sala. Semiclassical neutral atom as a reference system in density functional theory. *Physical review letters*, 106(18):186406, 2011. A.3.2, B.2, B.4.1, B.5.1

[22] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. A.3.2, B.1.4, B.1.4

[23] Brett I. Dunlap. Robust and variational fitting. *Phys. Chem. Chem. Phys.*, 2:2113–2116, 2000. doi: 10.1039/B000027M. URL http://dx.doi.org/10.1039/B000027M. A.4.1

[24] Hans Hellman. Einführung in die Quantenchemie. *Franz Deuticke, Leipzig*, 285, 1937. A.5, A.5

[25] R. P. Feynman. Forces in molecules. *Phys. Rev.*, 56:340–343, Aug 1939. doi: 10.1103/PhysRev.56.340. URL https://link.aps.org/doi/10.1103/PhysRev.56.340. A.5, A.5

[26] Peter Pulay. Ab initio calculation of force constants and equilibrium geometries in polyatomic molecules: I. Theory. *Molecular Physics*, 17(2):197–204, 1969. A.5, A.5, A.5

[27] Stefan Chmiela, Alexandre Tkatchenko, Huziel E Sauceda, Igor Poltavsky, Kristof T Schütt, and Klaus-Robert Müller. Machine learning of accurate energy-conserving molecular force fields. *Science advances*, 3(5):e1603015, 2017. A.5, D.1.1

[28] Kristof T Schütt, Huziel E Sauceda, P-J Kindermans, Alexandre Tkatchenko, and K-R Müller. SchNet – a deep learning architecture for molecules and materials. *The Journal of Chemical Physics*, 148(24), 2018. A.5

[29] Robert G Parr and Weitao Yang. Density-functional theory of atoms and molecules. 1989. A.6

[30] Jacob Hollingsworth, Li Li, Thomas E Baker, and Kieron Burke. Can exact conditions improve machine-learned density functionals? *The Journal of chemical physics*, 148(24):241743, 2018. A.6, C

[31] Bhupalee Kalita, Li Li, Ryan J McCarty, and Kieron Burke. Learning to approximate density functionals. *Accounts of Chemical Research*, 54(4):818–826, 2021. A.6, C

[32] P. García-González, J. E. Alvarellos, and E. Chacón. Nonlocal kinetic-energy-density functionals. *Phys. Rev. B*, 53:9509–9512, Apr 1996. doi: 10.1103/PhysRevB.53.9509. URL https://link.aps.org/doi/10.1103/PhysRevB.53.9509. B.1

[33] Yan Alexander Wang, Niranjan Govind, and Emily A Carter. Orbital-free kinetic-energy density functionals with a density-dependent kernel. *Physical Review B*, 60(24):16350, 1999. B.5.1

[34] Yan Alexander Wang and Emily A Carter. Orbital-free kinetic-energy density functional theory. *Theoretical methods in condensed phase chemistry*, 5:117–84, 2000.

[35] Wenhui Mi, Alessandro Genova, and Michele Pavanello. Nonlocal kinetic energy functionals by functional integration. *The Journal of Chemical Physics*, 148(18):184107, 05 2018. ISSN 0021-9606. doi: 10.1063/1.5023926. URL https://doi.org/10.1063/1.5023926.

[36] Andrew M Teale, Trygve Helgaker, Andreas Savin, Carlo Adamo, Bálint Aradi, Alexei V Arbuznikov, Paul W Ayers, Evert Jan Baerends, Vincenzo Barone, Patrizia Calaminici, et al. DFT exchange: sharing perspectives on the workhorse of quantum chemistry and materials science. *Physical chemistry chemical physics*, 24(47):28700–28781, 2022. B.1

[37] Sebastian Dick and Marivi Fernandez-Serra. Machine learning accurate exchange and correlation functionals of the electronic density. *Nature communications*, 11(1):3509, 2020. B.1

[38] Yixiao Chen, Linfeng Zhang, Han Wang, and Weinan E. DeePKS: A comprehensive data-driven approach toward chemically accurate density functional theory. *Journal of Chemical Theory and Computation*, 17(1):170–181, 2021. B.1, D.4.4

[39] Chengxuan Ying, Tianle Cai, Shengjie Luo, Shuxin Zheng, Guolin Ke, Di He, Yanming Shen, and Tie-Yan Liu. Do Transformers really perform badly for graph representation? *Advances in Neural Information Processing Systems*, 34:28877–28888, 2021. B.1, B.1.6, B.1.7

[40] Yu Shi, Shuxin Zheng, Guolin Ke, Yifei Shen, Jiacheng You, Jiyan He, Shengjie Luo, Chang Liu, Di He, and Tie-Yan Liu. Benchmarking Graphormer on large-scale molecular modeling datasets. *arXiv preprint arXiv:2203.04810*, 2022. B.1, B.1, B.1.4

[41] Shuxin Zheng, Jiyan He, Chang Liu, Yu Shi, Ziheng Lu, Weitao Feng, Fusong Ju, Jiaxi Wang, Jianwei Zhu, Yaosen Min, Zhang He, Shidi Tang, Hongxia Hao, Peiran Jin, Chi Chen, Frank Noé, Haiguang Liu, and Tie-Yan Liu. Towards predicting equilibrium distributions for molecular systems with deep learning. *arXiv preprint arXiv:2306.05445*, 2023. B.1

[42] Richard D. Bardo and Klaus Ruedenberg. Even-tempered atomic orbitals. VI. optimal orbital exponents and optimal contractions of Gaussian primitives for hydrogen, carbon, and oxygen in molecules. *The Journal of Chemical Physics*, 60(3):918–931, 1974. doi: 10.1063/1.1681168. URL https://doi.org/10.1063/1.1681168. B.1.1

[43] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. *Neural networks*, 2(5):359–366, 1989. B.1.3

[44] Dan Hendrycks and Kevin Gimpel. Gaussian error linear units (GELUs). *arXiv preprint arXiv:1606.08415*, 2016. B.1.3

[45] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016. B.1.4

[46] Jiequn Han, Linfeng Zhang, Roberto Car, et al. Deep Potential: A general representation of a many-body potential energy surface. *Communications in Computational Physics*, 23(3), 2018. B.2

[47] Feng Changyong, Wang Hongyue, Lu Naiji, Chen Tian, He Hua, Lu Ying, et al. Log-transformation and its implications for data analysis. *Shanghai archives of psychiatry*, 26(2):105, 2014. B.3

[48] Albert Musaelian, Simon Batzner, Anders Johansson, Lixin Sun, Cameron J Owen, Mordechai Kornbluth, and Boris Kozinsky. Learning local equivariant representations for large-scale atomistic dynamics. *Nature Communications*, 14(1):579, 2023. B.3.3

[49] CF von Weizsäcker. Zur theorie der kernmassen. *Zeitschrift für Physik*, 96(7):431–458, 1935. B.4.1, D.1.1

[50] M. Brack, B.K. Jennings, and Y.H. Chu. On the extended Thomas-Fermi approximation to the kinetic energy density. *Physics Letters B*, 65(1):1–4, 1976. ISSN 0370-2693. doi: https://doi.org/10.1016/0370-2693(76)90519-0. URL https://www.sciencedirect.com/science/article/pii/0370269376905190. B.5.1

[51] Valentin V Karasiev and Samuel B Trickey. Issues and challenges in orbital-free density functional calculations. *Computer Physics Communications*, 183(12):2519–2527, 2012. B.5.1

