Dissertation

submitted to the

Combined Faculty of Mathematics, Engineering and Natural Sciences

of Heidelberg University, Germany

for the degree of

Doctor of Natural Sciences

Put forward by

M.Sc. Roman Remme

born in: Wiesbaden

Oral examination: 21-01-2025

Machine Learning Chemically Accurate
Orbital-Free Density Functional Theory

Referees: Prof. Dr. Fred A. Hamprecht

Prof. Dr. Maurits W. Haverkort

Abstract

Orbital-free density functional theory (OF-DFT) is a cost-effective framework for electronic structure calculations. We demonstrate the feasibility of machine learning accurate and generalizable density functionals, particularly comprising the kinetic energy required for OF-DFT.

We introduce KineticNet, a deep neural network tailored to predict the kinetic energy density. Trained on varied data generated with a novel scheme based on sampling the external potential, KineticNet achieves chemical accuracy on small molecules and reproduces chemical bonding in orbital-free density optimization.

Expanding this success, we transition from grid-based density representations to the more efficient linear combination of atomic basis functions Ansatz. Adapting and improving our external potential sampling strategy, we achieve state-of-the-art results for OF-DFT on the QM9 dataset of organic molecules, in both energy and density prediction. Crucially, we address a key limitation of previous approaches by enabling convergent density optimization with chemical accuracy.

Finally, we propose surrogate functionals, enabling optimization of electron densities without directly replicating physical energy functionals. By integrating surrogate loss functions and a novel train-time density optimization scheme, we further boost the accuracy of density predictions while reducing training data requirements. This innovative approach opens new avenues for efficient and scalable energy functional development.

Zusammenfassung

Orbitalfreie Dichtefunktionaltheorie (OF-DFT) ist ein kostengünstiger Ansatz zur Berechnung elektronischer Strukturen. Wir zeigen, dass mittels maschinellen Lernens präzise und generalisierbare Dichtefunktionale entwickelt werden können, welche insbesondere die für OF-DFT benötigte kinetische Energie enthalten.

Wir stellen KineticNet vor, ein tiefes neuronales Netzwerk zur Vorhersage der kinetischen Energiedichte. Trainiert mit neuartigen variierten Daten welche mithilfe von Störungen des externen Potentials generiert wurden, erreicht KineticNet chemische Genauigkeit auf kleinen Molekülen und reproduziert chemische Bindung in orbitalfreier Dichteoptimierung.

Darüber hinaus wechseln wir von gitterbasierten Dichte-Repräsentationen hin zur effizienteren Linearkombination atomarer Basisfunktionen. Mit einer für diesen Ansatz angepassten und verbesserten Strategie zur Störung des externen Potentials erzielen wir erstklassige Ergebnisse für OF-DFT auf dem QM9-Datensatz organischer Moleküle für Energie- und Dichtevorhersagen. Ein entscheidender Fortschritt ist die stabil konvergente Dichteoptimierung mit chemischer Genauigkeit.

Abschließend führen wir Surrogat-Funktionale ein, welche die Optimierung von Elektronendichten ohne direkte Nachbildung physikalischer Energiefunktionale erlauben. Durch die Integration von Surrogat-Kostenfunktionen und einem neuartigen Optimierungsschema für Dichten während des Trainings steigern wir die Genauigkeit der Dichtevorhersagen weiter und verringern den Datenbedarf für das Training. Dieser innovative Ansatz eröffnet neue Möglichkeiten für die effiziente Entwicklung von skalierbaren Energie-Funktionalen.

Acknowledgements

First, I want to thank my advisor Fred Hamprecht, who first sparked my interest in machine learning through one of his lectures. He gave me the opportunity to join his group, explore my interests and develop my scientific career. Thank you for creating a friendly atmosphere and mentoring me throughout the years, always encouraging me when I doubted myself.

I would like to acknowledge Andreas Dreuw for supporting this work with his expert knowledge on the chemistry side. Furthermore, I would like to thank Maurits Haverkort for his interest in my project and kindly agreeing to serve as second referee for this thesis.

I was lucky to work amongst many inspiring collaborators and want to express my gratitude towards all of them. This thesis would not have been possible without Tobias Kaczun. His chemical expertise was a great help throughout the years and repeatedly proved invaluable. Furthermore, I want to thank the other brilliant and hard-working members of the OF-DFT team, Christof Gehrig, Manuel Klockow, Marc Ickler, Tim Ebert, Dominik Geng, Johannes Schmidt, Mats Kothe, Gerrit Gerhartz and Heinrich von Campe, for making the last year of my PhD particularly productive, and especially fun.

Former lab manager Barbara Werner supported me since my masters and had everyones back with her deep knowledge of the university. Her kindness left a lasting impression on me. Thanks also to Barbara Quintel for lifting many organizational weights of my shoulders now.

I would like to thank all members of the Hamprecht Lap I had the pleasure to work with throughout the years, in particular Lorenzo Cerrone, Enrique Fita Sanmartin, Sebastian Damrich, Ocima Kamboj, Alberto Bailoni and Steffen Wolf for countless fruitful scientific discussions and for making my time in the lab so much more enjoyable. Peter Lippmann, my office roommate during the better part of my PhD, who became a dear friend, I want to thank not only for listening to many of my pitches of good and bad ideas, and helping me to tell them apart from one another, but also for always lifting my spirits and motivating me.

I am immensely grateful to my mother Gabriele Remme and father Thomas Zours, for always believing in me and for their continuous support and encouragement throughout my life.

And finally I want to thank Julia Ivanova, the love of my life, for carefully proof-reading large parts of this thesis, but more importantly for always keeping me afloat, and for bettering my life in so many ways.

Contents

Abstract **v**

Zusammenfassung **vii**

Acknowledgements **ix**

Contents **xi**

List of Figures **xv**

List of Tables **xvii**

1 Introduction **1**

2 Density Functional Theory **5**

2.1 The Hohenberg-Kohn theorems and the energy functional 6

2.2 Levy-Lieb constrained search 8

2.3 Kohn-Sham DFT 9

2.3.1 Formulation under an atomic basis 11

2.3.2 Direct inversion of the iterative subspace 13

2.3.3 Scaling considerations 14

2.4 Orbital-free DFT 14

2.4.1 The kinetic energy functional 15

2.4.2 Density representations for OF-DFT 16

2.4.3 Density optimization 19

3 Geometric deep learning for atomistic systems **21**

3.1 Permutation invariance for graphs 21

3.2 Equivariance 22

3.2.1 Tensor fields 22

3.2.2 Canonicalization via local frames 23

3.3 Architectural desiderata 24

4 Learning a transferable kinetic energy functional on quadrature grids 27
4.1 Introduction 28
4.2 KineticNet: a deep equivariant architecture 30
4.3 Training data generation 33
4.4 Density optimization 35
4.5 Computational experiments 36
4.5.1 Training details 36
4.5.2 Test results 36
4.5.3 Density optimization 40
4.6 Conclusion 42

5 OF-DFT using LCAB 43
5.1 Generating labels with KS-DFT 45
5.1.1 Density fitting 46
5.2 Basis transformations 47
5.2.1 Natural reparametrization 47
5.3 Gradient projection 48
5.3.1 Interplay of basis transformation and projection 50
5.4 SAD guess 52
5.5 External potential perturbation 55
5.5.1 Details of external potential sampling 56
5.6 Architectural improvements 60
5.6.1 Graphormer 60
5.6.2 Equiformer 61
5.7 Implementation details 63
5.8 Results 64
5.8.1 SAD guess 64
5.8.2 Tuning loss weights 64
5.8.3 Perturbed training data 66
5.8.4 Training target 69
5.8.5 Training without natural reparametrization 69
5.8.6 Convergence and comparison to M-OFDFT 70
5.9 Discussion 73

6 Surrogate models to physical functionals 77
6.1 Motivation 77

6.2 Definitions: strong and weak . . . 79
6.3 Loss functions . . . 80
6.3.1 Lower bound loss . . . 81
6.3.2 Gradient to ground state loss . . . 81
6.3.3 Gradient norm range loss . . . 83
6.3.4 Gradient descent improvement loss . . . 83
6.4 Surrogate training data . . . 84
6.4.1 Sampling densities around the ground state . . . 85
6.4.2 Train-time density optimization via caching . . . 85
6.5 Surrogate architectures . . . 90
6.5.1 Dimension-wise rescaling . . . 90
6.5.2 Atomic reference module . . . 90
6.6 Results . . . 91
6.6.1 Data-driven Hyperparameter Choices . . . 91
6.6.2 Training on MD17 . . . 92
6.6.3 Training on QM9 . . . 93
6.7 Discussion . . . 95

7 Contributions and outlook 97

8 Appendix 101
A Learning a transferable kinetic energy functional on quadrature grids . 101
A.1 Data generation . . . 101
A.2 OF-DFT implementation . . . 102
A.3 Correspondence between KS and OF Ansatz for two electrons . 102
A.4 Atomic contributions . . . 102
A.5 Model hyperparameters . . . 103
B Equiformer hyperparameters . . . 106

List of Figures

1.1 Orbital-Free Density Functional Theory Energy Surface 2

2.1 Cusp of the electron density at a nucleus 5
2.2 Schematic of the SCF procedure 11

3.1 Local frames for an ethanol molecule 24

4.1 KineticNet Architecture 30
4.2 Schematic of adjusted radial basis 32
4.3 HF Dissociation Curve 37
4.4 Ne Dissociation Curve 37
4.5 Comparison of energies on H_3^+ 38
4.6 Slices through input, prediction, and error for H_3^+, on three test samples 39
4.7 H2 Dissociation curve via Density Optimization 40

5.1 Variational prediction in the LCAB Ansatz 44
5.2 The need for gradient projection 49
5.3 Preserving electron number in density optimization 51
5.4 Orthogonal projection and basis transformation 52
5.5 Different methods for normalizing the initial guess. 54
5.6 Slices of densities 57
5.7 Perturbed vs. unperturbed energy difference histogram 57
5.8 Distances of perturbed and unperturbed samples to ground state densities 58
5.9 Gradients of perturbed data point are more aligned with direction to ground state 59
5.10 Adapted embedding module for the EquifomerV2 model 62
5.11 Loss weights tuning 65
5.12 Density optimization on 500 QM9 Molecules 68
5.13 Radar plot: Ours vs. M-OFDFT 71
5.14 Gradient norm during density optimization, comparing M-OFDFT to ours 72

5.15 Total energy error during density optimization comparing M-OFDFT to ours 72

6.1 Surrogate Energy Landscape 78
6.2 2D Slice of energy and gradient-norm surface trained with gradient-to-ground-state loss alone 82
6.3 Train-time density optimization via caching 87
6.4 Ideal density optimization step 88
6.5 Data-driven choice of surrogate loss hyperparameters 91
6.6 Density optimization with a surrogate model on QM9 95

A.1 Distribution of the kinetic energy 104
A.2 Distribution of steps until convergence for the three two-electron systems and different density optimization modes 104
A.3 Effectiveness of subtracting atomic contributions 105

List of Tables

4.1 Kinetic energy mean absolute error for our model (KineticNet) and classical functionals 38
4.2 Density optimization results for two electron systems using our ML functional (KineticNet) as well as classical functionals 41

5.1 Density errors of initial guesses 64
5.2 Training data ablations 66
5.3 Graphormer ground state errors 70
5.4 Equiformer ground state errors 70
5.5 Comparison of OF-DFT methods on QM9 density optimization 73

6.1 Density optimization with surrogate functionals trained on MD17 . . . 92
6.2 Density optimization with surrogate functionals trained on QM9 94

A.1 Data sets used for training KineticNet 102
B.2 EquiformerV2 Hyperparameter choices 106

Chapter 1

Introduction

More than 50 years ago, Hohenberg and Kohn proposed a groundbreaking insight: The electron density $\rho(\mathbf{r})$ is sufficient to describe a quantum chemical system in its ground state, and there exists an energy functional E, mapping electron densities ρ to their corresponding energies $E[\rho]$. These energies can be minimized to find the ground state electron density ρ^*, see figure 1.1. Hence, Density Functional Theory (DFT) was born [1].

The energy functional E can be decomposed into a sum of several terms: The kinetic energy $T[\rho]$, the electron-electron interaction energy $V_{ee}[\rho]$, and the external potential energy $V_{ext}[\rho]$. While an analytic expression exists for the external potential energy, and generations of chemists have been developing ever more accurate approximations for the electron-electron interaction energy, computing the kinetic energy at chemical accuracy from the electron density alone remains a challenge [2].

This is why shortly after the original Hohenberg-Kohn theorems, Kohn and Sham proposed a practical solution: Instead of computing the kinetic energy directly from the electron density, they side-stepped the problem by introducing a set of auxiliary, non-physical orbitals ϕ_i which are used to compute the total kinetic energy as a sum of the kinetic energies of the individual orbitals: $T[\rho] = \sum_i \frac{1}{2} \int |\nabla \phi_i|^2 d^3\mathbf{r}$. This Kohn-Sham Density Functional Theory (KS-DFT) [3] has since become the workhorse of computational quantum chemistry and materials science. KS-DFT has been used to predict the properties of a wide range of systems, from small molecules to proteins, and from semiconductors to metals [4]. However, taking the detour of reintroducing orbitals comes at a cost: The computational scaling of KS-DFT is cubic with the number of atoms in the system, which limits its applicability to systems with a few hundred atoms at most.

Meanwhile, the original version of DFT, now known as Orbital-Free DFT (OF-

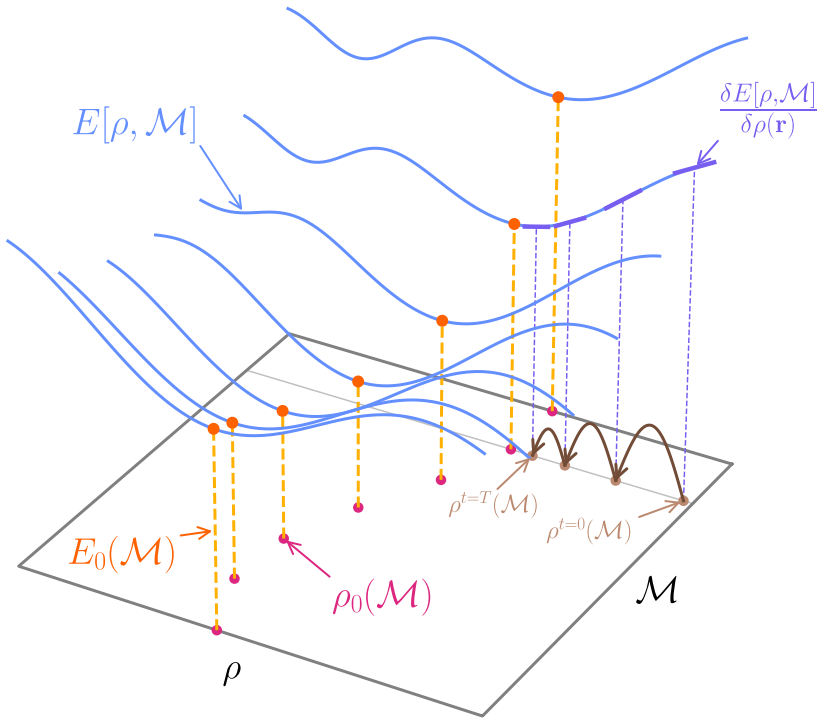


Figure 1.1. Orbital-Free Density Functional Theory. The functional $E[\rho, \mathcal{M}]$ (blue) maps electron densities ρ and molecules $\mathcal{M}$ (in general: external potentials) to their corresponding energies. Utilizing its functional derivative (purple), this energy functional can be minimized (brown), yielding ground state energies $\rho_0(\mathcal{M})$ and densities $E_0(\mathcal{M})$. In chapters 4 and 5, we will present our efforts to approximate $E[\rho, \mathcal{M}]$, while we present an alternate approach in chapter 6, compare with figure 6.1.

DFT)[1], holds the promise of linear scaling with system size. However, even though Hohenberg and Kohn already knew that a kinetic energy functional of the electron density alone exists, OF-DFT has been held back. Decades of research have not yet yielded a practical recipe to compute the kinetic energy from only the electron density, at chemical accuracy and across a wide range of chemical systems.

Today however, we are at the brink of a new era in quantum chemistry: Geometric deep learning [5] has enabled us to design neural networks that can operate on molecular graphs, exhibit the non-local nature of the kinetic energy functional, and are equivariant to rotations and translations, thereby respecting the symmetries of the underlying physical system. Equipped with these tools, we can now attempt to learn the energy functional from data, and are closer than ever to bridging the gap between the accuracy of KS-DFT and the improved scaling of OF-DFT.

In this thesis, we will present our efforts to develop machine-learned energy functionals for OF-DFT.

A recurring theme will be the challenge of density optimization: In order for the functional to be useful, it must not only generalize across the training distribution (as can be tested by validation accuracy), but must also have a minimum at the ground state of a system, i.e. it must not assign a lower energy to a non-ground state density than to the ground state density. Achieving this form of robustness in the high dimensional space of electron densities is a key challenge in the development of machine-learned energy functionals [6, 7, 8, 9].

In chapters 2 and 3, we will introduce the theoretical background of the two disciplines coming together in this thesis: Density functional theory and geometric deep learning, respectively. Chapter 4 will introduce KineticNet, the first deep neural network architecture which predicts the kinetic energy with chemical accuracy across a number of small molecules, generalizing over input densities and geometries, and reproducing a chemical bonding in orbital free density optimization, thereby serving as a proof of principle for machine-learned OF-DFT. In chapter 5 we describe the seminal work by Zhang, Liu, You, Liu, Zheng, Lu, Wang, Zheng, and Shao, M-OFDFT [9], and build upon it to alleviate some of its key shortcomings, in particular its lack of proper convergence in density optimization.

Chapter 6 will introduce the concept of *surrogate functionals*, whose goal is to replace the exact, physical, energy functional in density optimization without attempting to perfectly mimic it. Finally, in chapter 7, we will summarize the key findings of this thesis and give an outlook on future work.

Over the course of this thesis, I co-advised a number of master and bachelor stu-

[1]Sometimes also pure DFT

dents, who performed numerous experiments and contributed to the overall progress of the project. Specifically, Tim Ebert implemented and profiled density fitting (section 5.1.1), Manuel Klockow worked on the enhanced data generation of section 5.5, Dominik Geng on the natural reparametrization described in section 5.2.1, and finally, Mats Kothe has conducted many of the experiments presented in chapter 6..

Chapter 2

Density Functional Theory

Density functional theory (DFT) is a quantum mechanical method to describe the electronic structure of many-body systems. It is based on the insight that the ground state electron density $\rho(\mathbf{r})$ is sufficient to describe a quantum chemical system, and that an energy functional E, mapping electron densities ρ to their corresponding energies $E[\rho]$, exists, which can be minimized to find the ground state electron density ρ^*.

In this chapter, I will review the theoretical background of DFT, focusing on the Hohenberg-Kohn theorems (section 2.1), the Kohn-Sham DFT Ansatz (section 2.3), and the orbital-free DFT approach, in particular focussing on the variuos density representations used in this work (section 2.4). Please note that this chapter does not aim to provide a comprehensive introduction to DFT, but rather to sketch the basics and introduce the required notation for this thesis. For the former, we refer the interested reader to the many excellent textbooks on the subject, such as [10].

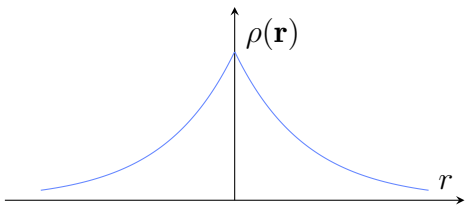


Figure 2.1. Cusp of the electron density at a nucleus. The electron density $\rho(\mathbf{r})$ has a cusp at each nucleus, whose shape uniquely determines the nuclear charge.

2.1 The Hohenberg-Kohn theorems and the energy functional

A central goal of computational quantum chemistry is solving for the ground state of the electronic Schrödinger equation for a given system with Hamiltonian

$$\hat{H} = \hat{T} + \hat{V}_{ee} + \hat{V}_{\text{ext}} \,, \tag{2.1}$$

where $\hat{T}$ is the kinetic energy operator, $\hat{V}_{ee}$ is the electron-electron interaction operator, and $\hat{V}_{\text{ext}}$ is the external potential operator, respectively given by

$$\hat{T} = -\frac{1}{2}\sum_{i=1}^{N} \nabla_i^2 \,, \tag{2.2}$$

$$\hat{V}_{ee} = \frac{1}{2}\sum_{1\leq i<j\leq N} \frac{1}{|\mathbf{r}_i - \mathbf{r}_j|} \,, \tag{2.3}$$

$$\hat{V}_{\text{ext}} = \sum_{i=1}^{N_\text{e}} V_{\text{ext}}(\mathbf{r}_i) \overset{(*)}{=} \sum_{i=1}^{N_\text{e}}\sum_{a=1}^{A} \frac{Z_a}{|\mathbf{R}_a - \mathbf{r}_i|} \,, \tag{2.4}$$

with the local external potential $V_{\text{ext}}(\mathbf{r})$, and the final equality $(*)$ in 2.4 holding in the case of a molecular system $\mathcal{M}$ with N electrons and A nuclei at positions $\{\mathbf{R}_a\}_{a=1,\dots,A}$ and charges $\{Z_a\}_{a=1,\dots,A}$. While all the operators have exact closed-form expressions, actually solving for the ground state wave function ψ_0 quickly becomes computationally infeasible for systems of more than a few electrons, because the dimensionality of the wave function scales exponentially with the number of electrons.

The two Hohenberg-Kohn theorems [1] lay the foundation for a way out of this conundrum: Instead of the wave function, they show that the one-particle electron density

$$\rho(\mathbf{r}) = N \int |\psi\left(\mathbf{r}, \mathbf{r}_2, \dots, \mathbf{r}_N\right)|^2 \, \mathrm{d}\mathbf{r}_2 \dots \mathrm{d}\mathbf{r}_N \,, \tag{2.5}$$

is sufficient to describe a quantum chemical system:

The first theorem states that for each ground state electron density $\rho_0(\mathbf{r})$, an external potential $V_{\text{ext}}(\mathbf{r})$ exists which is unique up to an additive constant. For molecular systems, this can be seen intuitively. The ground state electron density has cusps at the nuclei, and the shape of these cusps can be used to determine the nuclear charges via Kato's theorem [11] (illustrated in figure 2.1). Together with their locations, this uniquely determines the external potential.

As the external potential determines the Hamiltonian $\hat{H}$ of the system, which in turn determines the ground state wave function ψ_0, from which all ground state properties such as the total energy can be computed, this theorem implies that the

electron density uniquely determines the ground state energy E_0 by following the chain

$$\rho_0 \to V_{\text{ext}} \to \hat{H} \to \psi_0 \to E_0 \,. \tag{2.6}$$

The second theorem states that the ground state energy $E[\rho]$ is a unique functional of the electron density $\rho(\mathbf{r})$, i.e. there exists a functional $E[\rho]$ such that the ground state density is its minimizer, being mapped by the functional to the ground state energy.

This motivates the search for the exact energy functional $E[\rho]$ which, when minimized, yields the ground state electron density (The chain of implications 2.6 is of no use for this, as the third step is solving the Schrödinger equation).

The energy functional E can be decomposed into several terms: The kinetic energy $T[\rho]$, the electron-electron interaction energy $V_{ee}[\rho]$, and the external potential energy $V_{\text{ext}}[\rho]$:

$$E[\rho] = T[\rho] + E_{ee}[\rho] + E_{\text{ext}}[\rho] \,. \tag{2.7}$$

The kinetic energy is oftentimes replaced with its non-interacting counterpart T_{S}, and the electron-electron interaction approximated by the Hartree energy E_H. To keep the expression exact, the exchange-correlation functional E_{xc} is defined to absorb the differences, leading to the familiar decomposition of the total energy functional:

$$E[\rho] = T_{\text{S}}[\rho] + E_H[\rho] + E_{\text{xc}}[\rho] + E_{\text{ext}}[\rho] \,. \tag{2.8}$$

For the external potential and the Hartree energy, there exist analytic expressions. For a system in the Born-Oppenheimer approximation, under which we operate throughout this thesis, and nuclei at positions $\{\mathbf{R}_a\}_{a=1,\dots,A}$ and charges $\{Z_a\}_{a=1,\dots,A}$, they are given by:

$$E_{\text{ext}}[\rho] = \int \rho(\mathbf{r}) V_{\text{ext}}(\mathbf{r}) d\mathbf{r} = -\int \rho(\mathbf{r}) \sum_{a=1}^{A} \frac{Z_a}{|\mathbf{r} - \mathbf{R}_a|} \,, \tag{2.9}$$

$$E_H[\rho] = \frac{1}{2} \int\int \frac{\rho(\mathbf{r})\rho(\mathbf{r}')}{|\mathbf{r} - \mathbf{r}'|} d\mathbf{r} d\mathbf{r}' \,. \tag{2.10}$$

The exchange-correlation energy $E_{xc}[\rho]$ makes up a relatively small part of the total energy, but is still crucial for the accuracy of DFT calculations. A multitude of approximations for this term exist, ranging from simple local density approximations [12] via generalized gradient approximations [13] to highly sophisticated hybrid [14], and even double hybrid functionals [15].

2.2 Levy-Lieb constrained search

The original work of Hohenberg and Kohn has some shortcomings, for instance the fact that in their formulation the energy functional is only defined for so-called ν-representable densities, i.e. densities which can be obtained as the ground state density of some external potential V_{ext} (sometimes called ν).

Levy-Lieb constrained-search [16, 17] provides a formulation of DFT which allows the energy functional to be defined for all N-representable densities, i.e. densities resulting (via eq. 2.5) from some antisymmetric wave function representing N electrons. Let us follow their construction:

$$E^* = \min_{\psi:\text{antisym},\langle\psi|\psi\rangle=1} \langle\psi|\,\hat{T} + \hat{V}_{\text{ee}} + \hat{V}_{\text{ext}}\,|\psi\rangle\,. \tag{2.11}$$

Using the definition of the external potential operator (eq. 2.4) as well as that of the electron density (eq. 2.5), we can readily express the external energy as

$$\langle\psi|\,\hat{V}_{\text{ext}}\,|\psi\rangle = \int \rho(\mathbf{r})V_{\text{ext}}(\mathbf{r})\,\mathrm{d}\mathbf{r} =: E_{\text{ext}}[\rho], \tag{2.12}$$

thereby demonstrating equation 2.9 and allowing us to rewrite the minimization as a two-level problem:

$$E^* = \min_{\psi:\text{antisym},\langle\psi|\psi\rangle=1} \langle\psi|\,\hat{T} + \hat{V}_{\text{ee}}\,|\psi\rangle + E_{\text{ext}}[\rho] \tag{2.13}$$

$$= \min_{\rho:\rho\geq 0,\int d\mathbf{r}\rho(\mathbf{r})=N} \left(\min_{\psi:\text{antisym},\rho_{[\psi]}=\rho} \langle\psi|\,\hat{T} + \hat{V}_{\text{ee}}\,|\psi\rangle \right) + E_{\text{ext}}[\rho] \tag{2.14}$$

$$= \min_{\rho:\rho\geq 0,\int d\mathbf{r}\rho(\mathbf{r})=N} \left(U[\rho] + E_{\text{ext}}[\rho]\right)\,. \tag{2.15}$$

Here, $\rho_{[\psi]}$ denotes the electron density resulting from the wave function ψ and where we have defined the *universal functional* $U[\rho]$ as the result of the inner minimization[1]:

$$U[\rho] = \min_{\psi:\text{antisym},\rho_{[\psi]}=\rho} \langle\psi|\,\hat{T} + \hat{V}_{\text{ee}}\,|\psi\rangle\,. \tag{2.16}$$

The contribution to $U[\rho]$ from the electron-electron interaction operator $\hat{V}_{\text{ee}}$ is approximated by the Hartree energy $E_H[\rho]$ (eq. 2.10), and the kinetic energy operator $\hat{T}$ is approximated by the *non-interacting kinetic energy functional* $T_{\text{S}}[\rho]$:

$$T_{\text{S}}[\rho] \coloneqq \min_{\psi:\text{antisym},\rho_{[\psi]}=\rho} \langle\psi|\,\hat{T}\,|\psi\rangle\,. \tag{2.17}$$

To keep the theory exact, the errors of both approximations are absorbed into the exchange-correlation energy $E_{xc}[\rho]$:

$$E_{\text{xc}}[\rho] = U[\rho] - T_{\text{S}}[\rho] - E_{\text{H}}[\rho]. \tag{2.18}$$

[1] Detailed analysis of the optimization spaces in this two-level optimization and its equivalence to the original constrained search 2.11 was performed by Lieb in [17].

Adding the external energy back in, we once more arrive at the familiar decomposition of the energy functional (eq. 2.8) into kinetic, Hartree, exchange-correlation (xc), and external energy, now defined for all N-representable densities:

$$E[\rho] = \underbrace{T_s[\rho]}_{\substack{\text{kinetic,}\\ \text{unknown}}} + \underbrace{E_H[\rho]}_{\substack{\text{Hartree,}\\ \text{known}}} + \underbrace{E_{\text{xc}}[\rho]}_{\substack{\text{xc,}\\ \text{approx.}\\ \text{known}}} + \underbrace{E_{\text{ext}}[\rho]}_{\substack{\text{external,}\\ \text{known}}} . \tag{2.19}$$

The glaring problem is the non-ineracting kinetic energy functional $T_s[\rho]$, for which no exact, or even satisfactory approximate, expression for molecular systems is known.

The first widely applicable method addressing this problem was introduced by Kohn and Sham and is the subject of the next section.

2.3 Kohn-Sham DFT

The Kohn-Sham DFT Ansatz [3] side-steps the problem of computing the kinetic energy from the electron density alone by reintroducing an auxiliary, non-physical wave function describing N_e non-interacting electrons, from which $T_s[\rho]$ can be computed as the sum of the kinetic energies of the individual orbitals. This wave function is expressed by a single Slater determinant of N_e orbitals $\{\phi_i\}_{i=1,\dots,N_\text{e}}$:

$$\psi(\mathbf{r}_1, \dots, \mathbf{r}_{N_\text{e}}) \coloneqq \frac{1}{\sqrt{N_\text{e}!}} \det\left(\phi_i(\mathbf{r}_j)\right)_{ij} , \quad \rho(\mathbf{r}) = \sum_{i=1}^{N_\text{e}} |\phi_i(\mathbf{r})|^2 , \tag{2.20}$$

from which the non-interacting kinetic energy can be computed as

$$T_\text{S}[\rho] = \min_{\mathbf{\Phi}:\rho_{[\mathbf{\Phi}]}=\rho} \sum_{i=1}^{N_\text{e}} \langle\phi_i| \hat{T} |\phi_i\rangle . \tag{2.21}$$

Using this expression for the non-interacting kinetic energy allows us rewrite the expression for the energy as

$$E^* = \min_{\rho:\rho\geq 0, \int d\mathbf{r}\rho(\mathbf{r})=N} \left(\min_{\mathbf{\Phi}\text{: orthonormal, } \rho_{[\mathbf{\Phi}]}=\rho} \left[\sum_{i=1}^{N} \langle\phi_i| \hat{T} |\phi_i\rangle \right] + E_\text{H}[\rho] + E_\text{xc}[\rho] + E_\text{ext}[\rho] \right) \tag{2.22}$$

$$= \min_{\mathbf{\Phi}\text{: orthonormal}} \left[\sum_{i=1}^{N} \langle\phi_i| \hat{T} |\phi_i\rangle + \underbrace{E_\text{H}[\rho_{[\mathbf{\Phi}]}] + E_\text{xc}[\rho_{[\mathbf{\Phi}]}] + E_\text{ext}[\rho_{[\mathbf{\Phi}]}]}_{=:E_\text{eff}[\rho]} \right] \tag{2.23}$$

$$= \min_{\mathbf{\Phi}\text{: orthonormal}} \left[\sum_{i=1}^{N} \langle\phi_i| \hat{T} |\phi_i\rangle + E_\text{eff}[\rho] \right] , \tag{2.24}$$

where we went back to a formulation as a constrained single-level optimization problem, now over the *Kohn-Sham orbitals* $\mathbf{\Phi} = \{\phi_i\}_{i=1,\dots,N_\text{e}}$, which are orthonormal and

minimize the sum of the kinetic energy (computed directly from the orbitals) and the effective potential $E_{\text{eff}}[\rho]$ (computed via the electron density generated by these orbitals). Solving this minimization lies at the core of Kohn-Sham DFT.

To solve equation 2.24, we consider the variation of the energy functional with respect to the orbitals ϕ_i:

$$\frac{\delta E[\mathbf{\Phi}]}{\delta \phi_i} = \frac{\delta \left\langle \phi_i \right| \hat{T} \left| \phi_i \right\rangle}{\delta \phi_i} + \int \frac{\delta E_{\text{eff}}}{\delta \rho_{[\mathbf{\Phi}]}(\mathbf{r}')} \frac{\delta \rho_{[\mathbf{\Phi}]}(\mathbf{r}')}{\delta \phi_i(\mathbf{r})} \, \mathrm{d}\mathbf{r}. \tag{2.25}$$

$$= 2\hat{T}\phi_i + 2V_{\text{eff}[\rho_{[\mathbf{\Phi}]}]}\phi_i \tag{2.26}$$

$$= \hat{F}_{[\rho_{[\mathbf{\Phi}]}]}\phi_i \,, \tag{2.27}$$

where we have defined the Fock operator $\hat{F}_{[\rho_{[\mathbf{\Phi}]}]} = \hat{T} + V_{\text{eff}[\rho_{[\mathbf{\Phi}]}]}$ as the sum of the kinetic energy operator and the effective potential:

$$V_{\text{eff}[\rho]}(\mathbf{r}) \coloneqq \frac{\delta E_{\text{eff}}[\rho]}{\delta \rho(\mathbf{r})} = V_{\text{H}[\rho]}(\mathbf{r}) + V_{\text{xc}[\rho]}(\mathbf{r}) + V_{\text{ext}}(\mathbf{r}). \tag{2.28}$$

$$V_{\text{H}[\rho]}(\mathbf{r}) \coloneqq \frac{\delta E_{\text{H}}[\rho]}{\delta \rho(\mathbf{r})} = \int \frac{\rho(\mathbf{r}')}{\|\mathbf{r} - \mathbf{r}'\|} \, \mathrm{d}\mathbf{r}' \tag{2.29}$$

$$V_{\text{xc}[\rho]}(\mathbf{r}) \coloneqq \frac{\delta E_{\text{xc}}[\rho]}{\delta \rho(\mathbf{r})} \,. \tag{2.30}$$

Introducing Lagrange-multipliers $\boldsymbol{\epsilon} = \{\epsilon_i\}_{i=1,\dots,N}$ for the normalization part of the orthonormality constraint yields a Lagrangian

$$\mathcal{L}[\mathbf{\Phi}, \boldsymbol{\epsilon}] = E[\mathbf{\Phi}] - \sum_{i=1}^{N_{\text{e}}} \epsilon_i \left(\langle \phi_i | \phi_i \rangle - 1 \right) \,, \tag{2.31}$$

and substituting equation 2.27 yields the Kohn-Sham equations:

$$\hat{F}_{[\rho_{[\mathbf{\Phi}]}]}\phi_i = \left(\hat{T} + V_{\text{eff}[\rho_{[\mathbf{\Phi}]}]} \right) \phi_i = \epsilon_i \phi_i \,. \tag{2.32}$$

Since the Fock operator is Hermitian, solving these equations automatically yields orthogonal orbitals, such that the orthogonality part of the orthonormality constraint does not need to be enforced explicitly. The fact that the Fock operator depends on the orbitals themselves makes these equations non-linear, and renders solving them more challenging than a simple diagonalization. The typical approach is to solve them iteratively in a self-consistent field (SCF) procedure, using the orbitals from the previous iteration n to construct the Fock operator for the next iteration $n+1$ (see also the diagram in figure 2.2):

$$\hat{F}_{[\rho_{[\mathbf{\Phi}^{(n)}]}]}\phi_i^{(n+1)} = \epsilon_i^{(n+1)}\phi_i^{(n+1)} \tag{2.33}$$

Convergence is reached when the change in the orbitals between two iterations falls below a certain threshold, i.e. they are consistent with the Fock operator generated by

them. For the first SCF step, an initial guess for the density is required, for which we utilize the MINAO method [18, 19] throughout this work. In practice, the convergence of the SCF procedure is far from guaranteed, and many tricks and tweaks have been developed to improve it, such as the DIIS method [20], see section 2.3.2.

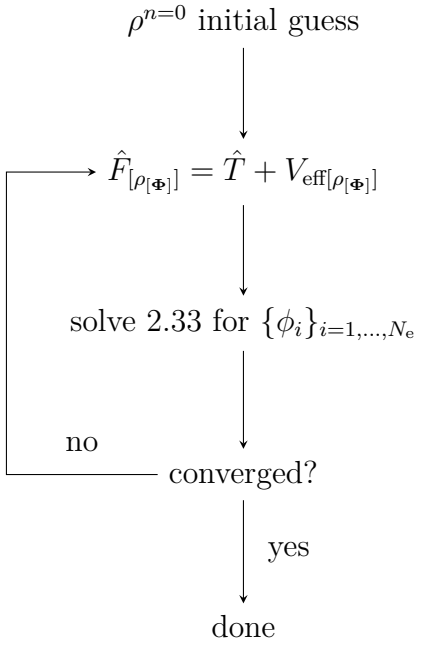


Figure 2.2. Schematic of the SCF procedure. The SCF procedure iterates over the Kohn-Sham equations, updating the orbitals ϕ_i until convergence is reached. In the inital step, the effective potential V_{eff} is constructed from the initial guess for the density $\rho^{n=0}$.

2.3.1 Formulation under an atomic basis

In order to solve the Kohn-Sham equations 2.32 computationally, the orbitals must be represented numerically in a form suitable for algorithmic optimization. To this end, the ϕ_i are expanded in a basis set of atom-centered basis functions[2] $\{\eta_\alpha(\mathbf{r})\}_{\alpha=1}^{B}$, leading to the expansion

$$\phi_i(\mathbf{r}) = \sum_{\alpha=1}^{B} C_{\alpha i}\eta_\alpha(\mathbf{r})\,. \tag{2.34}$$

[2]Sometimes also called "atomic orbitals" because of the similartiy of their angular part to physical orbitals of atoms. In this work, we use atom-centered basis function to avoid confusion with Kohn-Sham or the physical molecular orbitals.

This electron density is then expressed as

$$\rho(\mathbf{r}) = \sum_{i=1}^{N} |\phi_i(\mathbf{r})|^2 = \sum_{\alpha,\beta} \underbrace{\sum_{i=1}^{N_\mathrm{e}} C_{\alpha i} C_{\beta i}}_{=:\Gamma_{\alpha\beta}} \eta_\alpha(\mathbf{r})\eta_\beta(\mathbf{r}) = \sum_{\alpha,\beta} \Gamma_{\alpha\beta}\eta_\alpha(\mathbf{r})\eta_\beta(\mathbf{r})\,, \tag{2.35}$$

where we have defined the one-particle density matrix $\Gamma_{\alpha\beta}$. The number of parameters in this representation scales as $\mathcal{O}(NB)$, hence quadratic with system size.

Inserting the basis expansion of the Kohn-Sham orbitals into the Kohn-Sham equations 2.32 yields

$$\sum_\beta \hat{F}^{(n)} C_{\beta i}^{(n)} \eta_\beta(\mathbf{r}) = \epsilon_i^{(n)} \sum_\beta C_{\beta i}^{(n)} \eta_\beta(\mathbf{r}), \tag{2.36}$$

where we use the shorthand $\hat{F}^{(n)}$ for the Fock operator $\hat{F}_{[\rho_{[\mathbf{\Phi}^{(n)}]}]}$ of the n-th SCF iteration. We multiply with $\eta_\alpha(\mathbf{r})$ and integrate over $\mathbf{r}$ to project onto the basis functions and obtain

$$\sum_\beta F_{\alpha\beta}^{(n)} C_{\beta i}^{(n)} = \sum_\beta S_{\alpha\beta} \epsilon_i^{(n)} C_{\beta i}^{(n)}, \tag{2.37}$$

where we introduced

$$F_{\alpha\beta}^{(n)} \coloneqq \langle \eta_\alpha | \, \hat{F}^{(n)} \, | \eta_\beta \rangle \tag{2.38}$$

$$S_{\alpha\beta} \coloneqq \langle \eta_\alpha | \eta_\beta \rangle \;, \tag{2.39}$$

allowing us to write the Kohn-Sham equations in matrix form as

$$\mathbf{F}^{(n)}\mathbf{C}^{(n)} = \mathbf{S}\mathbf{C}^{(n)}\epsilon^{(n)}, \tag{2.40}$$

with the Fock matrix $\mathbf{F}^{(n)}$, overlap matrix $\mathbf{S}$ and a diagonal matrix $\epsilon^{(n)}$ containing the orbital energies:

$$\epsilon_i^{(n)} = \begin{pmatrix} \epsilon_1^{(n)} & & \\ & \ddots & \\ & & \epsilon_N^{(n)} \end{pmatrix}. \tag{2.41}$$

Each step in the SCF procedure now consists of solving this generalized eigenvalue problem for the new orbitals $\mathbf{C}^{(n+1)}$, which are then used to construct the Fock matrix for the next iteration.

Prior to introducing a basis, the Kohn-Sham procedure is in principle exact except for the need to approximate the exchange-correlation functional. Notably, the basis-set representation introduces an approximation, as solving this matrix-form is only guaranteed to provide an exact solution of the original Kohn-Sham equations 2.32 in the limit of an infinite, complete basis set.

The number of basis functions directly impacts the size of the matrices which are diagonalized in the SCF iteration, and thereby the computational cost of the method.

Hence, choosing an appropriate basis set is crucial, and this explains why dozens of basis-sets with different tradeoffs between precision and performance, specialized to different applications were developed [21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32].

We perform all KS calculations using the `PySCF` library [33, 34, 35].

Gaussian Basis Functions

A commonality between the basis sets which are most frequently used for molecular applications in practice is the functional form of the basis functions: *Atom-centered Gaussian basis functions* are given by the product of a Cartesian spherical harmonic and a linear combination of Gaussians:

$$\eta_\alpha(\mathbf{r}) = \|\mathbf{r}-\mathbf{R}_a\|^l Y_{lm}\left(\frac{\mathbf{r}-\mathbf{R}_a}{\|\mathbf{r}-\mathbf{R}_a\|}\right) \sum_{i=1}^{N_{\text{Gauss}}} c_i \exp\left(-\alpha_i \|\mathbf{r}-\mathbf{R}_a\|^2\right), \tag{2.42}$$

with the spherical harmonic Y_{lm}, for a basis function centered at position $\mathbf{R}_a$ with angular momentum quantum numbers l and m, and N_{Gauss} Gaussians with exponents α_i and contraction coefficients c_i.

The immense popularity of this type of basis function can be explained by the fact that it allows for highly efficient computation of the required spatial integrals, e.g. for constructing the overlap matrix $\mathbf{S}$, via analytical expressions.

For most practitioners of DFT, this efficiency and the relative ease of use resulting from the popularity of the Ansatz outweigh its prime disadvantages: Gaussians are ill-suited to accurately represent the cusps at the nuclei, due to their smoothness around zero. Contraction of multiple Gaussians in the radial part partially alleviates this problem, but never solves it completely. Furthermore, the actual ground-truth density decays exponentially with distance to the molecule far from the nuclei, not like the squared exponentials of the Gaussians. Basis sets which are better adapted to both of these aspects exist, but are less frequently used due to various reasons, for instance their typically higher computational cost (e.g. Slater type orbitals [36, 37]) and complexity (e.g. Muffin tin orbitals [38]).