[52] Mikito Fujinami, Ryo Kageyama, Junji Seino, Yasuhiro Ikabata, and Hiromi Nakai. Orbital-free density functional theory calculation applying semi-local machine-learned kinetic energy density functional and kinetic potential. *Chemical Physics Letters*, 748:137358, 2020. B.5.1, C, D.4.4

[53] Fumihiro Imoto, Masatoshi Imada, and Atsushi Oshiyama. Order-N orbital-free density-functional calculations with machine learning of functional derivatives for semiconductors and metals. *Physical Review Research*, 3(3):033198, 2021. C, D.4.4

[54] Roman Remme, Tobias Kaczun, Maximilian Scheurer, Andreas Dreuw, and Fred A Hamprecht. KineticNet: Deep learning a transferable kinetic energy functional for orbital-free density functional theory. *arXiv preprint arXiv:2305.13316*, 2023. B.5.1, C

[55] Roald Hoffmann. An Extended Hückel Theory. I. Hydrocarbons. *The Journal of Chemical Physics*, 39(6):1397–1412, 06 1963. ISSN 0021-9606. doi: 10.1063/1.1734456. URL https://doi.org/10.1063/1.1734456. B.5.2

[56] Susi Lehtola. Assessment of initial guesses for self-consistent field calculations. Superposition of atomic potentials: Simple yet efficient. *Journal of chemical theory and computation*, 15(3):1593–1604, 2019. B.5.2, D.1.1

[57] John C Snyder, Matthias Rupp, Katja Hansen, Klaus-Robert Müller, and Kieron Burke. Finding density functionals with machine learning. *Physical review letters*, 108(25):253002, 2012. B.5.2, B.5.2, C

[58] Felix Brockherde, Leslie Vogt, Li Li, Mark E Tuckerman, Kieron Burke, and Klaus-Robert Müller. Bypassing the Kohn-Sham equations with machine learning. *Nature communications*, 8(1):1–10, 2017. B.5.2, B.5.2, C

[59] John C Snyder, Matthias Rupp, Katja Hansen, Leo Blooston, Klaus-Robert Müller, and Kieron Burke. Orbital-free bond breaking via machine learning. *The Journal of chemical physics*, 139(22): 224104, 2013. B.5.2, B.5.2, C

[60] Li Li, Thomas E Baker, Steven R White, Kieron Burke, et al. Pure density functional for strong correlation and the thermodynamic limit from machine learning. *Physical Review B*, 94(24):245129, 2016. C

[61] Ralf Meyer, Manuel Weichselbaum, and Andreas W Hauser. Machine learning approaches toward orbital-free density functional theory: Simultaneous training on the kinetic energy density functional and its functional derivative. *Journal of chemical theory and computation*, 16(9):5685–5694, 2020. C, D.4.4

[62] Mohammed Alghadeer, Abdulaziz Al-Aswad, and Fahhad H Alharbi. Highly accurate machine learning model for kinetic energy density functional. *Physics Letters A*, 414:127621, 2021. C

[63] Shashikant Kumar, Edgar Landinez Borda, Babak Sadigh, Siya Zhu, Sebastian Hamel, Brian Gallagher, Vasily Bulatov, John Klepeis, and Amit Samanta. Accurate parameterization of the kinetic energy functional. *The Journal of Chemical Physics*, 156(2):024110, 2022. C

[64] Junji Seino, Ryo Kageyama, Mikito Fujinami, Yasuhiro Ikabata, and Hiromi Nakai. Semi-local machine-learned kinetic energy density functional with third-order gradients of electron density. *The Journal of Chemical Physics*, 148(24), 2018. C

[65] Junji Seino, Ryo Kageyama, Mikito Fujinami, Yasuhiro Ikabata, and Hiromi Nakai. Semi-local machine-learned kinetic energy density functional demonstrating smooth potential energy curves. *Chemical Physics Letters*, 734:136732, 2019. C, D.1.1

[66] Pavlo Golub and Sergei Manzhos. Kinetic energy densities based on the fourth order gradient expansion: performance in different classes of materials and improvement via machine learning. *Phys. Chem. Chem. Phys.*, 21:378–395, 2019. doi: 10.1039/C8CP06433D. URL `http://dx.doi.org/10.1039/C8CP06433D`. C

[67] Pablo del Mazo-Sevillano and Jan Hermann. Variational principle to regularize machine-learned density functionals: the non-interacting kinetic-energy functional. *arXiv preprint arXiv:2306.17587*, 2023. C, D.4.4

[68] Kun Yao and John Parkhill. Kinetic energy of hydrocarbons as a function of electron density and convolutional neural networks. *Journal of chemical theory and computation*, 12(3):1139–1147, 2016. C, D.1.1

[69] Kevin Ryczko, Sebastian J Wetzel, Roger G Melko, and Isaac Tamblyn. Orbital-free density functional theory with small datasets and deep learning. *arXiv preprint arXiv:2104.05408*, 2021. C

[70] Stefan Chmiela, Huziel E Sauceda, Igor Poltavsky, Klaus-Robert Müller, and Alexandre Tkatchenko. sGDML: Constructing accurate and data efficient molecular force fields using machine learning. *Computer Physics Communications*, 240:38–45, 2019. D.1.1

[71] F. L. Hirshfeld. Bonded-atom fragments for describing molecular charge densities. *Theoretica chimica acta*, 44(2):129–138, 1977. doi: 10.1007/BF00549096. URL `https://doi.org/10.1007/BF00549096`. D.1.2, D.2.1

[72] Philipp Thölke and Gianni De Fabritiis. Equivariant transformers for neural network based molecular potentials. In *International Conference on Learning Representations*, 2021. D.2.1, 17

[73] Yi-Lun Liao and Tess Smidt. Equiformer: Equivariant graph attention transformer for 3d atomistic graphs. *arXiv preprint arXiv:2206.11990*, 2022. D.2.1, 17

[74] Landrum Greg, Tosco Paolo, Kelley Brian, Ric, and Cosgrove David. RDKit: Open-source cheminformatics. 2013. doi: 10.5281/zenodo.591637. URL `https://www.rdkit.org`. D.2.2

[75] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020. D.3

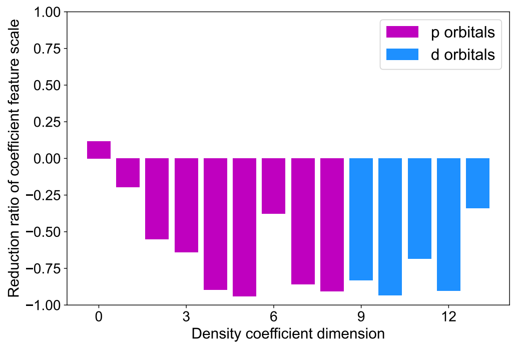
[76] Jianwei Yang, Chunyuan Li, Pengchuan Zhang, Xiyang Dai, Bin Xiao, Lu Yuan, and Jianfeng Gao. Focal self-attention for local-global interactions in vision transformers. *arXiv preprint arXiv:2107.00641*, 2021. D.3

[77] Kresten Lindorff-Larsen, Stefano Piana, Ron O Dror, and David E Shaw. How fast-folding proteins fold. *Science*, 334(6055):517–520, 2011. D.3