2.3.2 Direct inversion of the iterative subspace

Directly implementing the SCF procedure as described above leads to computations that can be slow to converge, and in some cases fail to converge at all. One method to greatly improve the convergence is the direct inversion of the iterative subspace (DIIS) method [20]. The idea is to use the history of the SCF procedure to construct a linear combination of previous Fock matrices, as the best guess for the ground state Fock matrix in the next iteration, instead of simply using the Fock operator

constructed from the current orbitals. Under the assumption that the errors lie in a linear subspace, a least-squares fit is performed to find the coefficients $\pi_k^{(n)}$ of the linear combination such that the expected error according to the linear model is minimized, for details see [20]. This amounts to an adjusted update rule for the Fock matrix:

$$\mathbf{F}^{(n)} = \sum_{k=0}^{n-1} \pi_k^{(n)} \mathbf{F}_{[\rho_{[\mathbf{\Phi}^{(k)}]}]} \,. \tag{2.43}$$

2.3.3 Scaling considerations

The re-introduction of the auxiliary wave function in Kohn-Sham DFT, comes with a cost: The time-scaling of the method becomes $\mathcal{O}(N^3)$, cubic with system size[3], as in every step, the Fock matrix has to be constructed and diagonalized. Furthermore, naive implementation of the Hartree term in a finite basis even results in a complexity of $\mathcal{O}(N^4)$, as it includes all combinations of four basis functions. This can however be mitigated using density fitting, which can reduce the number of basis functions in the integral from four to two, but entails cubic scaling in the fitting procedure itself.

Overall, the cubic scaling of Kohn-Sham DFT is a major bottleneck for the method, and has motivated the search for alternative formulations of DFT which avoid this scaling, such as orbital-free DFT, which we will discuss in the next section.

Attempts which aim to improve the scaling behavior within the framework of KS-DFT have been made ([39, 40, 41, 42]), but they entail disadvantages in precision and generality, and have not yet been able to replace the standard Kohn-Sham DFT in most applications. Here, we instead choose to pursue the development of the pure form of DFT that does not require the introduction of auxiliary orbitals, which we will describe in the next section.

2.4 Orbital-free DFT

Orbital-free DFT (OF-DFT), sometimes also "pure DFT", aims to avoid the cubic scaling by finding the ground state by direct optimization of the energy functional 2.19 in terms of the electron densty $\rho(\mathbf{r})$:

$$\rho^* = \underset{\rho:\, \int \rho(\mathbf{r})\mathrm{d}\mathbf{r} = N_\mathrm{e}}{\arg\min} \; E[\rho] \,. \tag{2.44}$$

The prime challenge in this approach is the kinetic energy functional $T_s[\rho]$. If an exact or satisfactory approximate expression for this functional is known, its favorable scaling might grant OF-DFT the potential to match and even surpass the role of Kohn-Sham DFT as the workhorse of quantum chemistry.

[3] Think $N = N_\mathrm{e}$ or $N = A$.

2.4.1 The kinetic energy functional

Above, we stated that no exact expression for the kinetic energy functional $T_s[\rho]$ is known. While true in general, there are some notable special cases where an exact expression is known:

Thomas-Fermi kinetic energy

The simplest approximation to the kinetic energy functional is the Thomas-Fermi kinetic energy functional, which is based on the Thomas-Fermi model of the free electron gas [43, 44]:

$$T_{\mathrm{TF}}[\rho] = \underbrace{\frac{3}{10}\left(3\pi^2\right)^{2/3}}_{c_{\mathrm{TF}}} \int \rho(\mathbf{r})^{5/3}\,\mathrm{d}\mathbf{r}\,. \tag{2.45}$$

This functional is exact for the kinetic energy of a free electron gas, i.e. spatially constant electron density. For inhomogeneous systems, it is on its own a poor approximation not accounting for the spatial variation of the electron density and failing to reproduce molecular bonding.

von-Weizsäcker kinetic energy

The von-Weizsäcker functional [45] is another special case where an exact expression for the kinetic energy functional is known. It describes single-orbital systems with at most two electrons:

$$T_{\mathrm{vW}}[\rho] = \frac{1}{8}\int \frac{\left(\nabla\rho(\mathbf{r})\right)^2}{\rho(\mathbf{r})}\,\mathrm{d}\mathbf{r}\,. \tag{2.46}$$

Similar to the Thomas-Fermi functional, this functional does not provide a good approximation outside its original domain of validity. Scaled versions of the von-Weizsäcker functional have been used as corrections to the Thomas-Fermi model.

Classical approximations

Countless further attempts to approximate the kinetic energy functional have been made. The classical, not machine-learned approaches fall in two classes, namely one- and two-point approximations. The former express the kinetic energy density based on the electron density and its derivatives at a single point, while the latter involves two-point spatial integrals. Generalized Gradient Approximations (GGA) [46] are a popular type of approximations among the former. They employ the dimensionless reduced energy gradient

$$s(\mathbf{r}) = \frac{\|\nabla\rho(\mathbf{r})\|}{(3\pi^2)^{1/3}\rho^{4/3}(\mathbf{r})} \tag{2.47}$$

to approximate the kinetic energy by parametrizing an enhancement factor $F(s)$ to the Thomas-Fermi kinetic energy density (see 2.4.1):

$$T_{\mathrm{S,\,GGA}}[\rho] = c_{\mathrm{TF}} \int \rho^{5/3}(\mathbf{r})\, F(s(\mathbf{r}))\, \mathrm{d}\mathbf{r} \tag{2.48}$$

One instance of this class is the APBEK functional [47], which we employ in chapter 5 as a reference functional for Δ-learning.

Up to fourth-order gradient expansions have been employed to approximate the kinetic energy functional, however they have proven inferior to their second-order counterparts regarding generalization across systems [7].

Classical approximations have demonstrated success in certain areas such as metals and semiconductors [48, 49] or warm dense matter [50]. So far, no satisfactory approximation achieving chemical accuracy for general molecular systems has been found, hindering the applicability of OF-DFT in many fields. For an extensive comparison of classical approximations to the kinetic energy functional and their respective merits, we refer the interested reader to [2].

2.4.2 Density representations for OF-DFT

Just as for KS-DFT (section 2.3.1), the quantity to be optimized over, here the electron density $\rho(\mathbf{r})$, must be represented in a numerical form.

Using the same density representation as in Kohn-Sham DFT (equation 2.35) may be technically possible, and even has appealing aspects such as simplifying training data generation with KS-DFT. However, it is not available here if one hopes to achieve linear scaling with system size.

There are multiple ways to represent the electron density $\rho(\mathbf{r})$ in orbital-free DFT which do not rely on the Kohn-Sham orbitals, and scale linearly with system size, some of which I will describe and discuss in the following, including their implications for the efficiency and accuracy of OF-DFT calculations.

Integration grids

Possibly the most straightforward way to represent a given electron density is to evaluate it on a grid of points in space. Formally, let $\{\mathbf{r}_k\}_{k=1,\dots,N_{\mathrm{grid}}}$ be a set of points in space. Then, the electron density can be represented as a vector $\boldsymbol{\rho} \in \mathbb{R}^{N_{\mathrm{grid}}}$ with components $\rho_i = \rho(\mathbf{r}_i)$. In order to compute integrals of quantities defined on the grid, the grid points can be equipped with weights $\{w_k\}_{k=1,\dots,N_{\mathrm{grid}}}$, to define a quadrature grid such that the integral of a quantity $f(\mathbf{r})$ over space can be approximated as $\int f(\mathbf{r}) d\mathbf{r} \approx \sum_k f(\mathbf{r}_k) w_k$.

Cartesian grids

This approach has successfully been used with a simple uniform on simple 1D systems [51], and with a Cartesian three-dimensional grid in a study of machine-learned OF-DFT on graphene [6].

For three-dimensional molecular systems however, the number of grid points required to represent the electron density with sufficient accuracy on a Cartesian grid can quickly become prohibitively large, as the cusps at the nuclei require a high density of grid points to resolve.

Radial grids

An alternative approach is to use radial quadrature grids, where grid points are placed on a set of concentric spheres around the nuclei. This has the great advantage that, by using a smaller spacing of shells near the nuclei, the grid resolution can be adapted to be high around the nuclei, where the electron density has cusps, and lower further away, where the density is smoother, allowing for a much lower number of required grid points. Regarding the exact placement of shells, the sampling of points on each shell, and the weights of the grid points, there are multiple choices to be made. Here, we will draw from prior work and use Treutler-Ahlrichs radii for the shells [52], and the Lebedev quadrature for the sampling of points on each shell [53].

We will take this approach to present the electron density to the machine learning model in chapter 4.

Scaling considerations

While radial grids are typically much more efficient than Cartesian grids for molecular systems, and their number of grid points scales linearly with system size, dealing with them still becomes very expensive for larger systems especially when computations should utilize limited GPU memory. Below I will discuss some alternative representations of the electron density which can be more efficient in terms of memory usage.

Notably, the computation of most classical approximations to the exchange-correlation energy also require evaluations on a quadrature grid. Hence, if one wants to run OF-DFT on systems so large that the use of a grid becomes prohibitively expensive, one has to find approximations not only of the kinetic energy, but also for all contributions that usually require a grid to evaluate.

Square of single "orbital"

Another way to represent the electron density is to use the square of a single "orbital" ϕ, which is not an orbital in the Kohn-Sham, or even a physical sense, but rather a simple function which is used to represent the electron density:

$$\rho(\mathbf{r}) = \phi(\mathbf{r})^2 = \left(\sum_\nu c_\nu \chi_\nu(\mathbf{r}) \right)^2 . \tag{2.49}$$

Here, the $\chi_\nu(\mathbf{r})$ are a set of atom-centered Gaussian basis functions, and c_ν are coefficients representing the electron density. Other than being more memory efficient than a grid, this representation has the advantage that any set of coefficients leads to strictly positive density, such that no un-physical negative densities can occur during optimization.

However, this potential advantage is offset by a disadvantage: If, e.g. at some point during density optimization, the "orbital" ϕ has nodal planes, i.e. its sign changes somewhere in space, the electron density will have a cusp at this nodal plane, which is unphysical. Furthermore, local optimization techniques such as gradient descent may fail to get rid of this nodal plane, since the required flip of the sign of the orbital in some regions of space is a large change in the parameter space which well may lead to worse (i.e. higher energy) intermediate densities.

This approach has been used in prior work, e.g. [8].

Linear combination of atomic basis functions

A third way to represent the electron density is to use a linear combination of atomic basis functions (LCAB):

$$\rho(\mathbf{r}) = \sum_\mu p_\mu \omega_\mu(\mathbf{r}) = \sum_{a=1}^{A} \sum_{\mu \in \mathcal{A}_a} p_\mu \omega_\mu(\mathbf{r}) \,, \tag{2.50}$$

with a set of atomic basis functions $\{\omega_\mu(\mathbf{r})\}$ and coefficients $\mathbf{p} = (p_\mu)_\mu$. In the second expression, we have made the decomposition of the basis by atom explicit, by introducing index sets $\mathcal{A}_a$ of basis functions centered on atom a (these depend on the molecule $\mathcal{M}$ at hand).

Like the square of a single "orbital" representation (see section 2.4.2), this approach is memory efficient. However, it does not have the issue of nodal planes: As the relation between electron density $\rho(r)$ and the coefficients $\mathbf{p}$ is linear, convex functionals in ρ lead to convex functions in $\mathbf{p}$, and the density optimization problem (see section 2.4.3 below) is better-behaved[4]. A drawback is that the electron density is not guaranteed

[4]The true energy functional may be non-convex globally, however convexity holds in a neighborhood of its global minimum.

to be positive, which may raise the issue of negative densities during optimization. We take no explicit measures to prevent this, but note that if the ground state density is approximated precisely, the total negative density must be negligible.

This approach has been previously used to fit KS-DFT densities in a procedure called *density fitting* [54, 55, 56] to cheapen the computation of certain integrals, e.g. the coulomb integrals in the calculation of the Hartree energy.

2.4.3 Density optimization

After the electron density has been represented in a suitable form, and the energy functional, in particular its kinetic part, has been approximated, the optimization problem 2.44 can be solved.

In comparison to KS-DFT, where SCF iterations are used to solve for the Kohn-Sham orbitals in a self-consistent manner, the optimization problem in OF-DFT is much more straightforward: Besides non-negativity, only a single constraint, the correct normalization of the electron density, has to be fulfilled.

Depending on the order of accessible and reliable derivatives of the density functional, different optimization algorithms can be used, such as simple gradient descent, or higher-order method. Enforcing the normalization constraint can be achieved via a Lagrange multiplier μ (which is also the electronic chemical potential), as we describe in more detail in section 4.4. Depending on the density representation, an alternative is to enforce the constraint across optimization steps by projection the updates. This option is used in [9], and we describe it in detail in section 5.3.

Depending on the functional approximations used, the energy functional is not guaranteed to be globally convex. Thus, in order for density optimization with a local optimization algorithm to find the global minimum corresponding to the physical ground state, a good initialization of the density optimization process is important. We comment on this issue of finding initial guesses for the density in section 5.4, where we also introduce our own version of the classical Superposition of Atomic Densities (SAD) approach.

In summary, to make OF-DFT successful, the key challenge lies in accurately approximating the total energy functional in a differentiable form that ensures a minimum near the true ground state density. Once this is achieved, finding this minimum by optimizing the density becomes a relatively straightforward task.

Chapter 3

Geometric deep learning for atomistic systems

In this chapter, we will introduce the field of geometric deep learning, and discuss some of the methods which apply it to atomistic systems. First, we will introduce the concept of message passing graph neural networks, which are the most widely used architecture for learning from graph-structured data in section 3.1. Then we will introduce the concept of equivariance in section 3.2, and discuss two methods for building equivariance into neural networks, tensor fields and local frames. Finally, we will introduce a list of desiderata for a machine-learned energy functional for orbital-free density functional theory in section 3.3, which will guide our architectural choices in the following chapters.

For a systematic introduction to the field of geometric deep learning we refer the reader to [5], and to [57] for a recent overview of its applications to atomistic systems.

3.1 Permutation invariance for graphs

In machine learning sense, a molecule $\mathcal{M}$ can be seen as a point-cloud, an unordered collection of points $\mathbf{R}_a$ in $\mathbb{R}^3$, possibly each equipped with some features. For a molecule, these features at least include the atomic numbers Z_a. Either explicitly by e.g. inferring chemical bonds, or implicitly inside the network e.g. via a distance cutoff, the point cloud is typically equipped with a graph structure, connecting certain points and thereby defining neighborhoods.

When designing machine learning models for this input modality, special care has to be taken in order to guarantee that the order of the points (atoms) does not matter. Graph Neural Networks (GNNs) [58, 59] obey this property. They rely on the concept of *message-passing*: The features f_a of some node a are processed jointly with those of

neighboring nodes b to yield messages $m_{a,b}(f_a, f_b)$ which are then accumulated in an permutation-invariant manner, e.g. by summation over b. The resulting feature vector is used to update the features at node a, and the process is repeated for all nodes. Multiple such message-passing layers are stacked on top of each other, yielding a deep neural network which is agnostic to the order in which the nodes are presented. If a graph-level prediction, such as the energy of a molecule, is required, a final aggregation layer summarizes the node-level features. For extensive properties such as the energy, an appropriate choice is to simply sum up the contributions of the individual nodes (i.e. atoms).

In recent years, attention mechanisms [60] have been successful in many disciplines of machine learning. Two architectures which employ attention for graph learning and have been shown to work well for atomistic data are the Graphformer [61] and the EquiformerV2 [62], which we will use in chapters 5 and 6 of this work.

3.2 Equivariance

Symmetry is a key concept in physics, widely used to simplify problems and to make predictions. Molecular systems are no exception: Additionally to the discrete permutation symmetry discussed in the previous section, the energy of a molecule is invariant under rotations and translations, and the potential is equivariant to them. In the context of machine learning, it has been shown that integrating such symmetries in neural networks can lead to more data-efficient learning and better generalization [63].

Formally, a function $f : V \to W$ between two vector spaces on which a group G acts via representations R_V and R_W is equivariant with respect to G if

$$f(R_V v) = R_W(g) f(v) \quad \forall v \in V, g \in G\,, \tag{3.1}$$

i.e. transforming the input by g and then applying the function is the same as applying the function and then transforming the output by g. If the representation R_W is the trivial representation, i.e. $R_W(g)$ the identity on W for all g, the function is invariant to the group action, which is a special case of equivariance.

There are multiple ways to build equivariance into neural networks, two of which we will discuss in the following subsections.

3.2.1 Tensor fields

One way to build equivariance into neural networks is demanding that all operations are equivariant, giving rise to equivariant features in all layers of the network and

ultimately guaranteeing an equivariant output.

This approach was pioneered in [64] in the context of image processing, and later adapted to 3D data [65, 66]. A Python library implementing this framework, `e3nn`, has been developed, see [67]. These approaches decompose the features into groups which each transform according to some irreducible representation of the rotation group, i.e. tensor fields.

A more recent architecture utilizing this framework and incorporating attention layers which has shown impressive results on atomistic data is the EquiformerV2 [62], which we adapt to the task of learning energy functionals (see section 5.6.2) and use to great effect in chapter 6.

One disadvantage of this approach is that equivariant replacements for most standard operations such as convolutions and nonlinearities are necessary, which tend to be both computationally expensive and may restrict the expressiveness of the model. Canonicalization, which we will introduce in the following section, avoids these problems, while introducing its own challenges.

3.2.2 Canonicalization via local frames

An alternative approach to equivariance is canonicalization. The idea is to define a canonical representation of the input data, which renders the features invariant to the symmetries of the problem. These invariant features can then be processed using arbitrary operations, yielding an invariant output. If the target transforms under a non-trivial representation, the invariant output is subsequently mapped back from the canonical frame to the original frame, resulting in an equivariant output [68]

For rotational equivariance of atomistic systems, this amounts to defining one or multiple coordinate frames that rotate along with the molecule. In these local frames, the input features (such as density coefficients in the LCAB Ansatz, see 2.4.2 and 5.2) are invariant to rotations, and arbitrary models can be used.

One way to canonicalize features associated with the individual atoms in a molecule, is to choose a local frame for each atom. In [9], this is done by pointing the first axis towards the closest non-hydrogen atom, the second axis towards the second-nearest neighbor (while orthogonal to the first), and completing to a right-handed coordinate frame via a cross-product. See figure 3.1 for an illustration of the resulting frames for an ethanol molecule.

A conceptual disadvantage of rotational equivariance via canonicalization is that it necessarily introduces discontinuities in the input geometry [69]. As long as geometries are treated independently, this is likely no major issue, but it might prove detrimental to geometry optimization.

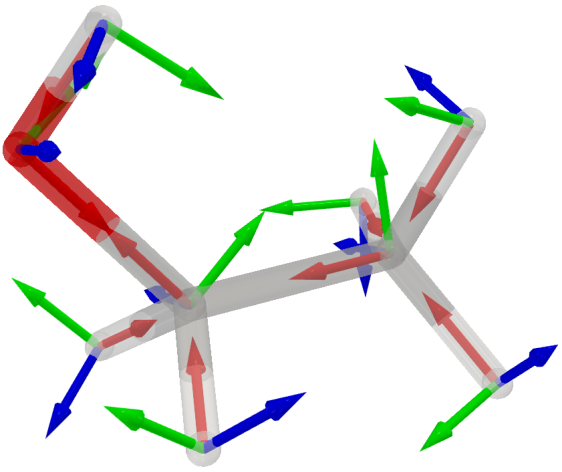


Figure 3.1. Local frames for an ethanol molecule. At each atom, a local frame is defined by the three nearest non-hydrogen neighbors. The first axis (red) points towards the nearest neighbor, the second (green) is orthogonal to the first and points towards the second nearest neighbor, and the third (blue) is orthogonal to the first two.

3.3 Architectural desiderata for modelling $E[\rho]$

There are a number of criteria that a machine learning model for the energy functional should satisfy, which we used to guide our architectural choices.

Non-locality

In the quantum chemistry community, functional approximation is called "local", if it is computed as the spatial integral over an energy density $t(\mathbf{r})$ which depends only on the electron density $\rho(\mathbf{r})$ at the respective point $\mathbf{r}$ in space. If also derivatives of the density at $\mathbf{r}$ are included, the functional is called "semi-local".

While much effort has been put into fitting the kinetic energy functional in such a local or semi-local manner (see section 2.4.1), no such functional has been found that works well for general molecular systems. Furthermore, one can include higher order derivatives in semi-local approaches, but this is already numerically challenging for the 4th order [7] and going beyond does not improve the results.

This is why we turn to a non-local approach, where the energy is predicted as a functional of the whole density, and not just its local properties.

Locality

While we expect the energy functionals that we aim to learn to be non-local in the sense described above, we do expect some degree of locality in a different sense: For most chemical systems, chemists are able to make predictions based on "local" structures spanning only a couple handfuls of atoms, such as functional groups. Thus, it is reasonable to expect that, for such systems, the energy functional can be learned from local structures without the need to consider the whole system at once (except in a final step, where the local predictions are combined to a global one, e.g. by summation).

We hence aim to design our model in such a way that it can learn from local structures. A secondary motivation for this lies in the fact that this is especially important for generalization in system size, as a model with a limited field of view is likely to also work well on larger systems, as long as the local structures in these systems are well represented in the training data.

Variational prediction

Ideally, the gradient of the total energy functional $\frac{\delta E[\rho]}{\delta \rho(\mathbf{r})}$ should be predicted in a variational manner, i.e. as the actual gradient of the total energy functional, including the machine-learned part. This is desirable on a theoretical level, since the variational principle lies at the heart of DFT, and learning a "potential" independently of the density would be a step away from this. Furthermore, for orbital-free geometry optimization, ground state electron densities in the sense of $\frac{\delta E}{\delta \rho} = 0$ are required, because otherwise the analytical nuclear gradients cannot be calculated.