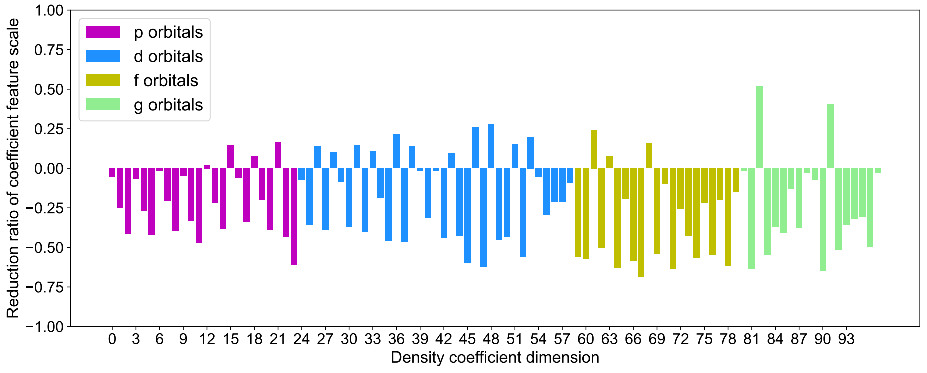
[78] James Kirkpatrick, Brendan McMorrow, David HP Turban, Alexander L Gaunt, James S Spencer, Alexander GDG Matthews, Annette Obika, Louis Thiry, Meire Fortunato, David Pfau, et al. Pushing the frontiers of density functionals by solving the fractional electron problem. *Science*, 374(6573): 1385–1389, 2021. D.4.4

[79] Ryo Nagai, Ryosuke Akashi, and Osamu Sugino. Completing density functional theory by machine learning hidden messages from molecules. *npj Computational Materials*, 6(1):1–8, 2020. D.4.4

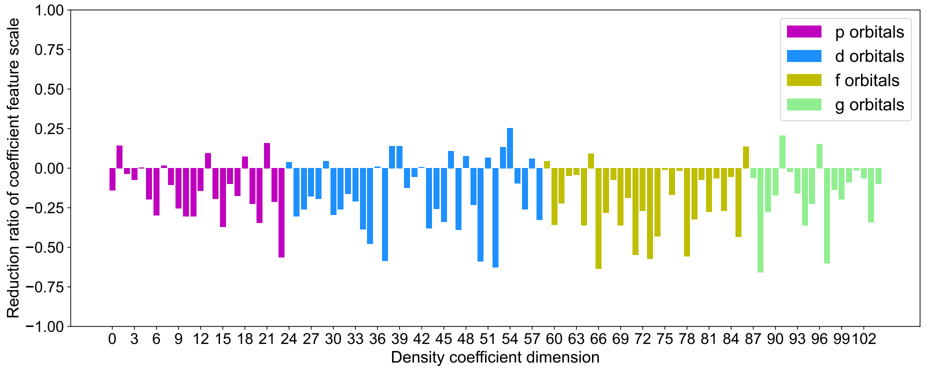
[80] Li Li, Stephan Hoyer, Ryan Pederson, Ruoxi Sun, Ekin D Cubuk, Patrick Riley, Kieron Burke, et al. Kohn-Sham equations as regularizer: Building prior knowledge into machine-learned physics. *Physical review letters*, 126(3):036401, 2021. D.4.4



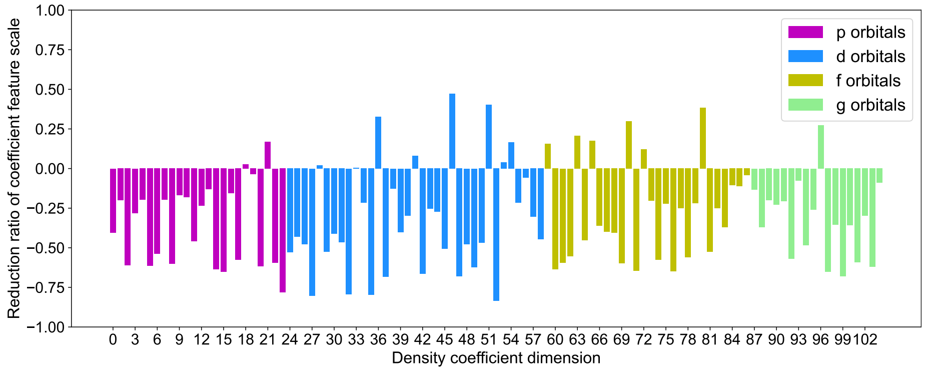
(a) H atom



(b) C atom

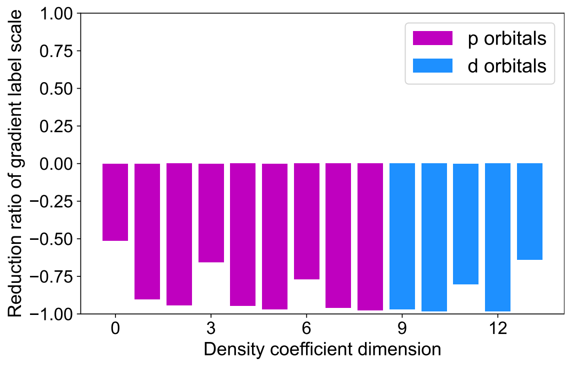


(c) N atom

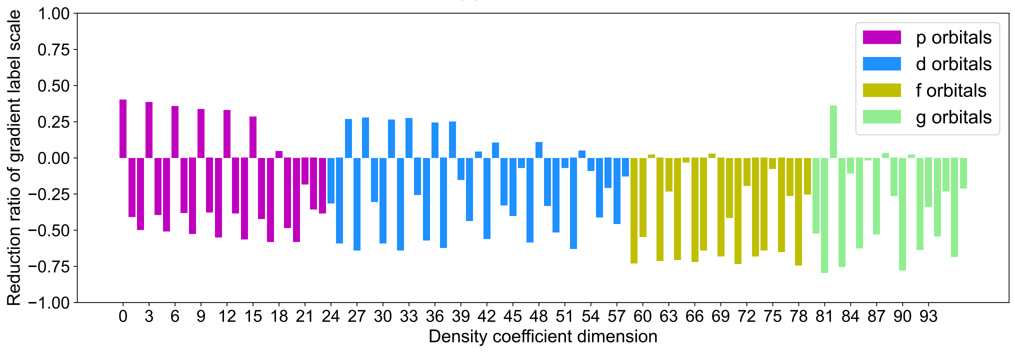


(d) O atom

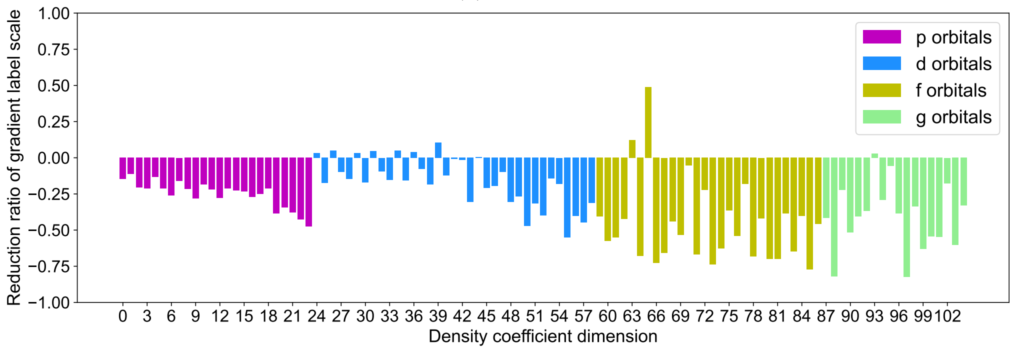
Supplementary Figure 4: Coefficient scale changes by the use of the local frame. For each coefficient dimension τ of atom type Z, the height of the bar represents the relative change in scale, calculated as $\frac{\text{std\_coeff}'_{Z,\tau} - \text{std\_coeff}_{Z,\tau}}{\text{std\_coeff}_{Z,\tau}}$, where $\text{std\_coeff}_{Z,\tau} := \text{std}\{\mathbf{p}^{(d,k)}_{a,\tau}\}_{a:Z^{(a)}=Z,\,k,\,d}$ and $\text{std\_coeff}'_{Z,\tau} := \text{std}\{\mathbf{p}'^{(d,k)}_{a,\tau}\}_{a:Z^{(a)}=Z,\,k,\,d}$ denote the coefficient scale before and after being transformed by the local frame, respectively. A negative value indicates a reduction in the scale of the coefficient due to the local frame transformation. Importantly, the local frame substantially decreases the coefficient scale across diverse basis dimensions and atom types.