Equivariance

The true energy functional is invariant to rotations and translations of the electron density, and the potential $\frac{\partial E[\rho]}{\partial \rho(\mathbf{r})}$ is equivariant to them. Hence, it is natural to require the same of the machine-learned part of the energy functional, especially as machine-learning methods have been developed to handle these symmetries (see section 3.2). Apart from the general motivations for building equivariance into models, in the case of an energy functional it is additionally clearly advantageous to fulfill these symmetries as precisely as possible, as a method which depends on the orientation of the input molecule would be at least inconvenient, if not unusable in practice. That being said, it may be feasible to learn very precise functionals without building in equivariance. For the aforementioned reasons, we still consider it a desideratum.

Chapter 4

Learning a transferable kinetic energy functional on quadrature grids

This chapter is based on the article "KineticNet: Deep learning a transferable kinetic energy functional for orbital-free density functional theory" [70], which is the result of a collaboration between Roman Remme, Tobias Kaczun, Maximilian Scheurer, Andreas Dreuw, and Fred A. Hamprecht.

Author Contributions

Roman Remme: Data Curation (equal); Methodology (lead); Software (equal); Writing/Original Draft Preparation (lead); Writing/Review & Editing (equal).
Tobias Kaczun: Data Curation (equal); Methodology (supporting); Software (equal); Writing/Original Draft Preparation (supporting); Writing/Review & Editing (equal).
Maximilian Scheurer: Software (equal); Writing/Review & Editing (supporting).
Andreas Dreuw: Conceptualization (supporting); Supervision (equal); Writing/ Review & Editing (supporting).
Fred A. Hamprecht: Conceptualization (lead); Supervision (equal); Writing/Review & Editing (supporting).

4.1 Introduction

Kohn-Sham density functional theory (KS-DFT, see section 2.3) has become the workhorse of quantum chemistry thanks to its appealing trade-off of computational cost vs. accuracy of molecular property predictions. Even so, its use of orbitals and resulting cubic scaling with system size precludes its application to larger systems with thousands of atoms that are needed to faithfully model, e.g., macromolecules in solution. The main reason that KS-DFT needs orbitals in the first place is that, despite decades of theoretical work, a concrete recipe to accurately compute the non-interacting kinetic energy $T_s[71]$ from the electron density has remained elusive; whereas it can be computed from Kohn-Sham orbitals ϕ_i via $T_s = \int t_s(\mathbf{r}) d^3\mathbf{r}$ with a kinetic energy density

$$t_s(\mathbf{r}) \quad = \tfrac{1}{2} \textstyle\sum_{i=1}^{N} |\nabla \phi_i(\mathbf{r})|^2 \,. \tag{4.1}$$

Yet, the mesmerizing promise of the Hohenberg-Kohn theorems is that it suffices to solve a single integrodifferential equation for the density $\rho(\mathbf{r})$ to find the ground state of a system, provided we find a concrete means to compute T_s and the kinetic potential (its functional derivative with respect to the electron density $\frac{\delta T_s}{\delta \rho}$) as a functional of the electron density only.

Extensive theoretical and experimental work has shown that the kinetic energy density is not merely local or "semi-local", i.e., $t_s(\mathbf{r})$ is not a function of the electron density $\rho(\mathbf{r})$ and its spatial derivatives only. On the other hand, aromatic systems and conductors aside, chemistry exhibits a large degree of locality, suggesting that it should be possible to learn a kinetic energy density functional that generalizes across relevant swathes of chemical space.

In response, we here propose a deep equivariant neural network architecture to approximate the kinetic energy density $t_s(\mathbf{r})$. Specifically, in this chapter we make the following contributions:

- We propose an equivariant deep architecture ingesting an electron density represented on a quadrature grid along with nuclear locations and charges, and predicting a kinetic energy density on the same grid.
- We show how to generate varied electron densities and associated kinetic energy potentials needed to achieve convergence when initiating density optimization far from the ground state.
- We demonstrate orbital-free density optimization in systems with two electrons, reproducing bonding with chemical accuracy.

- We offer machine learned functionals for the kinetic energy density and gradient which yield chemical accuracy in OF-DFT calculations, generalizing over input electron densities, external potential and molecular geometry.

Related Work

Machine learning has been used to improve DFT pipelines before. A large number of works[72, 73, 74, 75, 76, 77] focus on learning an approximation to the exchange correlation (XC) functional, where Kirkpatrick et al.[72] recently demonstrated impressive results. Dick and Fernandez-Serra[75] use an architecture that is similar in some respects to learn the XC functional. However, they move to invariant features early on in their model and do not learn the atomic representations, relying instead on hand-crafted features to encode the atomic environments. They train their model to only predict the total XC energy as a scalar, and compute the XC-potential in a variational manner by back-propagating through their model. A number of works demonstrate the potential of ML for OF-DFT on one-dimensional data sets, such as Snyder et al.[78] Meyer et al.[51] and Saidaoui et al.[79]. The approach of Ghasemi and Kühne[80] works in 3D, but on single, rotationally symmetric atoms only, effectively reducing the dimensionality to one. Golub and Manzhos[7] take a semi-local approach to the 3D problem as they train a neural network that takes five features reflecting the electron density, its gradient as well as its Laplacian to model the kinetic energy density, which they apply to each grid point individually. Seino et al.[81] and Fujinami et al.[8] show promising results for learning the kinetic energy and potential for molecules, however their models generalize only over different densities and hence only work for a single molecule with fixed geometry at a time. Ryczko et al.[6] learn the kinetic functional on a voxel-grid representation that works well for their application to graphene lattices, but is less suited for molecules. They present one of the few approaches with successful density optimization, however only for a learned functional that is trained to mimic the flawed Thomas-Fermi approximation. The most extensive results including density optimization are presented by Imoto et al.[82]. Like Golub and Manzhos, they use a simple neural network applied pointwise and learn an enhancement factor to the Thomas-Fermi (TF) functional in a way that guarantees both correct scaling and asymptotic behaviour. They outperform classical approximations, but not by the extent required to reach chemical accuracy.

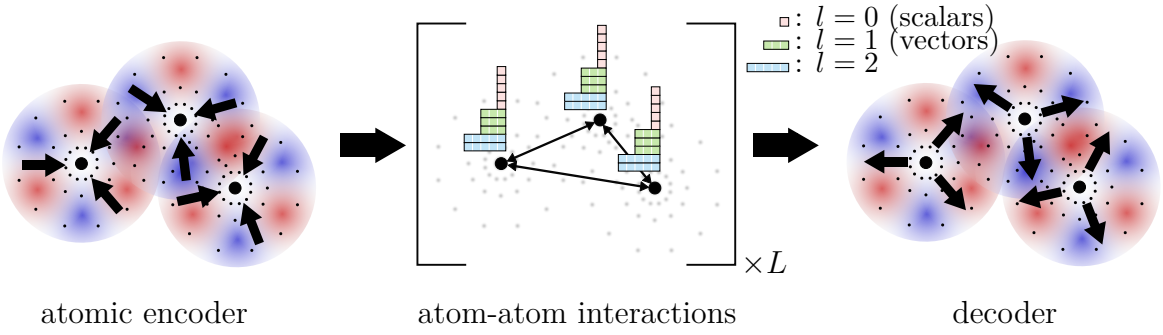


Figure 4.1. The proposed KineticNet architecture is an equivariant deep neural network with three types of layers: First, an atomic encoder relying on point convolutions (eq. 4.2) to summarize the density information on the quadrature grid in terms of tensorial features associated with the nuclei; then a number L of atom-atom interactions layers; and finally a decoding layer making predictions at all grid points.

4.2 KineticNet: a deep equivariant architecture

When developing the architecture for our machine learning model (figure 4.1), we were guided by a number of physically motivated constraints: Firstly, input and output should be represented on the quadrature grid (consisting of grid points evenly distributed on each of a number of spherical shells arranged around each atomic nucleus), such that it can seamlessly replace existing functional approximations. Secondly, the model should be equivariant with respect to the group $E(3)$, i.e. rotations and translations of the input molecule should not change the predicted kinetic energy, and the predicted kinetic potential should be transformed in accordance with the input. Finally, the field of view, i.e. the spatial extent of the input grid points that influence the output at a given point, should span several bond lengths. On the other hand, the model should still be local in the sense that for very big molecules, only nearby atoms influence the prediction, thus conceptually allowing for the generalization towards bigger molecules.

We guarantee translational equivariance by only using relative positions in our model, and rotational equivariance by using equivariant convolutions as presented in Tensor Field Networks [65] and implemented in the e3nn library [67]. This amounts to decomposing convolutional filters F into a radial part R depending on the distance $r = \|\mathbf{r}\|$ and an angular part Y, depending on the direction $\hat{\mathbf{r}} = \mathbf{r}/r$. The former is learned and the latter is given by the spherical harmonics (depending on the representation of the in- and output features of the convolution):

$$F_{cm}^{(l_f,l_i)}(\mathbf{r}) = R_c^{(l_f,l_i)}(r)\, Y_m^{(l_f)}(\hat{\mathbf{r}}) \tag{4.2}$$

with non-negative integer rotation orders of the input and filter l_i and l_f, channel index

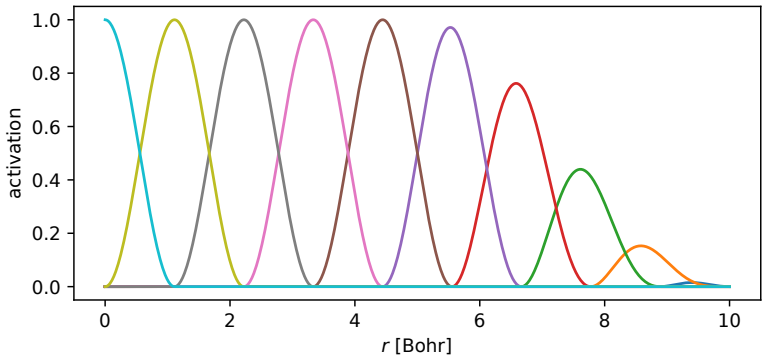
c and order inside the representation $m \in \{-l_f, .., l_f\}$. Multiplying the filters with the input features and computing a certain linear combination, using Clebsch-Gordan coefficients as weights, yields equivariant output features of the point convolution. We learn separate convolutional filters for each element and use tensorial features up to order $l = 4$.

To achieve a sufficient field of view while keeping the computational cost tractable, we use an encoder-decoder structure: In a first atomic encoding layer, we use a point convolution to compute features at every atom of the molecule (and not every input grid point). This is followed by a number of atom-atom interaction layers, each of which consists of a point convolution with the atomic nuclei positions as in- and outputs, followed by a nonlinear activation function. These layers are computationally cheap and greatly increase the field of view. Finally, a decoding layer, again a single point convolution, propagates the information back to the quadrature grid. This architecture has a sufficient field of view to capture functional groups and some of their context in molecules, while still being local and allowing for the generalization over molecule compositions. The learned atomic encoding layers are one advantage over prior work, as most commonly[75, 83] handcrafted features are used to encode the local environments of the atoms. When predicting energy densities, we additionally scale the output with a Superposition of Atomic Densities (SAD) (commonly used as an initial guess in KS-DFT), allowing the model to predict the correct asymptotic behaviour for larger distances from the atoms. In particular, the prediction of very small values becomes possible in low-density regions without extremely precise tuning of the parameters of the radial models, which would otherwise be necessary.

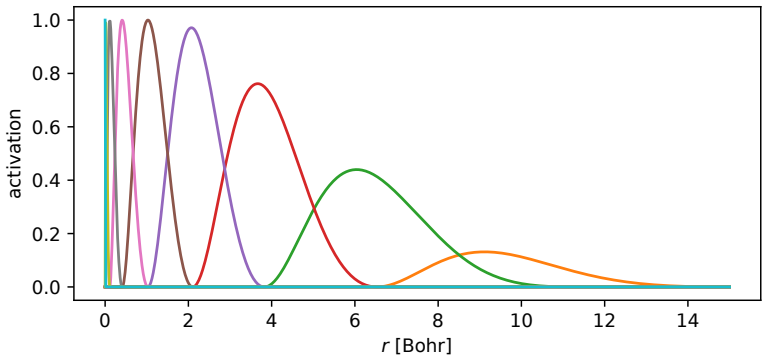
As a loss we use a smooth L1-loss, applied to the point-wise difference between the kinetic energy density and potential predictions and the corresponding ground truth on the grid. We use an adaptive scale parameter for the transition between the quadratic and linear regimes of the loss, the parameter grows linearly with the target value, but we threshold this value with $10^{-6}\,\mathrm{Ha/Bohr^3}$ from below, such that the quadratic region can be reached even at grid locations with very small target values (e.g. far away from the nuclei).

As mentioned above, the parameters of KineticNet lie in the radial models. We parameterize them as Weiler et al.[84], by a 3-layer fully connected MLP applied to the radius encoded by a set of cosine basis functions. For the initial atomic encoding and final decoder layer, we additionally transform the input distances r with the inverse of the Treutler-Ahlrichs[52] map f_{TA} before feeding them to each radial model R_i (where i is a shorthand for indices c, l_f, l_i in eq. 4.2):

$$\hat{R}_i(\mathbf{r}) = R_i\left(f_{\mathrm{TA}}^{-1}(\mathbf{r})\right) \quad . \tag{4.3}$$



(a) radial basis



(b) transformed radial basis

Figure 4.2. Schematic of the radial basis to model R in eq. 4.2, with and without transformation to adjust to the Treutler-Alrichs shells, as used in the atomic encoding and decoding layers. We apply a smooth cutoff towards the maximum radius.

This effectively changes the radial model to have a distance-dependent spatial resolution (see figure 4.2), exactly in correspondence to the spherical shells of the quadrature grid around the atoms.

One feature of our method is that the learning of a spatial filter in terms of absolute distances allows us dealing with varying grid resolutions, i.e. spacing of radial shells and angular grids. We do not exploit this explicitly in this work, but it can be useful to speed up the training by first utilizing lower-resolution samples before fine tuning in the high resolution setting, as well as granting the added flexibility of allowing a single model to be deployed at multiple grid resolutions.

4.3 Training data generation

Sufficiently large and representative data sets are as decisive for the success of a machine learning approach as the training setup and architecture. We use KS-DFT employing the BLYP XC functional[85, 86] with the cc-pVDZ basis set[87, 88] to generate ground truth data for the supervised training of our functionals. Generating a large number of training samples is easy, but to ensure sufficient variability in the data, we had to employ a new technique that we discuss in this section.

We use eq. 4.1 to generate ground truth kinetic energy density. Many other definitions exist, in particular any additive constant to t_s that integrates to zero yields the same total kinetic energy. That said we choose eq. 4.1 over other formulations for the kinetic energy density as its values lie in a smaller range, which is preferred for machine learning models. The kinetic potential $\frac{\delta T_s}{\delta \rho}$ can be computed[89] from

$$\frac{\delta T_s}{\delta \rho} - \mu = \frac{\sum_i^N -\frac{1}{2}\phi_i(\mathbf{r})\nabla^2\phi_i(\mathbf{r}) - \epsilon_i\phi_i^2(\mathbf{r})}{\rho(\mathbf{r})} \tag{4.4}$$

where ϵ_i stands for the eigenvalue/orbital energy of the i-th KS-Orbital, ρ for the electron density and μ for the chemical potential, which is assumed equal to the energy of the highest occupied molecular orbital ϵ_{HOMO}.[89]

The derivation by King and Handy[89] equates parts of the Euler and Kohn-Sham equations, suggesting that the equation is only valid for stationary states. Yet any OF-DFT algorithm will encounter non-stationary electron densities on its way from the initial guess to the true ground state. As generalization from ground state densities to these intermediate states cannot be expected, it is crucial to also include non-ground state electron densities in the training set. Such training makes the model sufficiently robust to achieve convergence of the iterative density optimization. This necessity has also been noted by Ryczko et al.[6] who observe convergence only for a functional trained to mimic the TF approximation on a varied data set, but not for the functional

trained on KS ground truth at ground states only. In summary, the paradoxical task is to compute the kinetic potential for densities other than the true ground state while at the same time eq. 4.4 holds only for stationary states. This is where our second contribution lies.

The first Hohenberg-Kohn theorem states that a one to one mapping exists between the external potential and the ground state electron density of a system.[1] Thus, slightly perturbing the external potential of a molecule will lead to a different electron density as ground state and thereby enable the use of eq. 4.4. Exploiting this observation, we perturb the external potential $v_{\text{ext}}^{\text{mol}}(\mathbf{r})$ in KS-DFT by adding a randomly sampled symmetric matrix $\mathbf{M}$ with an appropriately chosen norm to the matrix representation of the external potential in the atomic basis $\{\chi_\nu\}$ to generate our training data:

$$[v_{\text{ext}}]_{\mu\nu} = \langle \chi_\mu | v_{\text{ext}}^{\text{mol}} | \chi_\nu \rangle + M_{\mu\nu} \tag{4.5}$$

$$\mathbf{v}_{\text{ext}} = \mathbf{v}_{\text{ext}}^{\text{mol}} + \mathbf{M} \tag{4.6}$$

The pyscf software package[90, 91] is used for this purpose as it is efficient and well suited for the integration of ML models trained with python.

For our model, as with most neural networks, it is useful if the inputs and targets have similar scales, and that their values do not vary over many orders of magnitude within a single sample (and between samples). Hence, an important detail in the training data generation is how we deal with the cusps at the atoms, of both the input electron density and the output energy density and its potential. Here, we take the approach of subtracting spherically symmetric "Atomic Contributions" (ACs) for each atom and each of the fields. We compute them by applying restricted or restricted open shell KS-DFT to each atom type, and spherically symmetrizing the result, for details see appendix A.4 (in the few cases, in which KS-DFT did not converge, these non-converged solutions still fulfill their purpose). This greatly reduces the magnitude of the cusps, see figure A.3.

Another relevant detail is the choice of target for the kinetic potential: We follow Ryczko et al.[6] and do not directly predict the kinetic potential $\frac{\delta T_s}{\delta \rho}$, but rather $\sqrt{\rho}\frac{\delta T_s}{\delta \rho}$, its product with the square root of the electron density. They report that this gives a lower training error, and we have two additional reasons to make this choice: On the one hand, the denominator in eq. 4.4 leads to numerical problems for small densities, e.g. far from the atomic nuclei, which are alleviated by taking this product. On the other hand, in our OF-DFT calculations, the kinetic potential is multiplied with the square root of the density whenever evaluated, due to our Ansatz (eq. 4.7), see section 4.4 below, hence directly predicting this quantity is natural.

We generate data sets for a number of different atoms and molecules: First, the two-electron systems He, H_2 and H_3^+, and furthermore the molecules HF, H_2O as well as two neon atoms in the vicinity of each other as an instance of a non-binding system, which we label as Ne_2. We sample the perturbation of the external potential and the molecule geometry independently for each training instance. For the linear molecules, we sample the inter-atomic distance uniformly in a range from around 0.4 Å to around 2.0 Å, thus covering both strongly compressed structures as well as nearly dissociated ones. For H_3^+, we arrange the nuclei in an equilateral triangle of side length $\sqrt{2}$ Å and perturb the position of each atom by a random vector with a length that is sampled uniformly in $[0, 0.5\text{Å}]$. Lastly, for water, we apply the following procedure: Each O−H bond length is uniformly sampled between 90% and 110% of its equilibrium value. The bond angle is varied by uniformly sampling each O−H "vector" from a 10° conus.

For each system, we generate 8000 training samples, 2000 to use for validation and an additional 1000 for testing.

4.4 Density optimization

After successfully training these models, the next logical step is to use them "in the wild", i.e. in an actual OF-DFT calculation to compute the ground state of a geometry not seen at training time. To this end, we have implemented an OF-DFT solver based on the work of Chan et al.[92] and Ryley et al.[89] to see if density optimization is possible. We use the approach in which the density ρ is represented as the square of a single "orbital", or more precisely of a linear combination of atomic basis functions χ_ν:

$$\rho(\mathbf{r}) = \Big(\sum_\nu c_\nu \chi_\nu(\mathbf{r}) \Big)^2 . \tag{4.7}$$

The coefficients c_ν are the variables which are optimized to minimize the energy functional while ensuring the correct normalization of the density. This approach allows the use of well established quantum chemical libraries for the evaluation of many of the integrals.

To achieve this optimization of the total energy w.r.t. the electron density under the constraint of its normalization to the correct number of electrons N_e, a Lagrange multiplier μ is introduced:

$$\begin{aligned} \mathcal{L} =& T_s[\rho] + \int v_{\text{ext}}(\mathbf{r})\rho(\mathbf{r})\,\mathrm{d}\mathbf{r} + J[\rho] + E_{\text{xc}}[\rho] \\ & - \mu \left(N_e - \int \rho(\mathbf{r}) \mathrm{d}\mathbf{r} \right) \end{aligned} \tag{4.8}$$

where T_s is the non-interacting kinetic energy, J the classical electron-electron interaction, and E_{xc} the exchange correlation energy. The ground state electron density is then given by the global minimum of this equation. Therefore, its functional derivative w.r.t. the electron density gives us the stationarity condition

$$\frac{\delta \mathcal{L}}{\delta \rho} = 0 = \frac{\delta T_s}{\delta \rho} + v_{\text{ext}}(\mathbf{r}) + \frac{\delta J}{\delta \rho} + \frac{\delta E_{\text{xc}}}{\delta \rho} - \mu \,. \tag{4.9}$$

Introducing the basis expansion, the gradient w.r.t. the expansion coefficients is then given by

$$\frac{\partial \mathcal{L}}{\partial c_\sigma} = 2\langle \chi_\sigma \mid \frac{\delta T_s}{\delta \rho} + v_{\text{ext}}(\mathbf{r}) + \frac{\delta J}{\delta \rho} + \frac{\delta E_{\text{xc}}}{\delta \rho} - \mu \mid \sum_\nu c_\nu \chi_\nu \rangle \,. \tag{4.10}$$

Furthermore, the Lagrange multiplier μ, which corresponds to the chemical potential, needs to be optimized:

$$\frac{\partial \mathcal{L}}{\partial \mu} = N_e - \int \rho(\mathbf{r}) \, \mathrm{d}\mathbf{r} \,. \tag{4.11}$$

To iteratively solve this constrained optimization problem, we use the SLSQP solver [94] as implemented in the scipy package [95].