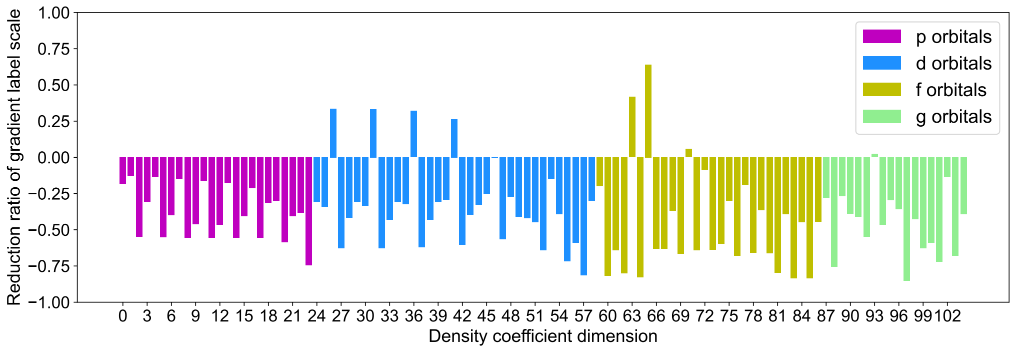
(a) H atom



(b) C atom

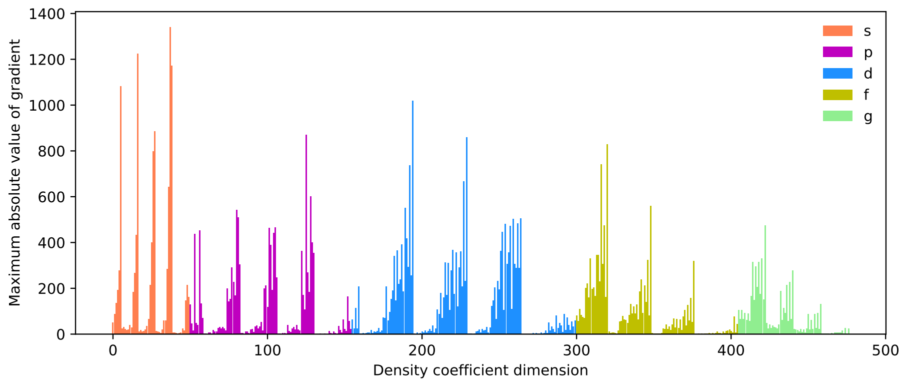


(c) N atom

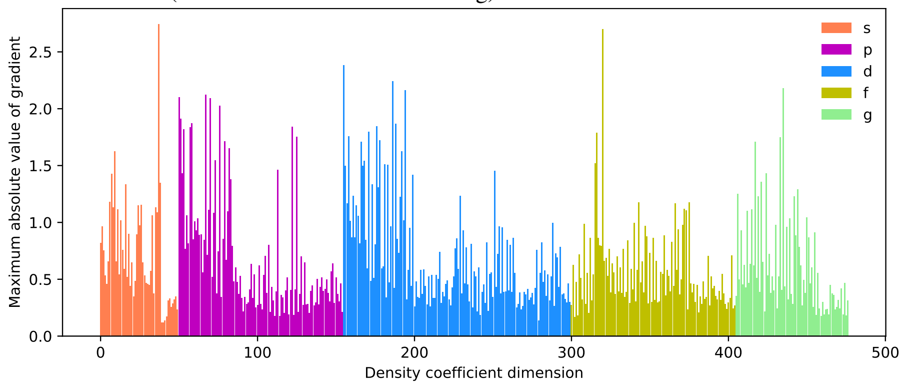


(d) O atom

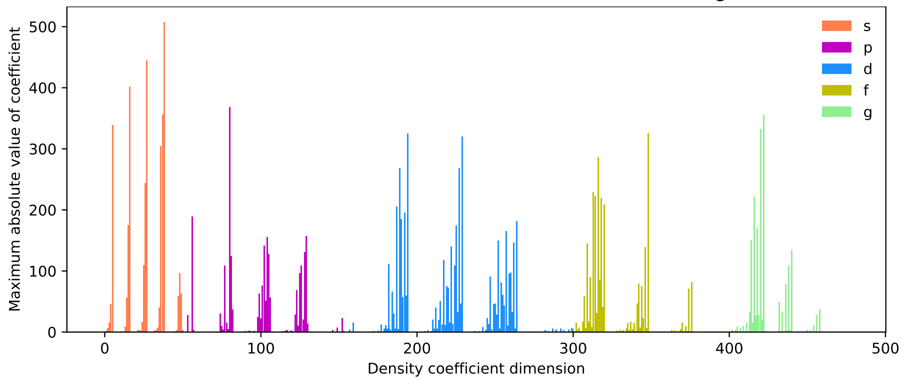
Supplementary Figure 5: Gradient scale changes by the use of the local frame. For each coefficient dimension τ of the atom type Z, the height of the bar represents the relative change in gradient scale, calculated as $\frac{\text{max\_grad}'_{Z,\tau} - \text{max\_grad}_{Z,\tau}}{\text{max\_grad}_{Z,\tau}}$, where $\text{max\_grad}_{Z,\tau} := \max\{\nabla_{\mathbf{p}_{a,\tau}} T_{\mathrm{S}}^{(d,k)}\}_{a:Z^{(a)}=Z,\, k,\, d}$ and $\text{max\_grad}'_{Z,\tau} := \max\{\nabla_{\mathbf{p}_{a,\tau}} T'^{(d,k)}_{\mathrm{S}}\}_{a:Z^{(a)}=Z,\, k,\, d}$ denote the gradient scale before and after being transformed by the local frame, respectively. The local frame transformation notably reduces the gradient scale across a wide range of basis dimensions and atom types.



(a) Gradient scales after processed by local frame, natural reparameterization, and atomic reference model (before dimension-wise rescaling).

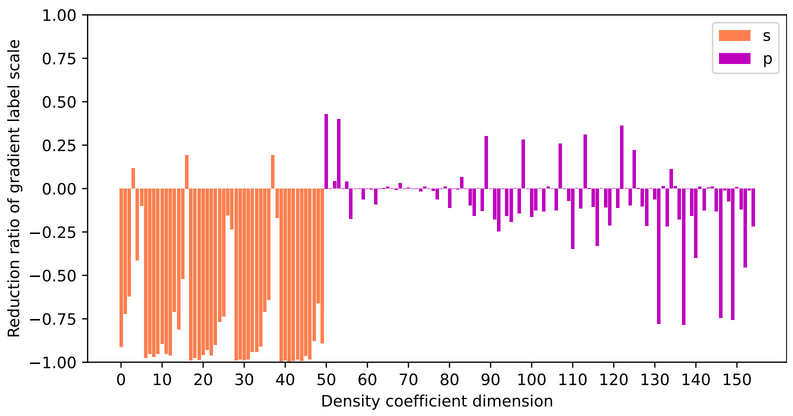


(b) Gradient scales further after dimension-wise rescaling.

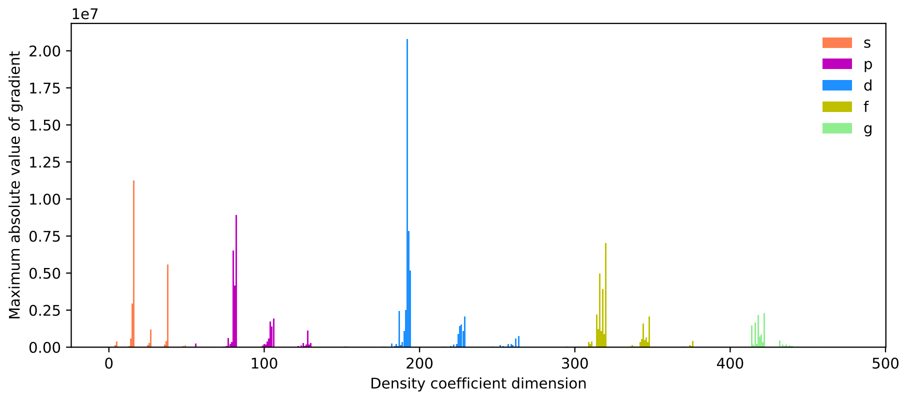


(c) Density coefficient scales further after dimension-wise rescaling.

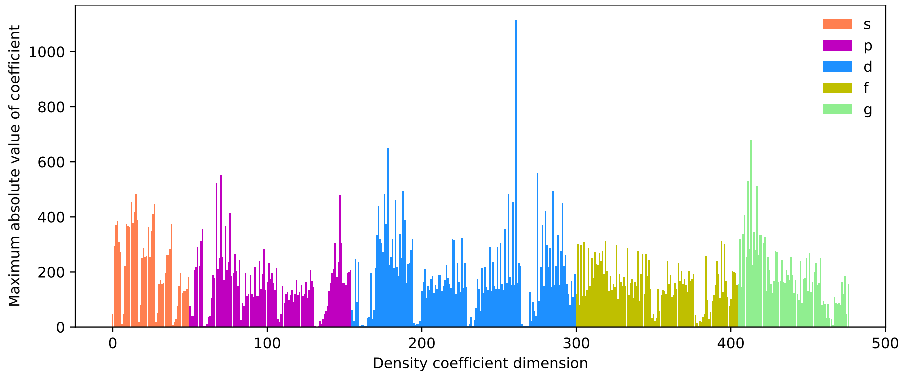
Supplementary Figure 6: Gradient and density coefficient scales over dimensions on the QM9 dataset in the setting of learning residual KEDF with APBE base KEDF. The maximum absolute value is used to measure the maximum scale of data. **(a)-(b)** present the gradient scale before and after the dimension-wise rescaling transformation. This technique properly balances the scales of gradient and density coefficient in each dimension, hence reduces the gradient scale for easier learning, while maintaining a reasonable density coefficient scale. **(c)** shows the corresponding coefficient scale after the dimension-wise rescaling transformation.



Supplementary Figure 7: Gradient scale changes by the use of the atomic reference module. For coefficient dimension τ of the atom type Z, the height of the bar is also calculated by $\frac{\text{max\_grad}'_{Z,\tau} - \text{max\_grad}_{Z,\tau}}{\text{max\_grad}_{Z,\tau}}$, where $\text{max\_grad}_{Z,\tau} := \max\{\nabla_{\mathbf{p}_{a,\tau}} T_{\mathrm{S}}^{(d,k)}\}_{a:Z^{(a)}=Z,\, k,\, d}$ and $\text{max\_grad}'_{Z,\tau} := \max\{\nabla_{\mathbf{p}_{a,\tau}} T'^{(d,k)}_{\mathrm{S}}\}_{a:Z^{(a)}=Z,\, k,\, d}$ denote the gradient scale before and after being centralized by the atomic reference module, respectively. The atomic reference module considerably covers the vast gradient scale.

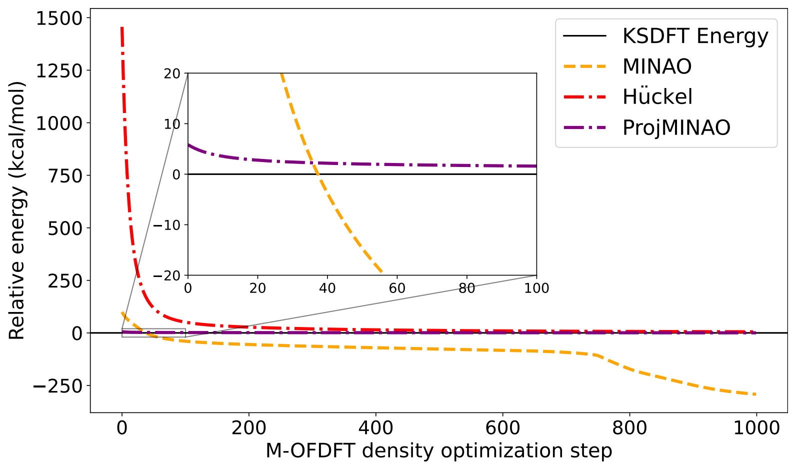


(a) Gradient scales after processed by local frame, natural reparameterization, atomic reference model, and dimension-wise rescaling, in parallel with Supplementary Figure 6(b).

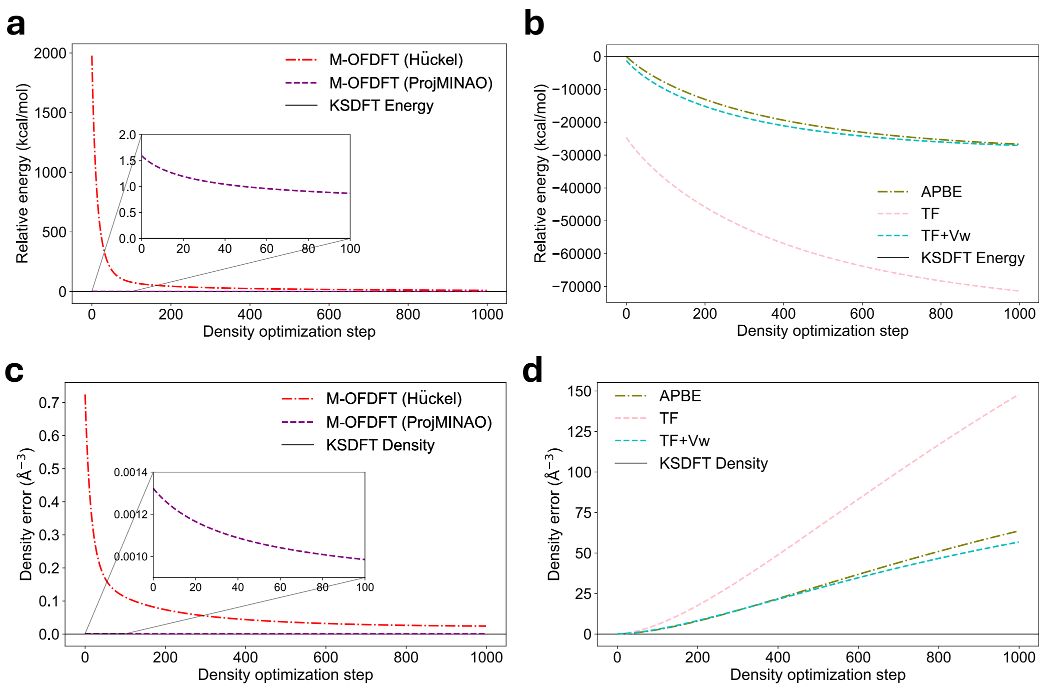


(b) Density coefficient scales after processed by local frame, natural reparameterization, atomic reference model, and dimension-wise rescaling, in parallel with Supplementary Figure 6(c).