For the initial guess in our optimizations we use an adapted version of a SAD guess. For this we use the atomic KS-DFT densities and fit OF-DFT density coefficients to it. Those coefficients are then used to construct a guess by simply placing them at the position of the corresponding atomic basis functions.

4.5 Computational experiments

4.5.1 Training details

We train, simultaneously but independently, two models: One to predict the kinetic energy density t_s (to be integrated to the kinetic energy T_s) and one for the kinetic potential $\frac{\delta T_s}{\delta \rho}$. Each of these models is trained on the union of all data sets. We train our models using the Adam optimizer[96] using default parameters and a learning rate of 0.01 that we decay exponentially after 10^5 training iterations. We use a batch size of 64 and train until convergence. We observe some amount of overfitting in the sense that the loss is greater during validation than training, however both decrease during the whole training procedure which allows us to simply evaluate the final saved model.

4.5.2 Test results

For all data sets the mean absolute error (MAE) of the predicted kinetic energy over 100 samples from the test set falls below the threshold of chemical accuracy, 1 mHa

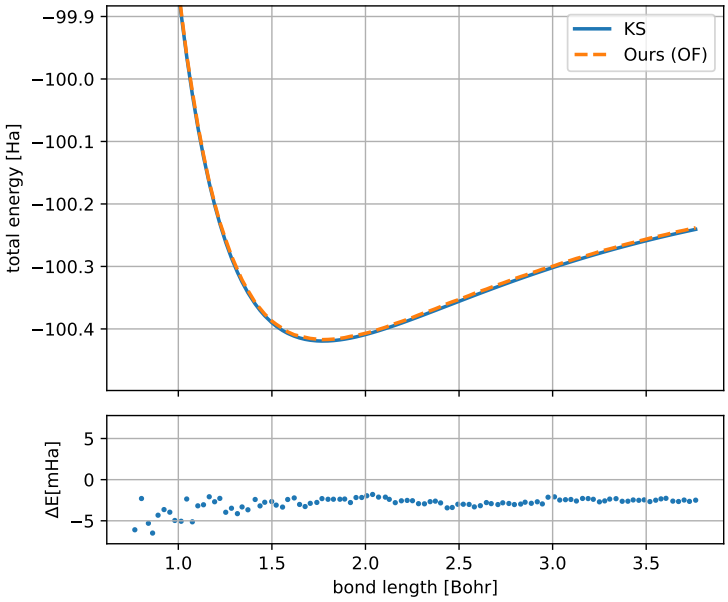


Figure 4.3. Total ground state energy of HF at different bond lengths, as computed by KS-DFT as well as the prediction of our ML functional on the KS ground state densities (without orbital-free density optimization).

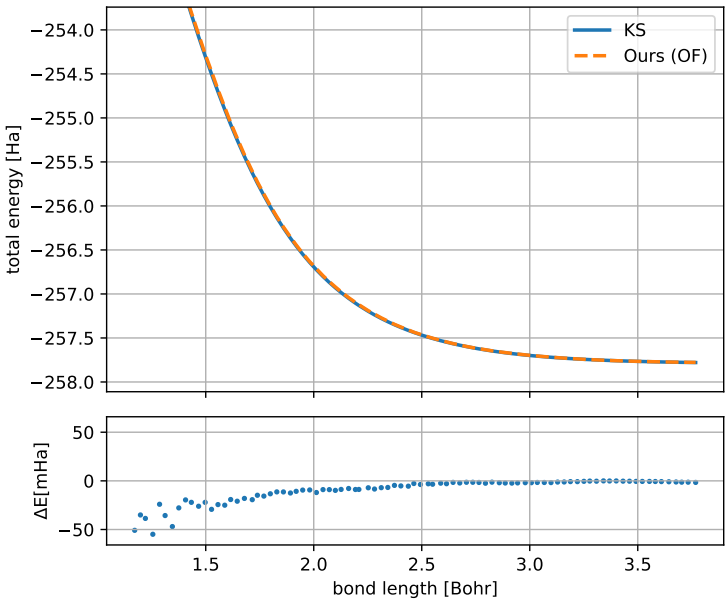


Figure 4.4. Total ground state energy of Ne_2 at different "bond" lengths, as computed by KS-DFT as well as the prediction of our ML functional on the KS ground state densities (without orbital-free density optimization).

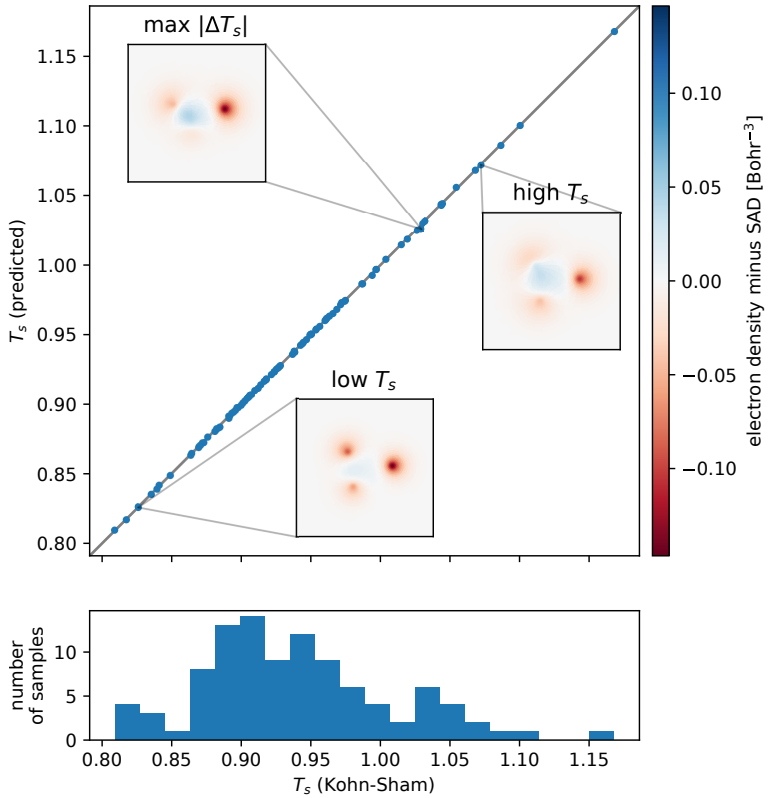


Figure 4.5. Comparison between predicted and KS kinetic energies for H_3^+ on 100 samples from the test set, and representative electron densities (minus SAD).

Table 4.1. Kinetic energy mean absolute error for our model (KineticNet) and classical functionals on test sets consisting of KS solution densities (for varying v_{ext}). Errors are given in mHa.

	He	H_2	H_3^+	HF	Ne_2	H_2O
Ours	1.3×10^{-1}	6.4×10^{-1}	4.7×10^{-1}	2.0	1.0×10^{1}	1.3
TF	2.9×10^{2}	2.1×10^{2}	2.8×10^{2}	9.1×10^{3}	2.2×10^{4}	6.9×10^{3}
vW	0	0	0	2.6×10^{4}	7.8×10^{4}	1.9×10^{4}
MGE2	7.9×10^{1}	1.0×10^{2}	1.6×10^{2}	7.8×10^{2}	2.6×10^{3}	8.1×10^{2}

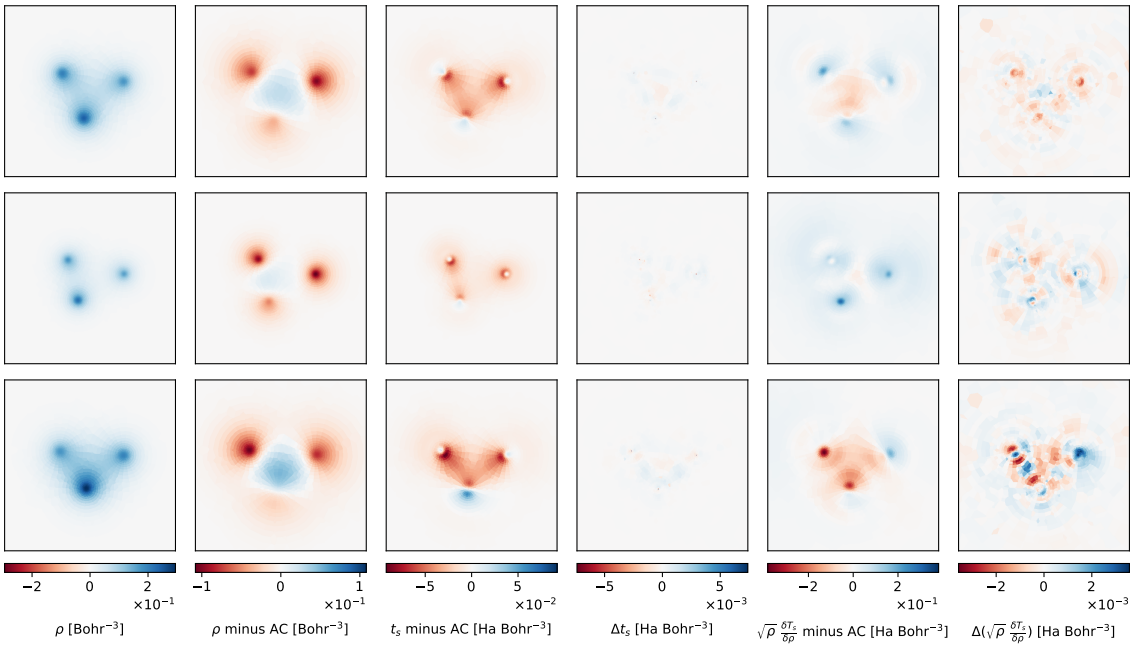


Figure 4.6. Slices through input, prediction, and error for H_3^+, on three test samples. From left to right: Electron density, electron density minus AC, predicted kinetic energy density minus AC, error of the electron density, predicted kinetic potential times square root of electron density and lastly its error.

per electron, see table 4.1. We compare these results to three classical approximations of the kinetic energy functional, the first-order Thomas-Fermi (TF) functional, the second order von Weizäcker (vW) correction and the MGE2 functional which is a linear combination of the two former approximations and which performed best in the extensive comparison by Fujinami et al.[8]. The superiority of the ML functional in this metric is very obvious as it outperforms the classical approximations by more than two orders of magnitude throughout, however with one exception: The vW functional is exact for two electron systems, hence its MAE is zero for He, H_2 and H_3^+. One could argue that the learning task for the ML model in these cases is also much easier for the ML functional, as the semilocal expression of the vW functional is already exact, but on one hand our model by construction cannot simply reproduce this term, and on the other hand we demonstrate a similar accuracy (per electron) on the bigger systems, where vW alone fails spectacularly.

For HF and Ne_2 we additionally demonstrate that our model is accurate enough to model chemical bonds (or the absence thereof) by evaluating it on the KS ground states for varying inter-atomic distances, and plotting the resulting dissociation curves in figures 4.3 and 4.4. Note that none of the geometries, nor any ground states (without a perturbed external potential) were part of the training sets. For H_3^+, a comparison

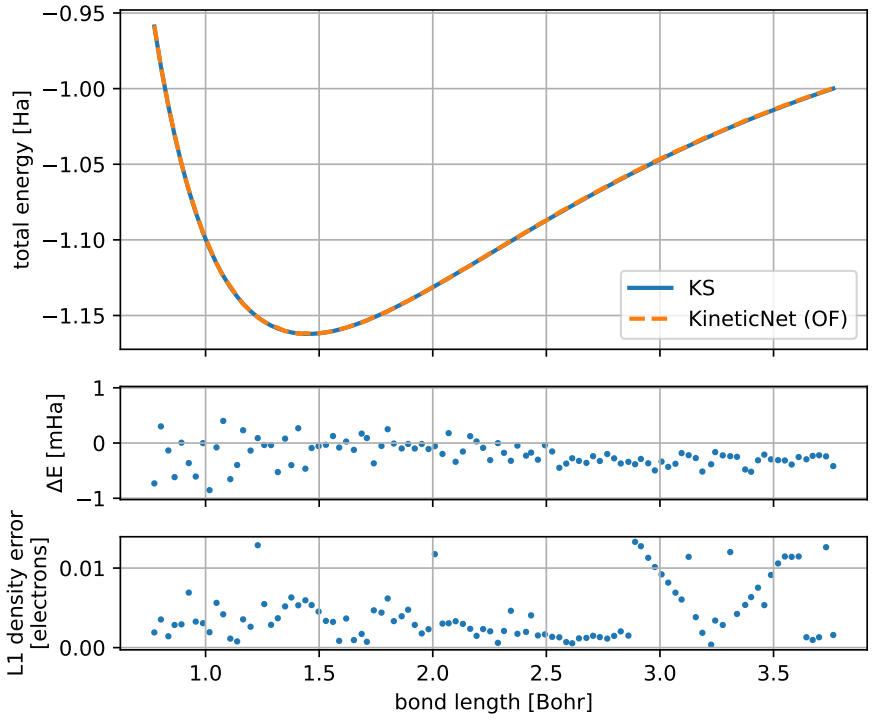


Figure 4.7. Total ground state energy of H_2 at different bond lengths, as computed by KS-DFT as well as OF-DFT using our machine learned functionals.

between predicted and target kinetic energies is shown in figure 4.5.

4.5.3 Density optimization

The results of applying our machine learned functionals in OF-DFT are summarized in table 4.2. We evaluated each of our models on 100 geometries from the corresponding test set (except of course for He, where only a single geometry is available), setting the SLSQP convergence threshold to 10^{-4} and allowing a maximum of 100 steps. While we always observe convergence using the classical functional approximations, in the few cases when no convergence is reached using our model, we evaluate the best solution obtained so far. To quantify the results, we compute the mean over the absolute energy errors, as well as the L1 density deviation

$$\|\rho - \rho_{\mathrm{KS}}\|_1 = \int |\rho(\mathbf{r}) - \rho_{\mathrm{KS}}(\mathbf{r})| \, \mathrm{d}\mathbf{r} \tag{4.12}$$

between the KS density ρ_{KS} and the result of the OF calculation ρ.

For He, H_2 and $H_3{}^+$, we obtain errors of less than 1 mHa and L1 density deviations on the order of 10^{-2} electrons, see table 4.2. This is more than precise enough to correctly model the H_2 bond, see figure 4.7.

Furthermore, the way our learned functionals generalize allows us to apply them in

Table 4.2. Density optimization results for two electron systems using our ML functional (KineticNet) as well as classical functionals. Note that the vW functional is exact for these systems, the small deviations are due to the limited steps and finite convergence threshold in our OF-DFT implementation.

		$\mathbf{v}_{\text{ext}} = 0$		$\mathbf{v}_{\text{ext}}$ from test		with solvation model	
	Data set	ΔE [mHa]	$\Delta\rho$ [a.u.]	ΔE [mHa]	$\Delta\rho$ [a.u.]	ΔE [mHa]	$\Delta\rho$ [a.u.]
Ours	He	0.02	0.0004	0.10	0.0028	0.02	0.0004
	H_2	0.34	0.0016	0.74	0.0162	0.34	0.0043
	H_3^+	0.44	0.0055	0.43	0.0105	0.44	0.0054
TF	He	381.64	1.4937	1022.02	1.6558	381.90	1.4962
	H_2	305.30	0.1282	305.26	0.1317	305.30	0.1282
	H_3^+	685.25	1.2828	727.40	1.3489	691.82	1.3057
vW	He	0.00	0.0001	0.17	0.0024	0.01	0.0001
	H_2	0.00	0.0000	0.00	0.0003	0.00	0.0000
	H_3^+	0.02	0.0007	0.02	0.0008	0.02	0.0007
MGE2	He	116.36	0.5778	531.88	1.3410	116.03	0.5802
	H_2	76.73	0.1367	76.52	0.1366	76.73	0.1367
	H_3^+	371.15	1.0247	411.11	1.1091	373.69	1.0459
LC	He	153.65	0.7310	756.51	1.5467	153.27	0.7336
	H_2	5.45	0.0878	5.83	0.0879	5.45	0.0878
	H_3^+	485.17	1.1877	535.98	1.2742	490.23	1.2150

different settings: Just as KS-DFT during data generation, we can apply orbital free density optimization on molecules in the presence of an additional external potential. For this, we use potentials from the test set and observe that the accuracy of our model is still good, see middle two columns in table 4.2.

We can also apply solvation models that simulate a chemical environment by a density-dependent contribution to the external potential. To this end we employ the ddCOSMO solvation model [97, 98, 99] as implemented in pyscf with default parameters, i.e. simulating a solution in water, see the two rightmost columns of table 4.2.

Note that none of these modifications would have been possible if we took a very direct black-box ML approach of e.g. directly predicting the ground state electron density.

The reason why we only present density optimization results for the two electron

systems He, H_2 and H_3^+ is that only for those there is an exact correspondence between the possible KS densities and the Ansatz we use in our OF calculations (eq.4.7, for details see appendix A.3). Hence, for these systems densities close to the KS ground state are obtainable. On the other hand, in the cc-pVDZ basis that we are using, it is impossible to model densities close to the KS ground state using the OF Ansatz, even fitting the coefficients to best mimic the KS density leads to an L1 deviation of multiple electrons. So while OF-DFT calculations using our learned functionals sometimes converge for these larger systems, either a sufficiently larger basis, maybe optimized for this application, or an entirely different Ansatz are required to reach quantitatively interesting results.

4.6 Conclusion

In this chapter, we presented KineticNet, a new equivariant machine learning model adapted for the prediction of molecular properties on quadrature grids. Using the electron density on the grid and the positions of all nuclei as input, it can successfully predict the corresponding non-interacting kinetic energy density for a variety of systems such as HF, H_2O and Ne_2. The new functional correctly describes both chemical bonding (as well as the absence of it in Ne_2). We offer proof of principle that this architecture can predict the kinetic potential with sufficient accuracy to allow actual OF-DFT density optimization to reach the respective KS-DFT ground state for the model systems H_2, H_3^+ and Ne. Additionally, we show that the generation of varied training data, by invoking fundamental concepts of DFT, allows training models with minimal overfitting which generalize over densities arising in the presence of different external potentials. This also includes simple solvent models such as ddCOSMO which can be applied out of the box without any additional retraining.

Given these encouraging results, the next step is to generalize the entire workflow to afford density optimization for more than two electrons. We conjecture that the principal obstacle in the setup described in this chapter is that the KS-DFT ground state cannot be represented by our combination of basis and description of the density. To overcome this limitation, we turn to the LCAB Ansatz for describing the density (see section 2.4.2) in the next chapter, in which Kohn-Sham ground densities can be represented accurately via density fitting.

Chapter 5

Orbital-free DFT using a linear combination of atomic basis functions

In March 2024, the Article "Overcoming the barrier of orbital-free density functional theory for molecular systems using deep learning" by Zhang et al. [9] was published in Nature Computational Science. They present remarkable results, achieving chemical accuracy in orbital-free density optimization for two datasets, namely ethanol structures from MD17 [100, 101] and the QM9 dataset [102, 103] of small organic molecules of up to nine heavy (i.e. non hydrogen) atoms. They use the LCAB Ansatz (see section 2.4.2) to represent the electron density, and pair it with a Graphormer architecture [61] which uses local frames (see section 3.2.2) to achieve rotational invariance. However, their work, albeit impressive, leaves some questions open, and has some key limitations, the most glaring being the lack of proper convergence in their density optimization process. Using their learned functionals does not result in local minima in the energy landscape, hence, density optimization tends to eventually diverge to energies far below the true minimum value. They address this with a complex hand-crafted criterion which picks an intermediate density from the optimization trajectory as the final result, based on the trajectories of both the energy and gradient norm. This is not only unsatisfactory from a theoretical standpoint, but also limits the applicability of their method because computation of nuclear gradients for geometry optimization requires a variational ground state in the sense of $\frac{\delta E}{\delta \rho} = 0$ [104], see section 3.3 for details. Furthermore, there are practical advantages to proper convergence: It allows stopping density optimization early when the gradient norm falls below a certain numerically-sound threshold. In case this does not happen, it serves as a valuable indication that the given system is not well represented by the

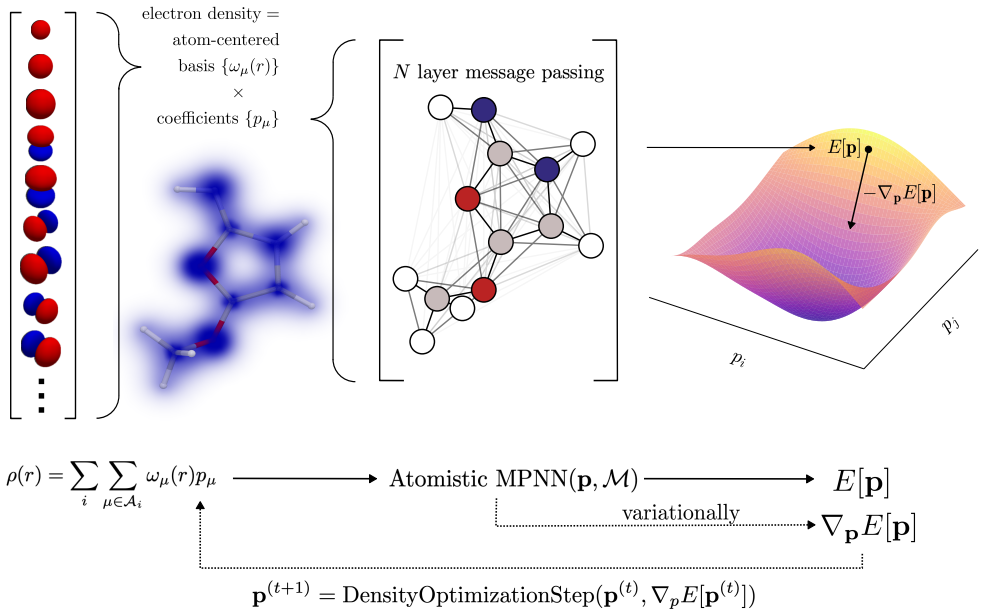


Figure 5.1. Variational prediction in the LCAB Ansatz. The electron density $\rho(\mathbf{r})$ is represented via coefficients $\mathbf{p}$ as a Linear Combination of Atomic Basis functions (LCAB), which are fed into a deep atomistic neural network to predict the total energy $E[\mathbf{p}]$ (possibly contributions such as E_H are added, not show). The variationally predicted gradient $\boldsymbol{\nabla}_{\mathbf{p}} E[\mathbf{p}]$ guides density optimization. Figure drawn by master student Johannes Schmidt.

model.