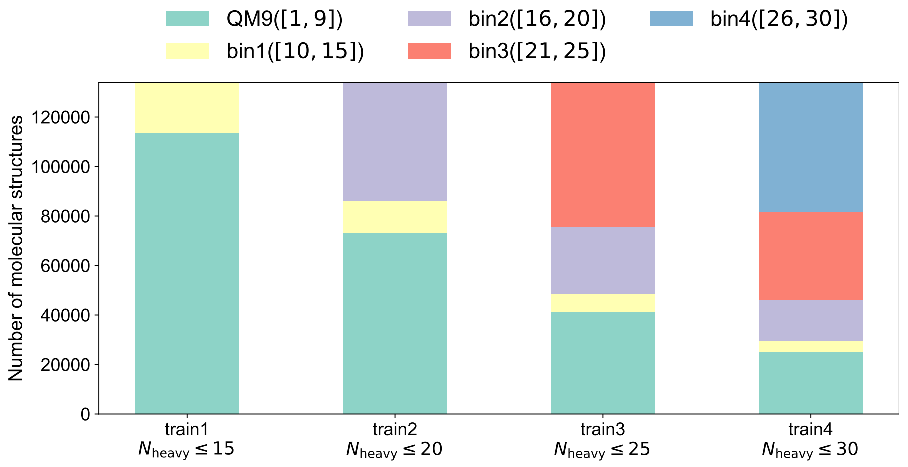
Supplementary Figure 8: Gradient and density coefficient scales over dimensions on the QM9 dataset in the setting of learning residual KEDF with the vW base KEDF. The maximum absolute value is used to measure the maximum scale of data. **(a)** and **(b)** respectively present the gradient scales and density coefficient scales after processed by local frame and all enhancement modules (natural reparameterization, atomic reference model, and dimension-wise rescaling), in parallel with Supplementary Figure 6(b) and (c) for APBE base KEDF respectively. Although these techniques have reduced the original gradient scale under a reasonable density scale, the processed gradient scale is still exceedingly large $\sim 10^7$ for a neural network to learn.



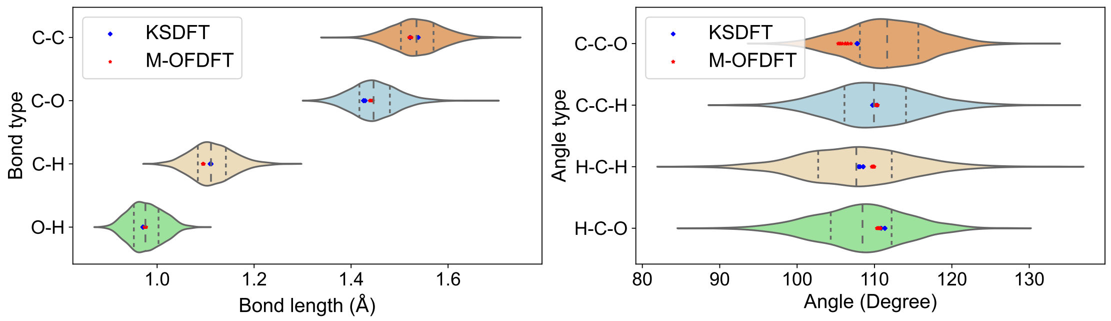
Supplementary Figure 9: Typical density optimization curves of M-OFDFT for a QM9 molecule with different initialization methods. MINAO, the common KSDFT initialization, leads the optimization to a large gap from the target energy, since it is not from the eigensolution to an effective Hamiltonian matrix, hence lies off the training-data manifold (out of distribution). Hückel initialization solves an eigenvalue problem, which indeed converges the curve with a much smaller energy error of 5.34 $\mathrm{kcal/mol}$. ProjMINAO initialization uses a deep-learning model to project the MINAO density onto the training-data manifold, which also converges the curve and achieves the best result of 0.60 $\mathrm{kcal/mol}$ energy error. The inset figure highlights the role of density optimization even though the ProjMINAO density is close to the ground-state density.



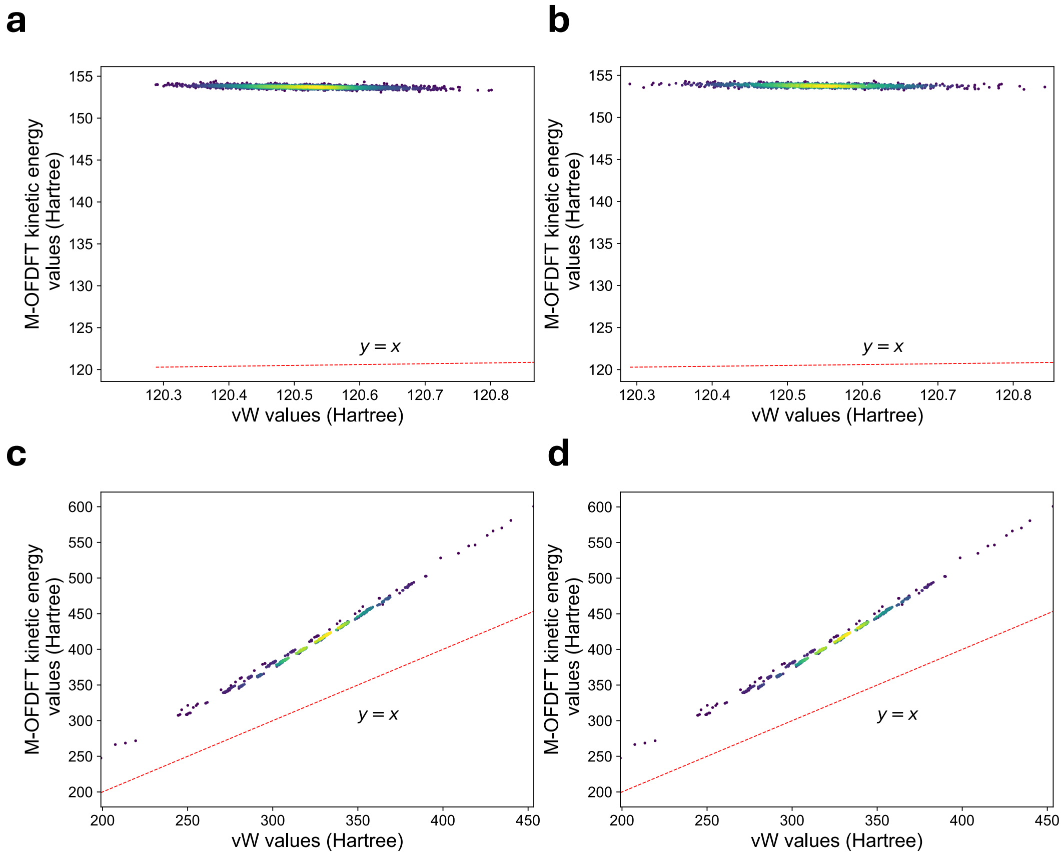
Supplementary Figure 10: Density optimization curves by M-OFDFT and OFDFT with classical KEDFs. The figures come in parallel with Fig. 9 but on a different QM9 molecule. **(a)** Curves in relative energy along the density optimization process of M-OFDFT with different initialization methods. M-OFDFT converges closely to the true ground-state energy using either initialization method. **(b)** Curves in relative energy along the density optimization process of OFDFT using classical KEDFs. The curves run below the true ground-state energy with a large gap. **(c)** Curves in density error along the density optimization process of M-OFDFT with different initialization methods. M-OFDFT converges closely to the true ground-state density using either initialization method, even though the optimization is not driven by minimizing the density error. **(d)** Curves in density error along the density optimization process of OFDFT using classical KEDFs. The curves diverge quickly. Curves for TF+$\frac{1}{9}$vW are omitted in **(b)** and **(d)** since they soon run out of the shown range.