Hence, we set out to improve upon the results of Zhang et al. [9], in particular addressing the central issue of convergence. In this chapter, we will describe the method resulting of these efforts to train a variational model for OF-DFT (see figure 5.1) in detail, mostly following Zhang, Liu, You, Liu, Zheng, Lu, Wang, Zheng, and Shao [9] in parts of the pipeline that we have changed only in minor ways, and compare the results.

In the following three sections, we will describe the data generation process in the LCAB density representation (see 2.4.2), largely following [9]. However, we will be more explicit about the optimization target in density fitting (see 5.1.1), and the role of basis transformations (see 5.2). In section 5.3, we provide a more thorough description of gradient projection to deal with the offset in the gradient labels, and discuss the interplay of this projection with basis transformations.

Afterward, we present our improvements: In section 5.5, we introduce a novel method to generate perturbed data, refining the external potential perturbation introduced in chapter 4 (see section 4.3) in the context of the LCAB Ansatz. Section 5.4 then describes the generation of a cheap but accurate initial guess for density optimization. After discussing a number of conceptually minor, but practically relevant architectural improvements to the Graphormer architecture as utilized by [9], and how we adapted the Equiformer architecture [62] to our setting in section 5.6, we describe and discuss our experimental results, see sections 5.8 and 5.9.

5.1 Generating labels with KS-DFT

Zhang et al. [9] demonstrated how to generate a label for the kinetic energy functional $T_\mathrm{S}[\rho]$ and its gradient $\frac{\delta T_\mathrm{S}[\rho]}{\delta \rho(\mathbf{r})}$ from each iteration in a Kohn-Sham DFT procedure. The key insight is that $\Phi^{(n)}$, the result of any SCF iteration

$$\hat{F}_{[\rho_{[\mathbf{\Phi}^{(n-1)}]}]}\phi_i^{(n)} = \epsilon_i^{(n)}\phi_i^{(n)}\,, \tag{5.1}$$

describes the ground state of a certain non-interacting system (i.e. without E_H and E_xc), by choosing the external potential $V_\mathrm{eff}^{(n)} = V_{\mathrm{eff}[\rho_{[\mathbf{\Phi}^{(n-1)}]}]}$. Indeed, if one follows the derivation of the Kohn-Sham equations for this non-interacting system starting at

$$E^{*(n)} = \min_{\mathbf{\Phi}\text{: orthonormal}} \left[\sum_{i=1}^{N} \langle \phi_i | \, \hat{T} \, | \phi_i \rangle + \int \rho_{[\mathbf{\Phi}]}(\mathbf{r}) V_\mathrm{eff}^{(n)}(\mathbf{r}) \, \mathrm{d}\mathbf{r} \right], \tag{5.2}$$

one finds equation 5.1 as the optimality condition. As the kinetic energy operator is the only contribution to the minimized energy in equation 5.2 which explicitly depends on

the orbitals $\mathbf{\Phi}$ and not the electron density alone, it follows that the resulting kinetic energy is the minimal one for the given density $\rho_{[\mathbf{\Phi}^{(n)}]}$. This, however, is precisely how the non-interacting kinetic energy functional is defined (see equation 2.17), and hence the kinetic energy found in each SCF iteration can be used as a label for it.

Furthermore, the variation of equation 5.2 with respect to the density ρ gives rise to the optimality condition

$$\frac{\delta T_{\mathrm{S}}[\rho]}{\delta \rho} + V_{\mathrm{eff}}^{(n)} = \mu^{(n)} \quad \Leftrightarrow \quad \frac{\delta T_{\mathrm{S}}[\rho]}{\delta \rho} = \mu^{(n)} - V_{\mathrm{eff}}^{(n)}\,, \tag{5.3}$$

with the Lagrange multiplier $\mu^{(n)}$ enforcing the density normalization constraint, also called the chemical potential. Hence, we also obtain labels for the kinetic potential, up to a scalar offset $\mu^{(n)}$. How to deal with the offset in the gradient labels will be discussed in section 5.3.

We follow [9] and use the 6-31G(2df,p) basis set [105, 106, 107, 108] and the PBE exchange-correlation functional [13] in the Kohn-Sham calculations.

5.1.1 Density fitting

Density fitting is necessary, as we generate labels using Kohn-Sham DFT where densities are represented via a set of molecular orbitals, but we want to make OF-DFT work in the LCAB Ansatz for the electron density. The first step to mediating between these two density representation is density fitting: Its goal is to map density matrices $\mathbf{\Gamma}$ (see eq. 2.35) to LCAB coefficients $\mathbf{p}$ (compare eq. 2.50).

Outside OF-DFT, density fitting is used for approximating e.g. the electron repulsion integrals in Hartree-Fock and Kohn-Sham DFT methods, improving the quartic scaling of their naive computation to cubic [56].

Naively, one might think that the best way to fit the density is to minimize some distance measure, such as the L2 norm $\|\rho_{\mathbf{p}} - \rho_{\mathrm{KS}}\|_2$, between the two densities. However, if one also cares about energies, this is not the best choice, as they tend to deviate substantially if only the density difference is minimized.

Hence, we follow Zhang et. al. [9] in minimizing both the Hartree energy of the residual density, $E_{\mathrm{H}}[\rho_{\mathbf{p}} - \rho_{\mathrm{KS}}]$ as well as the squared residual external energy, $(E_{\mathrm{ext}}[\rho_{\mathbf{p}}] - E_{\mathrm{ext}}[\rho_{\mathrm{KS}}])^2$. After writing both error measures as quadratic forms in the density coefficients $\mathbf{p}$ (this involves computing coulomb integrals between pairs of basis functions), the optimization problem is solved with a least-squares solver. We follow [9] and use an even-tempered basis set for density fitting.

A systematic comparison of different optimization targets was performed by Tim Ebert and is presented in his master thesis.

5.2 Basis transformations

As described above, after generating the training data using KS-DFT, we transform into the density fitting basis, expressing the density in an LCAB Ansatz (see eq. 2.50).

This density basis is optimized to express the density as precisely as possible. As the Ansatz for the density is linear, given an invertible basis transformation matrix $\mathbf{A}$ we can define a vector of new basis functions $\boldsymbol{\omega}$ and coefficients $\mathbf{p}'$ according to

$$\mathbf{p}' = \mathbf{A}\mathbf{p} \tag{5.4}$$

$$\boldsymbol{\omega}' = \mathbf{A}^{-\top}\boldsymbol{\omega} \tag{5.5}$$

without changing the density:

$$\rho' = \boldsymbol{\omega}'^{\top}\mathbf{p}' = \boldsymbol{\omega}^{\top}\mathbf{A}^{-1}\mathbf{A}\mathbf{p} = \boldsymbol{\omega}^{\top}\mathbf{p} = \rho\,. \tag{5.6}$$

In other words, the sets of densities expressible in both basis sets is identical. It is straightforward to calculate how other quantities which depend linearly or quadratically on the basis functions, such as the basis integrals $\mathbf{w}$, gradient labels $\mathbf{g}$, overlap matrix $\mathbf{W}$ and coulomb matrix $\mathbf{C}$, transform.

Why would we want to perform such basis transformations? The two main reasons lie on the machine learning side: First of all, basis transformations can be used to achieve equivariance by transforming features into local frames, see section 3.2.2. Secondly, utilizing the fact that basis transformations change the overlap matrix $\mathbf{W}$, *natural reparametrization* is a basis transformation that diagonalizes $\mathbf{W}$. We motivate and describe it below, see section 5.2.1.

5.2.1 Natural reparametrization

In the standard, atom centered and axis-aligned density-fitting basis used in the LCAB Ansatz, the overlap matrix $W_{\mu\nu} = \int \omega_\mu(\mathbf{r})\omega_\nu(\mathbf{r})\,\mathrm{d}\mathbf{r}$ is not diagonal. When a vector of density coefficients $\mathbf{p}$ is perturbed by some $\Delta\mathbf{p}$, the change in density can be quantified via

$$\int |\Delta\rho(\mathbf{r})|^2\,\mathrm{d}\mathbf{r} = \sum_{\mu,\nu}\int \Delta p_\mu\omega_\mu(\mathbf{r})\,\Delta p_\nu\omega_\nu(\mathbf{r})\,\mathrm{d}\mathbf{r} = \Delta\mathbf{p}^{\top}\mathbf{W}\Delta\mathbf{p}\,. \tag{5.7}$$

Hence, if $\mathbf{W}$ is not diagonal, the magnitude of the change in density will depend on the direction of the perturbation in coefficient space. For instance, if the overlap matrix has very small eigenvalues, even significant change of the coefficients in the corresponding directions will have little effect on the density. This is suboptimal for machine learning, as very similar densities may be assigned very different energies by

the model, if their coefficients are very different. This adds to the original motivation for diagonalizing the overlap matrix given in [9], which is reducing the variance in coefficient and gradient scale across coefficients.

From equation 5.7 we can see that transforming the coefficients $\mathbf{p}$ by a matrix $\mathbf{A} = \mathbf{M}^\top$

$$\mathbf{p}' = \mathbf{M}^\top \mathbf{p} \tag{5.8}$$

with $\mathbf{M}\mathbf{M}^\top = \mathbf{W}$ will result in an isotropic change in density $\int |\Delta\rho(\mathbf{r})|^2 \, \mathrm{d}\mathbf{r} = \Delta\mathbf{p}'^\top \Delta\mathbf{p}'$ as the overlap matrix is diagonalized. The transformation matrix $\mathbf{M}$ is not unique, but only determined up to an orthogonal transformation, i.e. rotation in coefficient space. One natural choice was first described by Löwdin [109], which minimizes the distance between original and transformed basis functions. This also leads to the procedure to be equivariant to permutations of the basis functions, which is critical as we want our model to fulfill this symmetry (the order of atoms in the molecule should not matter). Following [109] and [9][1], we compute the eigenvalue decomposition of the overlap matrix $\mathbf{W} = \mathbf{U}\mathbf{\Lambda}\mathbf{U}^\top$ and set $\mathbf{M} = \mathbf{U}\mathbf{\Lambda}^{-1/2}\mathbf{U}^\top$.

A conceptual disadvantage of this basis transformation is the cubic scaling of the required eigenvalue decomposition. While it is only necessary once per molecule, this still is prohibitive for true linear scaling. Hence, local versions of the natural reparametrization which diagonalize the overlap matrix only for a subset of the basis functions while scaling linearly might be a promising avenue for future research.

5.3 Gradient projection

Introducing a gradient projection is motivated by two reasons: Firstly, as described in section 5.1 and visualized in figure 5.2, while we can use KS-DFT to generate labels on an absolute scale for the different contributions to the total energy, our gradient labels only reflect the true values of $\frac{\delta E_\text{target}}{\delta \mathbf{p}_\mu}$ (E_target being the contribution to the total energy including T_S which we try to learn, e.g. T_S or $T_\text{S} + E_\text{xc}$) up to a scalar multiple of the vector of basis integrals $\mathbf{w}$. Hence, to train a model on $\boldsymbol{\nabla}_\mathbf{p} E_\text{target}$, we should not require it to exactly match our gradient labels $\mathbf{g}$. Rather, we should allow an offset in the direction of $\mathbf{w}$. Below, we will demonstrate how this can be incorporated in the loss function using a projection.

Secondly, orbital-free density optimization (equation 2.44) is a constrained problem: The total electron number is given, and we are only searching the space of

[1] They specify $\mathbf{M} = \mathbf{U}\mathbf{\Lambda}^{-1/2}$ in their paper, but do use the symmetric variant we describe here in their code.

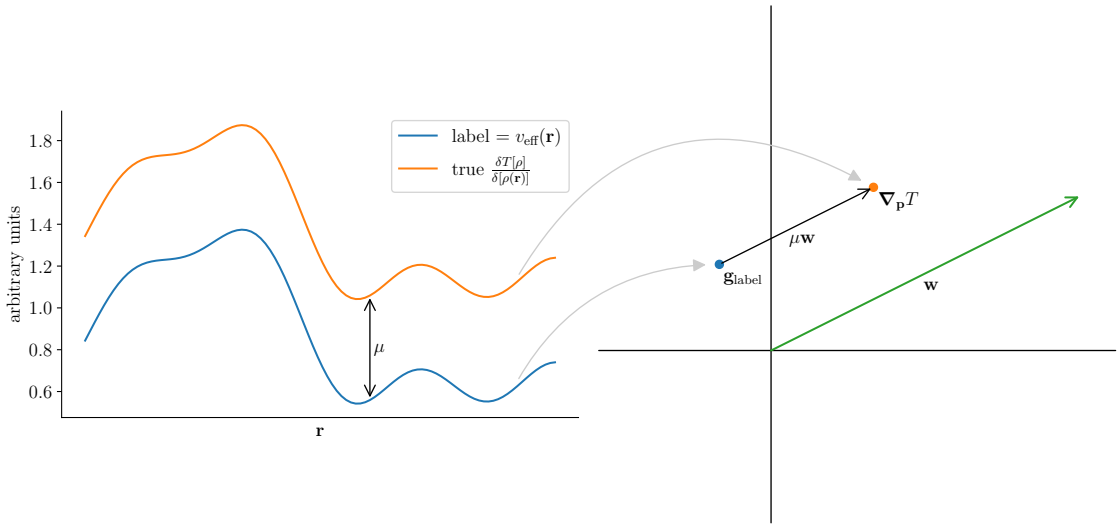


Figure 5.2. The need for gradient projection. As we only know the kinetic potential up to a constant offset μ (left, illustrated in 1D), our gradient labels $\mathbf{g}$ only agree with the true gradients $\boldsymbol{\nabla}_{\mathbf{p}} T$ up to an offset of $\mu\mathbf{w}$ (right, illustrated for two coefficient components).

densities that match it. One approach is using a constrained optimization algorithm, such as SLSQP [94] as we chose to do in KineticNet, see section 4.4. However, in the LCAB Ansatz for the density we have another, more direct, option: As the electron-number constraint is linear in the coefficients $\mathbf{p}$ (explicitly: $\mathbf{w}^\top \mathbf{p} = N_{\text{e}}$), we can use an iterative optimization procedure where we linearly project the update steps in coefficient space such that they leave the electron number invariant. For gradient-descent, this amounts to projecting the gradients before taking the step, hence we find the second motivation to introduce a gradient projection. This approach was first used in the context of OF-DFT by M-OFDFT [9]. Below, we will derive the advantageous properties of this projection, and expand on how it interacts with basis transformations in section 5.3.1.

Let us make the gradient projection explicit: We project along $\mathbf{w}$ onto its orthogonal complement $\langle \mathbf{w} \rangle^\perp$ with the projection matrix

$$\mathbf{P}_{\mathbf{w}} = \mathbb{1} - \frac{\mathbf{w}\mathbf{w}^\top}{\mathbf{w}^\top \mathbf{w}} \,. \tag{5.9}$$

It is easy to check that indeed,

$$\mathbf{P}_{\mathbf{w}} \mathbf{w} = \mathbf{w} - \frac{\mathbf{w}\mathbf{w}^\top \mathbf{w}}{\mathbf{w}^\top \mathbf{w}} = \mathbf{w} - \mathbf{w} = 0 \quad \Rightarrow \quad \mathbf{P}_{\mathbf{w}} \langle \mathbf{w} \rangle = 0 \,, \tag{5.10}$$

$$\forall \mathbf{a} \in \langle \mathbf{w} \rangle^\perp : \quad \mathbf{P}_{\mathbf{w}} \mathbf{a} = \mathbf{a} - \frac{\mathbf{w}\mathbf{w}^\top \mathbf{a}}{\mathbf{w}^\top \mathbf{w}} = \mathbf{a} \quad \Rightarrow \quad \operatorname{Im} \mathbf{P}_{\mathbf{w}} = \langle \mathbf{w} \rangle^\perp \,. \tag{5.11}$$

Let us now demonstrate how this projection solves the two problems mentioned above.

Firstly, in the loss function, one can use $\mathbf{P}_\mathbf{w}(\hat{\mathbf{g}} - \mathbf{g}) = \mathbf{P}_\mathbf{w}(\hat{\mathbf{g}} - \mathbf{g} + \mu\mathbf{w})$ to compare (variationally) predicted gradients $\hat{\mathbf{g}}$ to label gradients $\mathbf{g}$ only up to a scalar multiple of $\mu\mathbf{w}$:

$$\mathcal{L}_\text{gradient} = ||\mathbf{P}_\mathbf{w}(\hat{\mathbf{g}} - \mathbf{g})||_1 \tag{5.12}$$

$$= ||\mathbf{P}_\mathbf{w}(\hat{\mathbf{g}} - \mathbf{g} + \mu\mathbf{w})||_1 \,. \tag{5.13}$$

Secondly, in density optimization, the projection can be used exactly as described above: For an iterative optimization procedure, we modify the update rule to first project the step $\Delta\mathbf{p}_t$:

$$\mathbf{p}^{(t+1)} = \mathbf{p}^{(t)} + \Delta\mathbf{p}^{(t)} \quad \mapsto \quad \mathbf{p}^{(t+1)} = \mathbf{p}^{(t)} + \mathbf{P}_\mathbf{w}\Delta\mathbf{p}^{(t)} \,. \tag{5.14}$$

As the projection is onto the orthogonal complement of $\mathbf{w}$, updates of this form do not change the electron number:

$$\mathbf{w}^\top\mathbf{p}^{(t+1)} = \mathbf{w}^\top\mathbf{p}^{(t)} + \mathbf{w}^\top(\mathbf{P}_\mathbf{w}\Delta\mathbf{p}^{(t)}) = \mathbf{w}^\top\mathbf{p}^{(t)} \,. \tag{5.15}$$

Hence, if the initial guess is normalized correctly, $\mathbf{w}^\top\mathbf{p}^{(0)} = N_\text{e}$, it follows by induction that the final density with coefficients $\mathbf{p}^{(T)}$ is as well. A conceptual illustration of using the projection in density optimization can be seen in figure 5.3.

5.3.1 Interplay of basis transformation and projection

We have detailed in section 5.2 that it can be useful to linearly transform the basis functions for canonicalization via local frames (see section 3.2.2) or natural reparametrization (section 5.2.1). How should we project gradients in this new basis? One approach is to follow the chain of thought presented above with the transformed basis functions ω'_μ, leading to transformed basis integrals $\mathbf{w}'$ and an associated projection matrix $\mathbf{P}_{\mathbf{w}'}$. Alternatively, one could argue that the original projection matrix $\mathbf{P}_\mathbf{w}$ should simply be transformed into the new basis, i.e that

$$(\mathbf{P}_\mathbf{w})' \coloneqq \mathbf{A}^{-\top}\mathbf{P}_\mathbf{w}\mathbf{A}^\top = \mathbb{1} - \frac{\mathbf{A}^{-\top}\mathbf{w}\mathbf{w}^\top\mathbf{A}^\top}{\mathbf{w}^\top\mathbf{w}} \tag{5.16}$$

should be used[2].

It turns out that, in general, $\mathbf{P}_{\mathbf{w}'}$ and $(\mathbf{P}_\mathbf{w})'$ are not equal. This discrepancy arises because $\mathbf{P}_{\mathbf{w}'}$ is the unique projection along $\mathbf{w}'$ onto its orthogonal complement. In contrast, $(\mathbf{P}_\mathbf{w})'$ also projects points along the vector $\mathbf{w}'$, but onto the subspace $\langle\mathbf{A}\mathbf{w}\rangle^\perp$,

[2] $\mathbf{A}^\top$ is multiplied on the right to first transform gradients from the new to the original basis, where the coefficients are projected, and then transformed back into the new basis with $\mathbf{A}^{-\top}$.