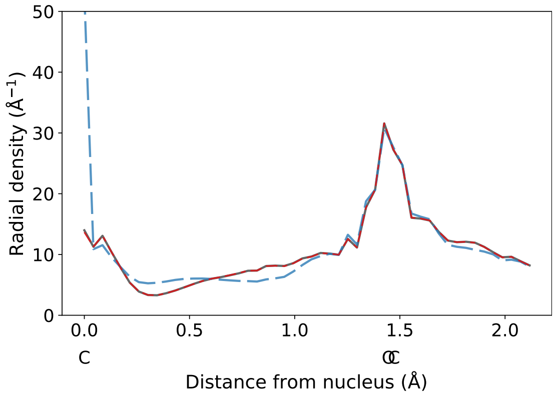
Supplementary Figure 11: Compositions of training datasets from QM9 and QMugs for the extrapolation setting in Fig. 3(b). The training datasets contain the same number of datapoints (molecular structures), while involve increasingly larger molecules. The proportion of molecules in each scale range (that is, QM9, bin1-bin4) respects the proportion in the joint dataset. The range of the number of heavy atoms in each data source is listed in the figure (that is, $[1, 9]$ for QM9).



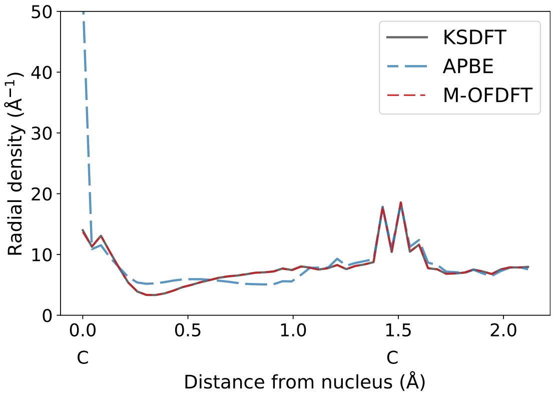
Supplementary Figure 12: Bond lengths and bond angles of ethanol structures by geometry optimization with M-OFDFT and KSDFT. Different points represent the optimizes results from different initial structures. The violin plots in the background depict the distributions of the corresponding bond lengths or angles under thermodynamic equilibrium, which are calculated on ethanol structures (n=20,000). The vertical dashed lines represent the quartiles. The mean RMSD between the optimized structure by M-OFDFT and by KSDFT over the initial structures is 0.07Å.



Supplementary Figure 13: Kinetic energy value comparison between M-OFDFT and the vW KEDF. Each point represents the vW KEDF value (x-axis coordinate) and the kinetic energy value by our learned KEDF model (y-axis coordinate) of each electron density. Two versions of our learned KEDF model are considered, including the residual KEDF version **(a,c)** and the TXC functional version **(b,d)**. The densities for evaluation come from ground-state densities optimized by our KEDF model on 10,000 unseen ethanol test structures **(a-b)** or 13,388 unseen QM9 test structures **(c-d)**.

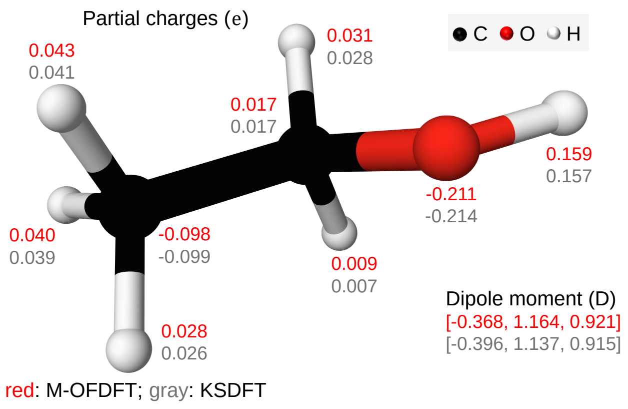


(a) α-carbon

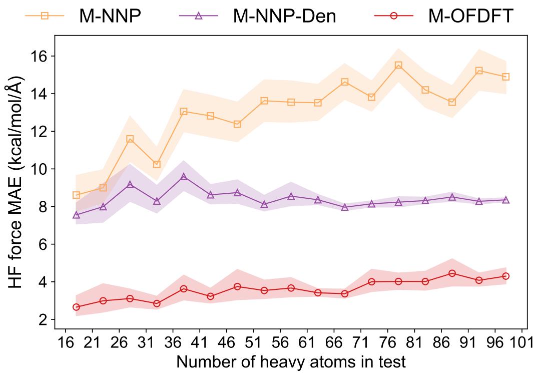


(b) β-carbon

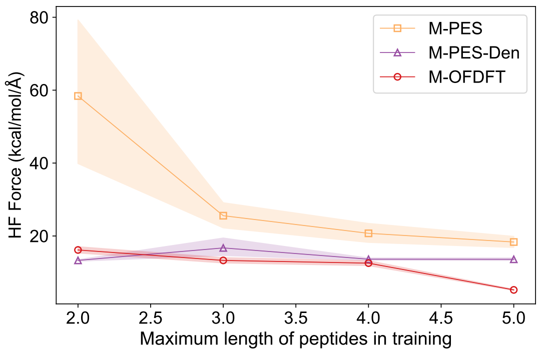
Supplementary Figure 14: Additional visualization of the density optimized by various methods. Integrated density on spheres of varying radii around each of the two carbon atoms in an ethanol structure is plotted in each figure, in parallel with Fig. 2(b).



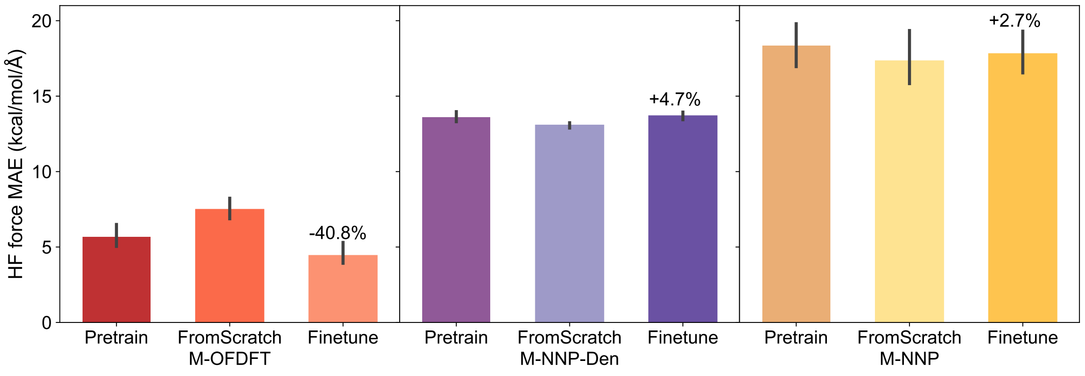
Supplementary Figure 15: Visualization of Hirshfeld atomic partial charge on each atom in an ethanol structure as well as the dipole moment of the structure. Both the atomic partial charges and the dipole moment derived from the solved electron density by M-OFDFT align closely with those solved by KSDFT.