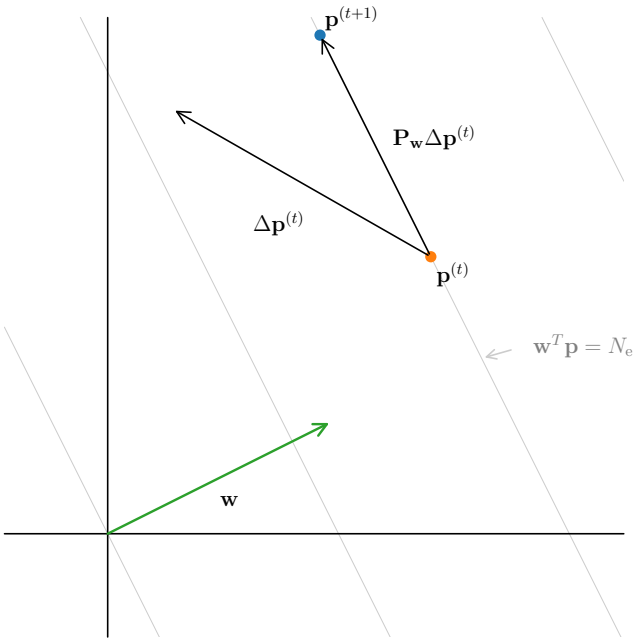


Figure 5.3. Preserving electron number in density optimization. By projecting the offset vector $\Delta\mathbf{p}_t$ to be orthogonal to the weight vector $\mathbf{w}$, the electron number stays invariant in density optimization.

which differs from $\langle\mathbf{w}'\rangle^\perp$ for any non-orthogonal basis transformation $\mathbf{A}$ (see figure 5.4):

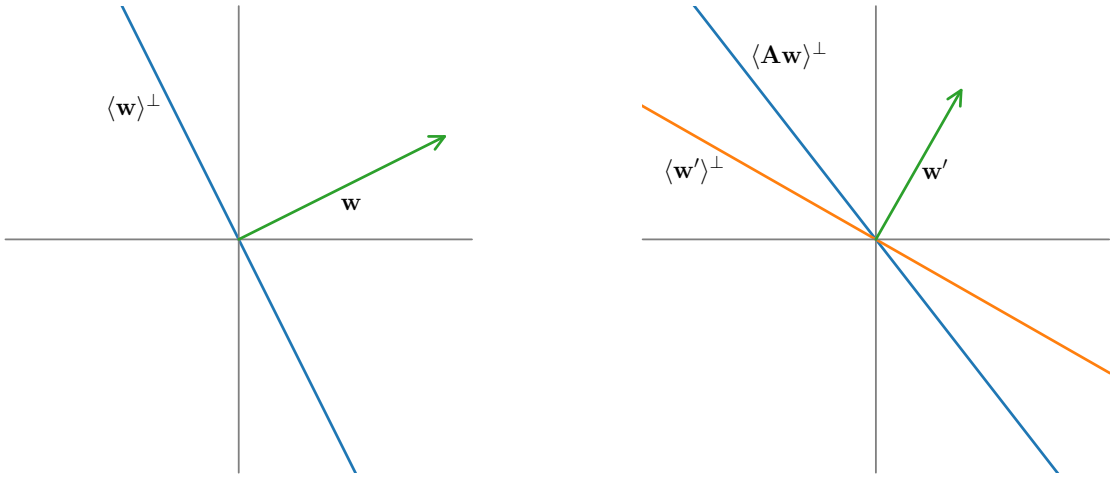
$$(\mathbf{P_w})'\mathbf{w}' = (\mathbf{A}^{-\top}\mathbf{P_w}\mathbf{A}^\top)(\mathbf{A}^{-\top}\mathbf{w}) = \mathbf{A}^{-\top}\underbrace{\mathbf{P_w}\mathbf{w}}_{=0} = 0\,, \tag{5.17}$$

and

$$\forall\mathbf{a} \in \langle\mathbf{A}\mathbf{w}\rangle^\perp : \quad (\mathbf{P_w})'\mathbf{a} = \mathbf{a} - \frac{\mathbf{A}^{-\top}\mathbf{w}\mathbf{w}^\top\mathbf{A}^\top\mathbf{a}}{\mathbf{w}^\top\mathbf{w}} = \mathbf{a} - \frac{\mathbf{A}^{-\top}\mathbf{w}(\mathbf{A}\mathbf{w})^\top\mathbf{a}}{\mathbf{w}^\top\mathbf{w}} = \mathbf{a} \tag{5.18}$$

$$\Rightarrow \quad \mathrm{Im}(\mathbf{P_w})' = \langle\mathbf{A}\mathbf{w}\rangle^\perp\,. \tag{5.19}$$

Thus, while either projection could be used in the loss function (both map the true $\boldsymbol{\nabla}_\mathbf{p}E$ and the label $\mathbf{g}$ to the same point), only the first version, $\mathbf{P_{w'}}$, is appropriate to use in density optimization in the transformed basis to preserve the correct electron number. Since accurate density optimization is what we aim for, we also use this version in the loss function, as this aligns the loss with the final goal as closely as possible. To the best of our knowledge, this is the same choice as in [9], but we are not certain as they do not discuss this interaction of basis transformation and gradient projection.



(a) **(b)**

Figure 5.4. Orthogonal projection and basis transformation. (a) Orthogonal projection in the original basis with $\mathbf{P}_\mathbf{w}$ is along $\mathbf{w}$ onto $\langle \mathbf{w} \rangle^\perp$ (blue line). (b) After a non-orthogonal basis-transformation, the image of the original projection (blue line) is no longer orthogonal to $\mathbf{w}$. Hence, $\mathbf{P}_{\mathbf{w}'}$ should be used to project onto $\langle \mathbf{w}' \rangle^\perp$ (orange line).

5.4 SAD guess

A multitude of established methods for generating initial guesses for KS-DFT exist, such as the MINAO initialization [18, 19], which is used in M-OFDFT [9]. However, while it is cheap compared to the Kohn-Sham iterations because of the minimal basis that it utilizes, it is unaffordable if one aims for linear scaling of the whole method, as it scales cubically with system size. Furthermore, it is only implemented in the KS basis, and hence density-fitting is required to express the guess in the LCAB Ansatz. This step becomes expensive for larger systems as well. Therefore, the need arises for a cheaper alternative. One simple, linearly scaling, but still accurate option is a Superposition of Atomic Densities (SAD). The idea is to compute spherically symmetric densities for each atom type, and then superpose them to get the total density. There are established procedures for this in the KS case, e.g. spherically averaged Hartree-Fock. If density-fitting is done once for each atom-type, the guess can be generated in linear time.

However, here we choose another way to generate the atomic densities, enabled by the fact that we have datasets with ground state LCAB density coefficients at hand, as they are required for training: We take all instances of each atom type (i.e. chemical element) in the dataset, and take the average of the corresponding coefficients over

all these instances. For any certain molecule $\mathcal{M}$, the SAD $\bar{\mathbf{p}}$ is then constructed by concatenating these averages for all atom types in the molecule. However, this superposition of atomic densities is not necessarily normalized to the correct electron number. Since the electron number stays invariant during density optimization, we need to normalize the guess. We will discuss two methods for this in the following sections.

Normalization by simple scaling

One approach is to scale the coefficients linearly to the correct electron number N_e, leading to

$$\mathbf{p}_\mathrm{scaled} = \bar{\mathbf{p}} \, \frac{N_\mathrm{e}}{\mathbf{w}^\top \bar{\mathbf{p}}} \, . \tag{5.20}$$

While simplicity is certainly one advantage of this method, it has a major shortcoming: The largest part of the electron density lies close to the cores, and this core density varies only very little between different instances of the same atom type between molecules. Thus, the SAD guess describes it very precisely. The coefficients of the inner $l = 0$ basis functions largely describe this core density and should hence be varied very little in the normalization. However, scaling all coefficients by the same factor to achieve normalization does not respect this. For example, the core density of atomic species with high electronegativity (whose corresponding coefficients, on average, describe a higher number of electrons than their atomic number indicates), would be scaled down and hence underestimated. This motivates the need for a more sophisticated normalization procedure.

Variance adapted normalization

To alleviate the problem described above, we propose a normalization procedure that takes into account the variance of the coefficients over the dataset: Coefficients with high variance should be scaled more than those with low variance, as they are more likely to be far from the mean. In order to quantify this likelihood, we assume that the coefficients are normally distributed over the dataset. As the mean $\bar{\mathbf{p}}$ and variance σ_μ of each coefficient are known, we can formulate this as an optimization problem, where we aim to find the coefficients $\mathbf{p}_\mathrm{normalized}$ that maximize the Gaussian likelihood of the coefficients over the dataset, given a constraint on the electron number (see also figure 5.5):

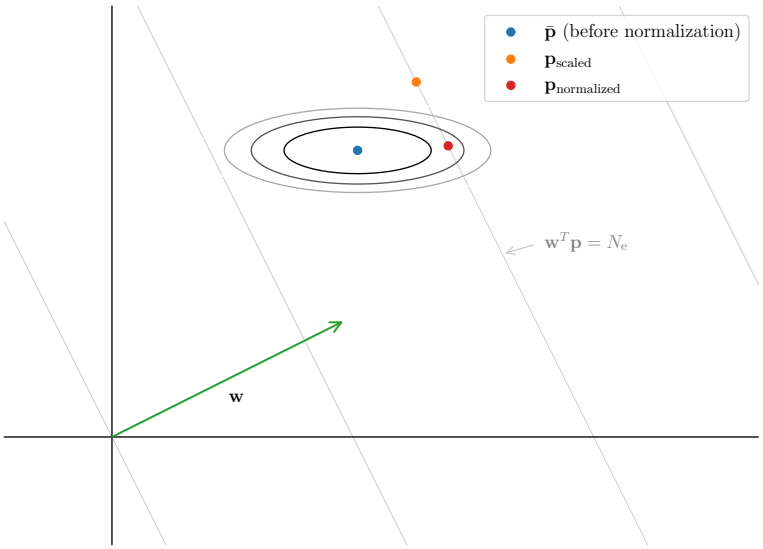


Figure 5.5. Different methods for normalizing the initial guess. Simply scaling the atomic initial guess leads to $\mathbf{p}_{\text{scaled}}$, which can be quite different to the most likely normalized guess, $\mathbf{p}_{\text{normalized}}$, in accordance with a Gaussian distribution deduced from the dataset statistics (grey ellipses).

$$\mathbf{p}_{\text{normalized}} = \underset{\mathbf{p},\, \mathbf{w}^\top\mathbf{p}=N_e}{\arg\max}\, P(\mathbf{p}) = \underset{\mathbf{p},\, \mathbf{w}^\top\mathbf{p}=N_e}{\arg\max} \sum_\mu \frac{(p_\mu - \bar{p}_\mu)^2}{2\sigma_\mu^2} \tag{5.21}$$

$$= \bar{\mathbf{p}} + \underset{\mathbf{d},\, \mathbf{w}^\top\mathbf{p}=\Delta N_e}{\arg\max} \sum_\mu \frac{d_\mu^2}{2\sigma_\mu^2} \tag{5.22}$$

with $\Delta N_e = N_e - \mathbf{w}^\top\bar{\mathbf{p}}$, the difference between the desired electron number, and that corresponding to the mean coefficients. Introducing a Lagrange-multiplier λ, we get:

$$\mathcal{L}(\mathbf{d}, \lambda) = \sum_\mu \frac{d_\mu^2}{2\sigma_\mu^2} + \lambda\left(\left(\sum_\mu w_\mu d_\mu\right) - \Delta N_e\right), \tag{5.23}$$

and can solve first for d:

$$\frac{\partial\mathcal{L}}{\partial d_\mu} = \frac{d_\mu}{\sigma_\mu^2} + \lambda w_\mu \overset{!}{=} 0 \tag{5.24}$$

$$\Rightarrow \quad d_\mu = -\lambda\sigma_\mu^2 w_\mu\,, \tag{5.25}$$

and then for λ:

$$\frac{\partial\mathcal{L}}{\partial\lambda} = \left(\sum_\mu w_\mu d_\mu\right) - \Delta N_e \overset{(5.25)}{=} -\lambda\sum_\mu \sigma_\mu^2 w_\mu^2 - \Delta N_e \overset{!}{=} 0 \tag{5.26}$$

$$\Rightarrow \quad \lambda = -\frac{\Delta N_e}{\sum_\mu \sigma_\mu^2 w_\mu^2}\,. \tag{5.27}$$

Finally, we plug the result for λ into the one for d, equation (5.25), and that into equation (5.22) to finally find

$$(\mathbf{p}_{\text{normalized}})_\mu = \bar{p}_\mu + \Delta N_\text{e} \frac{\sigma_\mu^2 w_\mu}{\sum_\mu \sigma_\nu^2 w_\nu^2} \,. \tag{5.28}$$

This result matches the intuition established above: The correction to each component of the average coefficients $\bar{\mathbf{p}}$ is proportional both to the variance of it over the dataset, and the weight of its corresponding basis function. An illustration of this method compared to simply scaling the guess is shown in Figure (5.5).

One could either apply this normalization for the whole molecule at once, or for each atom separately. While the former may be slightly more precise, since it is more flexible, the latter is more intuitive and easier to implement. Hence, we normalize the electron number per atom and before applying any basis transformations such as natural reparametrization. This makes the guess a superposition of atomic densities in the truest sense, even after the normalization: The density coefficients placed at each atom only depend on its type, not on all atom types present in the molecule, as would be the case were one to choose to normalize the whole molecule at once.

We evaluate the accuraccy of this SAD guess and compare it to the MINAO guess in section 5.8.1.

5.5 External potential perturbation

We aim to solve the convergence issues of the M-OFDFT method [9] by extending the idea of perturbing the external potential to generate training data we pioneered in chapter 4: We propose a similar method which not only generalizes our approach to the LCAB Ansatz but also improves it by generating an independently perturbed data point for each step of the KS-DFT procedure.

This is achieved by sampling a perturbation $\delta V^{(n)}(\mathbf{r})$ in each SCF iteration n, and adding it to the effective potential:

$$V_\text{eff}^{(n)}(\mathbf{r}) \mapsto V_\text{eff}^{(n)}(\mathbf{r}) + \delta V^{(n)}(\mathbf{r}) \,. \tag{5.29}$$

This perturbation is sampled via coefficients $\delta V_\mu^{(n)}$ in the density basis:

$$\delta V^{(n)}(\mathbf{r}) = \sum_{\mu=1}^{B} \delta V_\mu^{(n)} \omega_\mu(\mathbf{r}). \tag{5.30}$$

To incorporate this perturbation into the SCF iteration, we calculate its influence on

the Fock matrix:

$$\mathbf{F}^{(n)}_{\alpha\beta} \mapsto \mathbf{F}^{(n)}_{\alpha\beta} + \left\langle \eta_\alpha \left| \delta V^{(n)} \right| \eta_\beta \right\rangle = \mathbf{F}^{(n)}_{\alpha\beta} + \int \delta V^{(n)}_\mu \omega_\mu(\mathbf{r}) \eta_\alpha(\mathbf{r}) \eta_\beta(\mathbf{r}) \mathrm{d}\mathbf{r} \tag{5.31}$$

$$= \mathbf{F}^{(n)}_{\alpha\beta} + \delta V^{(n)}_\mu W_{\mu\alpha\beta}\,, \tag{5.32}$$

with the three-function overlap between density basis and Kohn-Sham basis functions $W_{\mu\alpha\beta} = \int \omega_\mu(\mathbf{r}) \eta_\alpha(\mathbf{r}) \eta_\beta(\mathbf{r}) \mathrm{d}\mathbf{r}$.

As the external potential enters the calculation of our gradient labels (eq. 5.3), we have to adjust the gradient labels accordingly:

$$\frac{\delta T_\mathrm{S}\left[\rho_{[\mathbf{p}^{(n)}]}\right]}{\delta \rho} = -V^{(n)}_{\text{eff, unperturbed}} - \delta V^{(n)} + \mu^{(n)}. \tag{5.33}$$

In components, this implies

$$\frac{\partial T_\mathrm{S}[\rho_{[\mathbf{p}^{(n)}]}]}{\partial p_\mu} = \underbrace{\int \left(\mu^{(n)} - V^{(n)}_\text{eff}(\mathbf{r})\right) \omega_\mu(\mathbf{r})\, \mathrm{d}\mathbf{r}}_{\text{gradient without perturbation}} - \int \delta V^{(n)}(\mathbf{r}) \omega_\mu(\mathbf{r}) \mathrm{d}\mathbf{r} \tag{5.34}$$

and we can compute the adjustment to the gradient labels due to the perturbation as

$$\delta v^{(n)}_\mu \coloneqq \int \delta V^{(n)}(\mathbf{r}) \omega_\mu(\mathbf{r}) \mathrm{d}\mathbf{r} = \delta V^{(n)}_\nu \int \omega_\mu \omega_\nu \mathrm{d}\mathbf{r} = \delta V^{(n)}_\nu W_{\mu\nu}, \tag{5.35}$$

with the overlap matrix of basis functions in the LCAB ansatz $W_{\mu\nu}$. One subtlety of this procedure is its interaction with DIIS (see section 2.3.2): We choose to sample the perturbation and make the according adjustments after the DIIS extrapolation and caching of the Fock matrices, such that the unperturbed Fock matrices are used in the DIIS procedure.

5.5.1 Details of sampling $\delta V^{(n)}$

We sample the coefficients $\delta V^{(n)}_\mu$ from a centered normal distribution. To generate perturbations of varying strength, we adjust the standard deviation of the distribution according to the SCF iteration number n: In the first five iterations, no perturbation is added. Perturbations start at iteration 6 with a standard devation of 0.102, which is linearly decayed to 0.002 until iteration 26. In the final iterations until convergence, again no perturbation is added. This delayed start of perturbations is motivated by the fact that if the perturbation is applied immediately, densities very far from the ground state are generated which are likely not useful for training the model. Also, it oftentimes leads to non-convergent SCF procedures. Some example densities with and without perturbation from the dataset are shown in figure 5.6.

Figure 5.7 shows distributions of the difference between non-interacting kinetic energy labels from samples, relative to the corrsponding ground state labels, for the

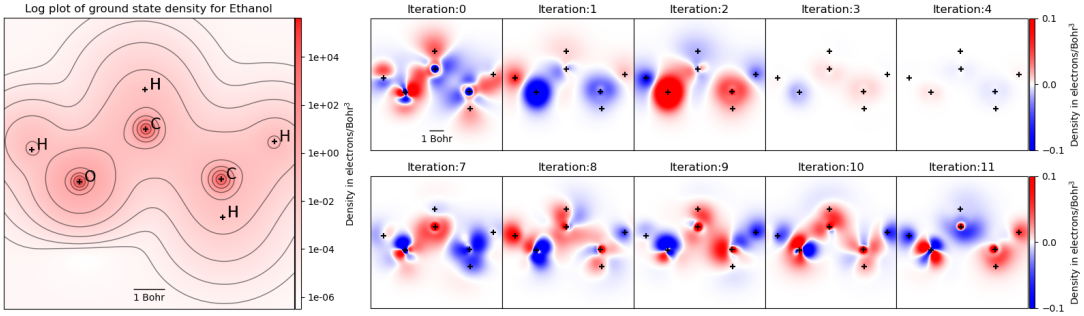


Figure 5.6. Slices of densities. Left: A slice through the ground state density for an ethanol molecule. Right: The same slice for samples from our dataset. Plots in the grid show differences between the ground state density at ten SCF iterations. For iterations zero to four, shown in the top row, no perturbation is applied yet, whereas to generate the densities of SCF iterations seven to eleven, the external potential is perturbed in every iteration. Figure created by master student Tim Ebert.

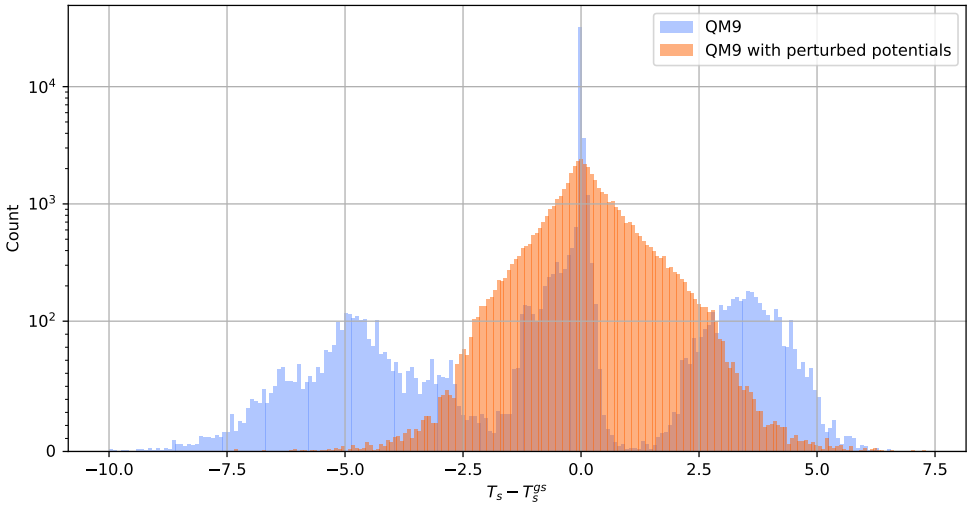


Figure 5.7. Perturbed vs. unperturbed energy difference histogram. The histogram shows distributions of the differences between label kinetic energies from all SCF iterations and the corrsponding ground state, for the perturbed and unperturbed QM9 datasets. While the distribution without perturbation has gaps with almost no data, the perturbed data is more evenly distributed around the ground state. Figure created by master student Manuel Klockow.

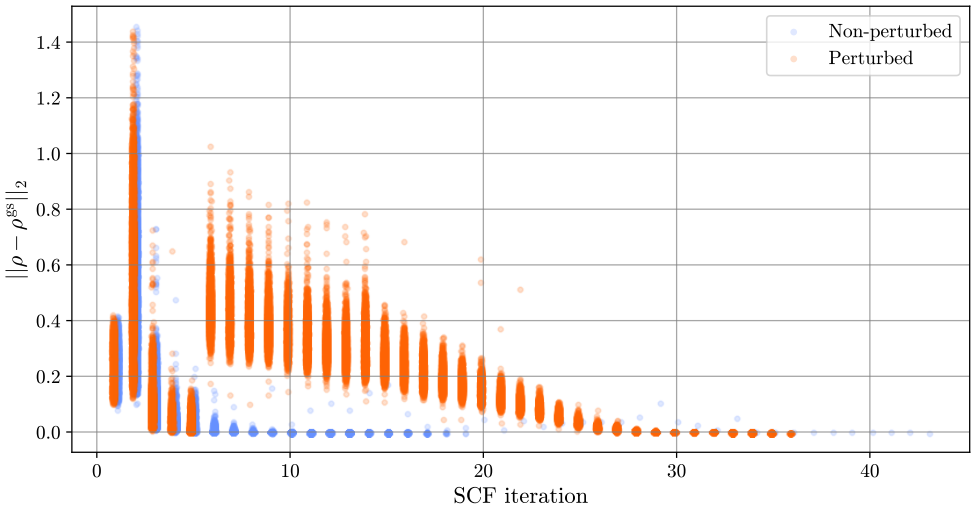


Figure 5.8. Distances of perturbed and unperturbed samples to ground state densities. For all molecules of the QM9 validation set, we plot the L2 distance to their ground state density for each SCF iteration, both for the unperturbed data and our perturbed datasets. Without perturbations (blue), the KS procedure usually converges by iteration 20, with few outliers. For the perturbed data (red), the magnitude of the perturbation of the external potential is reflected in the plot: the L2 norm jumps up when the first and largest perturbation is applied in iteration 6, and then decreases roughly linearly until iteration 26, when the final and smallest perturbation is applied. Most samples then converge in about 10 iterations. Figure created by master student Manuel Klockow.

perturbed and unperturbed QM9 dataset.They indicate that the perturbed data is more evenly distributed around the ground state, without the obvious gaps that can be seen in the histogram of the unperturbed data.