(a) QMugs (fixed training dataset, in parallel with Fig. 3(a))

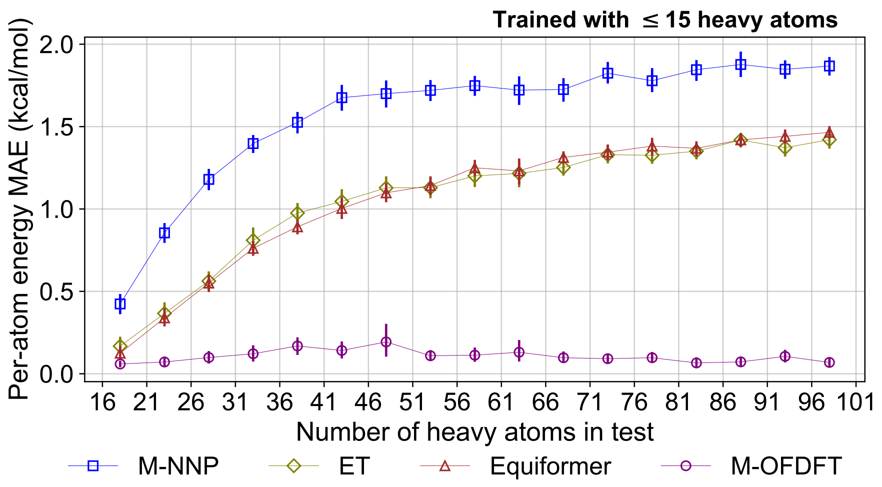


(b) Chignolin (fixed test dataset, in parallel with Fig. 3(d))

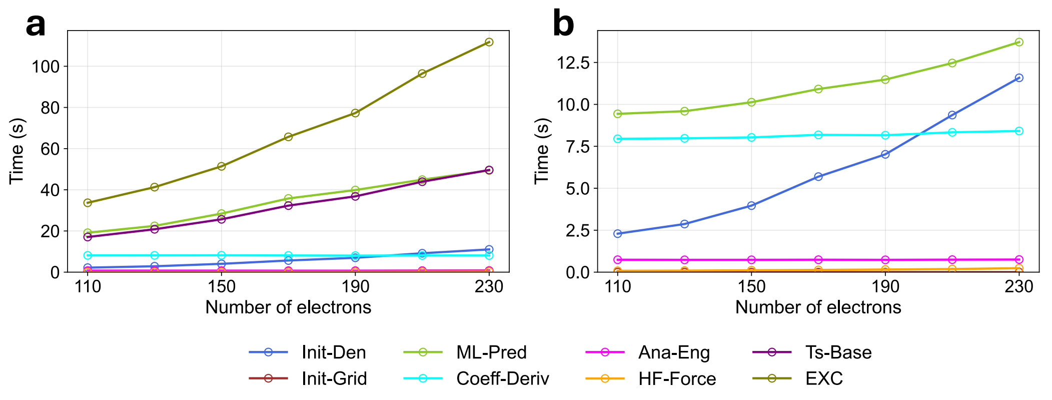


(c) Chignolin (finetuning setting, in parallel with Fig. 3(e))

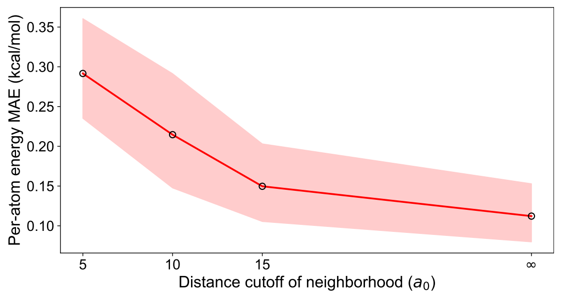
Supplementary Figure 16: Extrapolation performance in HF force of M-OFDFT compared with M-NNP and M-NNP-Den on the two extrapolation settings. The data are mean and the shades as well as error bars in all figures show 95% confidence intervals. **(a)** HF force MAE on increasingly larger molecules from the QMugs dataset, using models trained on molecules with no more than 15 heavy atoms from QM9 and QMugs datasets, in parallel with Fig. 3(a). Each value is calculated on 50 QMugs molecules. **(b)** HF force MAE on Chignolin structures (n=1,000), using models trained on a series of datasets including increasingly longer peptides, in parallel with Fig. 3(d). **(c)** HF force MAE on Chignolin structures (n=50), using models trained on all peptides without ('Pretrain') and with 'Finetune' on 500 Chignolin structures, in parallel with Fig. 3(e). Also marked are error reduction ratios by the finetuned models over models trained 'FromScratch' on the 500 Chignolin structures only.



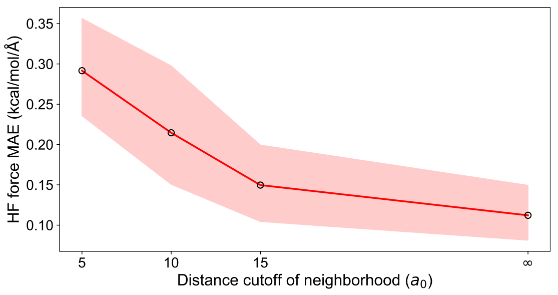
Supplementary Figure 17: Extrapolation performance of M-OFDFT compared to other advanced deep-learning architectures. Each line denotes the mean absolute error (MAE) in per-atom energy on increasingly larger molecules from the QMugs dataset, using a model trained on molecules with no more than 15 heavy atoms from QM9 and QMugs datasets. Each value is calculated on 50 QMugs molecules and the bars show 95% confidence intervals. The setting is in parallel with Fig. 3(a). Beyond M-NNP, that is, the NNP using the same model architecture as M-OFDFT and already presented in Fig. 3(a), two more architectures for NNP are investigated: Equivariant Transformer (ET) [72] and Equiformer [73].



Supplementary Figure 18: Empirical time cost of various computational components in the density optimization process of M-OFDFT, under **(a)** the residual KEDF $T_{\mathrm{S,res},\theta}$ formulation and **(b)** the TXC functional $E_{\mathrm{TXC},\theta}$ formulation. Computational components are defined in the following. "Init-Den": initialization of density (including density fitting); "Init-Grid" (only for the $T_{\mathrm{S,res},\theta}$ formulation): generation of grid points and evaluation of basis function values on them; "ML-Pred": evaluation of the deep-learning model, $T_{\mathrm{S,res},\theta}$ or $E_{\mathrm{TXC},\theta}$, including the local frame module (Supplementary Section B.2) and enhancement modules (Supplementary Section B.3); "Coeff-Deriv": automatic differentiation to compute the gradient of the deep-learning model with respect to input density coefficients; "Ana-Eng": computation of the values and gradients with respect to density coefficients of energy terms that have analytical expressions, that is, the Hartree energy E_{H} (Eq. (47)) and the external potential energy E_{ext} (Eq. (49)); "HF-Force": Hellmann-Feynman force computation (Supplementary Section A.5) conducted after density optimization; "Ts-Base" (only for the $T_{\mathrm{S,res},\theta}$ formulation): evaluation of the value and gradients with respect to density coefficients of the base KEDF on the grid; "EXC" (only for the $T_{\mathrm{S,res},\theta}$ formulation): evaluation of the value and gradients with respect to density coefficients of the XC functional on the grid.



(a) Energy results



(b) HF force results

Supplementary Figure 19: Ablation study results for the nonlocality of the functional model architecture. The shades show 95% confidence intervals. **(a)** and **(b)** show the energy error and HF force error of M-OFDFT on QMugs molecules (n=50) with 56-60 heavy atoms, for which the model is trained on QM9 and QMugs molecules of no more than 15 heavy atoms. The distance cutoff ∞ represents a fully-connected graph (nonlocal model). The experiment is conducted on the extrapolation setting in parallel with Supplementary Table 6.