Figure 5.8 shows the influence of the perturbation schedule on the L2 norm of the difference between the sample coefficients and the ground state coefficients, demonstrating a clear correlation between the perturbation strength and the resulting distance to the ground state.

Finally, figure 5.9 shows that the gradients of the perturbed data points are more aligned with the direction to the ground state, as measured by the cosine similarity between the total gradient labels and the vector pointing from the ground state to the coefficients. This is a curious observation, for which we have no clear explanation. We will revisit it in the discussion, as this might play a role in the success of our method.

We note that a multitude of different perturbation schemes are conceivable, and the one we chose is only one of many possible choices. For instance, sampling pertur-

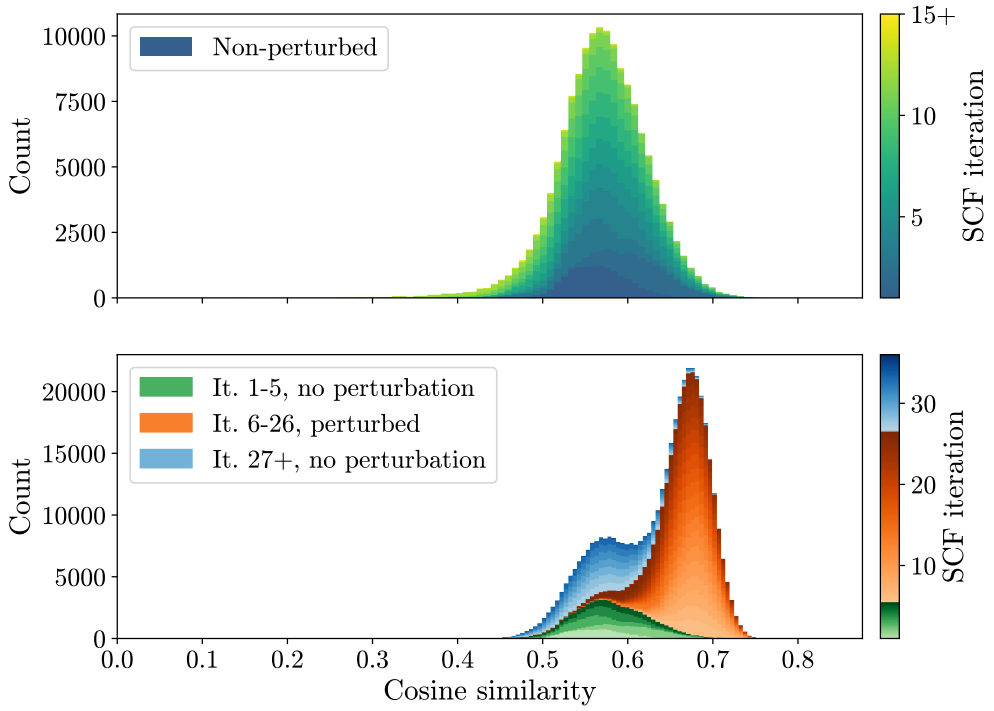


Figure 5.9. Gradients of perturbed data point are more aligned with direction to ground state. Histograms of the cosine similarities between the total gradient labels $\nabla_{\mathbf{p}} E_{\text{tot}}$ and the vector pointing from the ground state $\mathbf{p}^*$ to the coefficients $\mathbf{p}$ for QM9 validation data, with the unperturbed data generated following [9] (top), and our dataset generated with perturbed V_{eff} (bottom). Cosine similarities are computed in the orthogonalized natural reparametrization basis. Histograms are coloured by SCF iteration. For the perturbed data set, perturbations are applied at iterations 6 to 26, and the resulting samples exhibit a higher similarity than the unperturbed samples.

bations in the natural reparametrization basis might yield even better results, as the orthogonality of basis functions in this basis might lead to more evenly distributed labels around the ground state. Another conceivably useful and more physical approach would be to place point charges with randomized positions and strengths, and calculate the perturbed potential from these charges.

A more detailed description of external potential sampling for varied data generation, including consistency checks to verify the correctness of generated labels, is given in the master thesis of Manuel Klockow.

5.6 Architectural improvements

5.6.1 Graphormer

Zhang et al. [9] adapt the Graphormer architecture [61], a transformer-like architecture for graph data, for the task of learning energy functionals and achieve rotational invariance via local frames (see section 3.2.2)

We follow their lead and use the same architecture, but find that some simple modifications improve its performance. Firstly, in preliminary experiments we have found the original Graphormer architecture to be prone to overfitting. This problem is alleviated by reducing the number of G3D layers (the self-attention blocks making up the main part of the model) quite drastically from 12 to just 4, without a significant drop in performance. Hence, we use this reduced number of layers in all our experiments. Secondly, after reducing the number of layers, disabling all dropout layers in the model further improved the results, which is why we do not use dropout in our model.

Learned initial guess

In [9], the Graphormer is equipped with an additional output head to predict the difference between the input density coefficients and the ground state coefficients $\mathbf{p}^*$. This is then used in density optimization to "project" the MINAO guess to an improved, learned initial guess which is already very close to the ground state. We have found that this learned guess can be further improved by only training the respective head on the MINAO samples on which it is actually used, and, if one really wants to get the best guess possible, by training a model exclusively for this task. The potential benefits of multitask learning do not seem to be realized in this case. Once we realized that our method achieves proper convergence, we dropped the initial guess head from the model compeletely, as we achieve identical results with a classical SAD guess (see section 5.4).

Conceptual disadvantages

The Graphormer is a powerful architecture which has shown to be capable of learning accurate energy functionals for OF-DFT. However, it is not without its shortcomings: On one hand, achieving invariance via local frames comes with the cost of accepting discontinuities at configurations where the nearest-neighbor atoms change. While unobtrusive for density optimization of single geometries, this could become a serious problem for geometry optimization, where geometries are updated in small steps, and

flips of local frames will occur. Furthermore, the information which is passed between atoms consists of only invariant quantities, and also the relative orientation of local frames does not enter the model. The Graphormer architecture could be adapted facilitating the method introduced in [68] to allow the passing of geometric features between frames. However, we here attempt to mitigate both problems at once: We have adapted the EquiformerV2 [62] architecture to our task, as we describe in the next section.

5.6.2 Equiformer

The EquiformerV2 architecture [62], in the following simply "Equiformer", is an equivariant neural network for graph data where geometric features (equivariant tensor fields) are communicated between neighboring nodes via a local attention mechanism. Like the Graphormer, it has been shown to perform well on a variety of equivariant graph-learning tasks, so we set out to adapt it to the task of learning energy functionals for OF-DFT. Manuel Klockow has implemented the required changes to the Equiformer architecture which we describe in the following.

Firstly, we of course drop the local-frame mechanism, as the Equiformer is equipped to deal with features transforming under irreducible representations of the rotation group, just as the density coefficients do.

Node embedding

In the original Equiformer, the input to the model consists of the molecular structure $\mathcal{M}$ alone, while in our case we also pass the density coefficients $\mathbf{p}$. Hence, a first layer processing these additional features and joining them with the features encoding the molecular structure has to be added, mirroring the Node Embedding module in the Graphormer. This layer is shown in figure 5.10. We apply the atom-hot encoding as in the Graphormer, and then pass the density coefficients through a linear layer which only mixes tensor fields of the same order. The resulting features are added to the standard Equiformer embedding, which encodes the molecular structure.

Feature-wise rescaling

Unlike for the Graphormer, we do not use local frames for canonicalization. Thus, the input features are no longer invariant under rotations, and special care to preserve their transformation behaviour has to be taken in all layers that modify the input features, in particular the three enhancement modules introduced in [9]. While the natural reparametrization of the density coefficients (see section 5.2.1) does not have

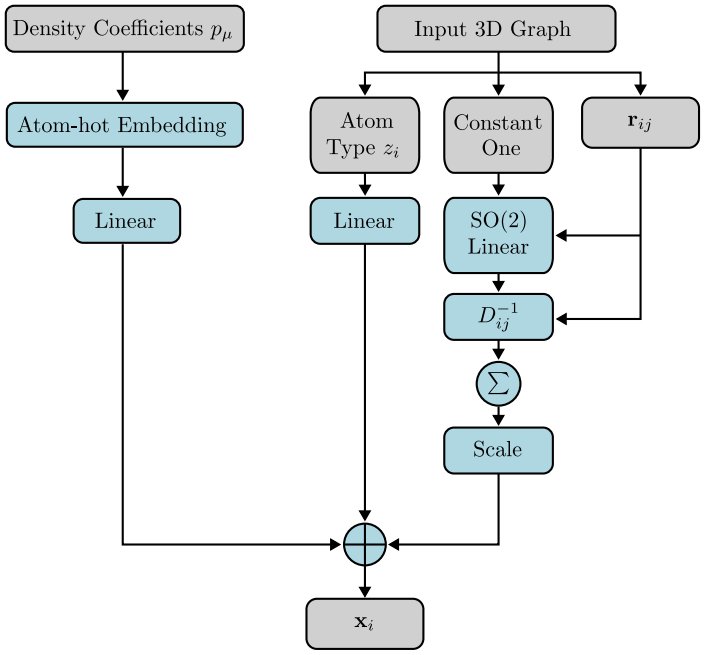


Figure 5.10. Adapted embedding module for the EquifomerV2 model. After the initial atom-hot encoding, a linear layer, which only mixes tensor fields of the same order, transforms the density coefficients to match the shape of the standard Equiformer embedding. This encodes the molecular structure $\mathcal{M}$, consisting of atom posistions and types. The sum of both, $\mathbf{x}_i$, is the input to the subsequent first Equiformer block (not shown). Figure created by master student Manuel Klockow.

to be adapted as it already transforms the features in an equivariant manner, the two other modules have to be adjusted.

The dimension-wise rescaling module as used in the Graphormer shifts and scales each coefficient for each atom-type independently. This is in general not equivariant, hence we introduce a similar *feature-wise rescaling* module as a replacement, which scales all coefficients belonging to the same tensorial feature jointly. Furthermore, the shift operation is only applied to scalar features, as it would otherwise break equivariance as well. The parameters of the feature-wise rescaling module are determined almost identically as for the dimension-wise rescaling module in the Graphormer, with the only difference being that the standard deviations and maximal gradients are calculated over all atoms and coeffiecients belonging to a certain tensorial feature.

Atomic reference module

The fact that the atomic reference module as introduced in [9] is generally not equivariant can be seen by considering the predicted gradients: The atomic reference module effectively adds the gradient $\bar{\mathbf{g}}_{\mathcal{M}}$, constructed from the per-coefficient means over the

training to the model gradients. This gradient does not change when the input features are rotated, and hence breaks the transformation behaviour of gradient prediction, if any non-scalar components of $\bar{\mathbf{g}}_{\mathcal{M}}$ are non-zero.

Hence, we make the adjustment of only considering the mean gradient for scalar features. In practice this is easily implemented by setting the average gradients $\bar{\mathbf{g}}^{l>0}$ to zero. This version with $l > 0$ features zeroed out is then also used in the computation of the per-atom and global biases for the atomic reference module.

As a side note, observe that if the orientations of the molecules in the training set is random, the expected mean gradient for $l > 0$ features is zero anyways, hence this adjustment does not diminish the effectiveness of the atomic reference module in the case of equivariant features.

5.7 Implementation details

Implementing the method was a joint effort: PhD students Marc Ickler and Tobias Kaczun as well as master students Tim Ebert, Christof Gehrig, Dominik Geng, Gerrit Gerhartz, Manuel Klockow and Johannes Schmidt collaborated with the author of this thesis in the implementation.

As [9] does not provide training code, we started by reimplementing their method. We wrapped the trained model provided by them to work with our evaluation code, in order to be able to compare our results to theirs on metrics not provided in their article. In the results section, we have marked results obtained with the model trained by them, but evaluated with our implementation of density optimization as such. All other scores and data used for figures are obtained either straight from the paper or their model and their evaluation code.

We implemented the machine-learning part in the PyTorch framework, using the PyTorch Geometric library [110] for graph neural networks and the PyTorch Lightning library [111] for training. Data generation was done using the pyscf library [90, 91], on which we also based our code for Orbital-Free DFT.

For basis transformations such as local frames and natural reparametrization, we took a different approach than [9]: Instead of computing them on the fly inside the neural network (as we have deduced they do, inspecting their provided model), we instead precompute them and store them on disk. This is faster and decreases load during training, with the slight downside of increased disk space usage.

Table 5.1. Density errors of initial guesses. We compare the accuracy of the MINAO guess [18, 19] and our SAD (see section 5.4) on 1000 molecules from the QM9 dataset.

	$\frac{\lVert\rho_{\text{guess}}-\rho^*\rVert_2}{N_e}$		$\frac{\lVert\rho_{\text{guess}}-\rho^*\rVert_1}{N_e}$	
	mean [%]	std [%]	mean [%]	std [%]
MINAO	0.54	0.03	7.80	0.74
SAD	**0.40**	0.03	**7.34**	0.70

5.8 Results

In the following, we present the results of our experiments. We take the molecular geometries from the QM9 dataset [102, 103], consisting of 134 thousand small organic molecules with up to 9 heavy atoms, following [9].

For all experiments utilizing the Graphormer architecture, we configure it as [9] with the changes discussed in section 5.6.1. We utilize a cosine learning rate schedule [112], interpolating between an initial learning rate of 7×10^{-5} and a final one of zero. The AdamW optimizer [113] is used with a minor weight decay of 1×10^{-10} and a batch size of 128. The complete list of hyperparameters used for training the Equiformer models are given in the appendix, see table B.2.

5.8.1 SAD guess

We evaluate the accuracy of the SAD guess on 1000 molecules from the QM9 dataset, and compare it to the MINAO guess, results are shown in table 5.1. The SAD guess is slightly more accurate than the MINAO guess, both in terms of the L2 and L1 norm of the difference to the ground state density. More critically, the SAD guess can be computed cheaply even for large molecules. However, that the accuracy of the initial guess is not the most important factor for the success of our method. If it were, then a learned initial guess, as proposed in [9] and discussed in section 5.6.1, would be neccessary.

5.8.2 Tuning loss weights

As we found the ratio of loss weights between the L1 energy and gradient loss to be crucial for the performance of our models, we conducted a systematic hyperparameter search to find the optimal ratio. To this end, we have trained a Graphormer model on the QM9 dataset with varying ratios of loss weights, while keeping their sum fixed to 1, and compared the resulting models by evaluating the mean absolute errors (MAEs)

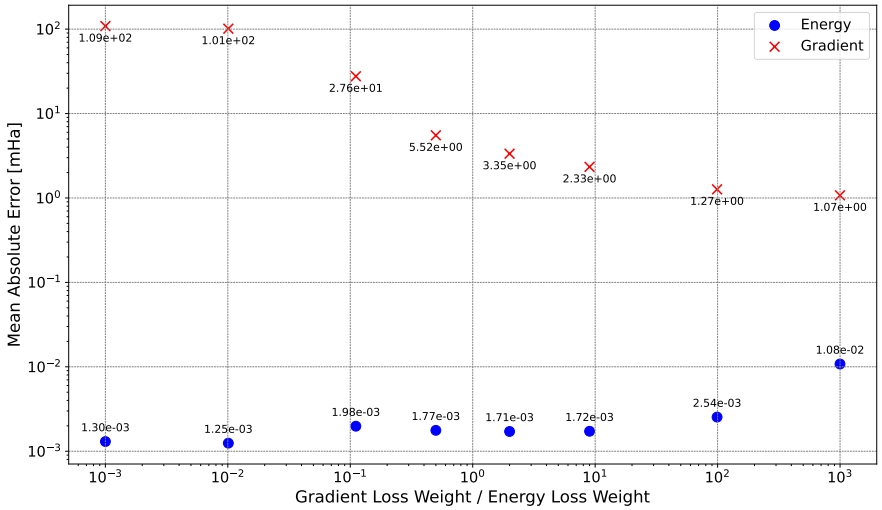


Figure 5.11. Loss weights tuning. Validation gradient- and energy mean absolute errors are plotted against the ratio of corresponding loss weights used during training. Expectedly, in most cases, increasing a contributions relative loss weight improves it. The energy loss however displays a plateau between ratios of 0.1 and 10, where increasing the gradient loss weight even leads to a slightly decreasing energy loss. Hence, we choose the last point on this plateau, a ratio of 10 to train our models.

of the energies and gradients on the validation set. The results are shown in figure 5.11. In general, increasing the loss weight of a contribution leads to a decrease in the corresponding error, with one curious exception: The energy loss plateaus between ratios of 0.1 and 10, where we even observe a slight decrease in MAE with increasing gradient loss weight. We choose the rightmost point on this plateau, corresponding to loss weights of 0.1 for the energy loss and 0.9 for the gradient loss for training our models.

One might argue that this tradeoff between energy and gradient loss should not be neccessary, as for the true energy functional both objectives would be perfectly fulfilled. However, the two tasks of precise energy and gradient prediction on some finite training set do not align perfectly, models that solve one but fail at the other are conceivable in both directions. Instead, the need for this tradeoff can be seen as an indication that our training procedure and model still fail to capture the true energy functional, possibly because of a lack of expressivity or a lack of data.

This hyperparameter search was conducted once, and the resulting loss weights kept for all subsequent experiments in this chapter. In particular, the Equiformer models were trained with these loss weights as well. The experiments for this hyperparameter search were conducted by master student Dominik Geng.

Table 5.2. Training data ablations. We compare the performance of Graphormer models trained on different datasets: The unperturbed QM9 dataset (as described in [9]), and three subsets containing different SCF iterations of our perturbed version (see section 5.5.1). Density optimization was performed on 500 randomly chosen but fixed molecules from the validation set. The mean absolute energy errors of different energies at the predicted vs the label ground state density are given in mHa, as well as the mean L2 density deviation and percentage of converged molecules is reported.

Training Dataset	$\lvert\Delta E_{\text{tot}}\rvert$	$\lvert\Delta E_H\rvert$	$\lvert\Delta(T_{\text{S}} + E_{\text{xc}})\rvert$	$\lvert\Delta E_{\text{ext}}\rvert$	$\lVert\Delta\rho\rVert_2$	conv. [%]
unperturbed	33.20	119.25	665.64	703.97	0.283	0
perturbed, all it.	1.60	29.29	20.73	41.56	0.0393	99
perturbed, it. $\geq$15	4.08	46.14	16.85	43.61	0.0454	**100**
perturbed, it. 5-26 + ground state.	**0.77**	**21.46**	**7.85**	**24.51**	**0.0225**	**100**

5.8.3 Perturbed training data

Having fixed the loss weights to 0.1 for the energy loss and 0.9 for the gradient loss, we now investigate the effect of the perturbed training data on the performance of our models.

We compare the performance of Graphormer models trained on different datasets: The unperturbed QM9 dataset (generated directly following [9]), and three subsets containing different SCF iterations of our perturbed version (see section 5.5.1):

1. **Perturbed, all iterations:** Contains all samples of the perturbed dataset, including the initial and final SCF iterations at which no perturbation is applied. This dataset is the most diverse, but also contains samples far from the ground state.

2. **Perturbed, iterations $\geq$ 15:** Contains the samples of SCF iteration 15 and higher, including the final iterations before convergence with no perturbations. This dataset is most concentrated around the ground state.

3. **Perturbed, iterations 6-26 + ground state:** Samples from the 5th to the 26th SCF iteration, and the ground state. This datasets includes all iterations where perturbations are applied, but excludes the first iterations which are typically very far from the ground state (see figure 5.8), and the last set of unperturbed SCF iterations to avoid oversampling of the ground state.

The number of training epochs is adjusted for each dataset to ensure that a similar number of training steps are performed for each dataset, resulting in 161 epochs for

the untransformed data, and 58, 90 and 99 epochs for the three perturbed datasets, respectively.

After training, the models are deployed in density optimization on a random subset of the QM9 validation set consisting of 500 molecules. Density optimization is performed with gradient descent with momentum ([114]) as implemented in `PyTorch` ([115]). Each density optimization run is terminated if the gradient norm either falls below a threshold of 1×10^{-4} or the maximum number of iterations, set to 10 000 is reached, which we classify as failure to converge.

The results are shown in table 5.2, listing the mean absolute energy errors of different contributions to the total energies, evaluated at the coefficients resulting from density-optimization vs. the label at the true ground state density, as well as the mean L2 density deviation and percentage of converged molecules. As expected, the model trained on the unperturbed data performs poorly, never converging in density optimization. There, the resulting metrics at the end of the optimization run are insatisfactory, with a mean absolute energy error of 33.20 mHa, and a mean L2 density deviation of 0.283. The model trained on the perturbed data set with iterations 6-26 and the ground state yields the best results in all metrics, with the lowest mean absolute energy error of 0.77 mHa, thereby achieving chemical accuracy. A more detailed view on these density optimziation runs is presented in figure 5.12: All density optimization on the 500 random molecules from the QM9 validation set using this model reliably converge within 1000 steps, consistent with the steady decrease of gradient norm over the course of the optimization run.

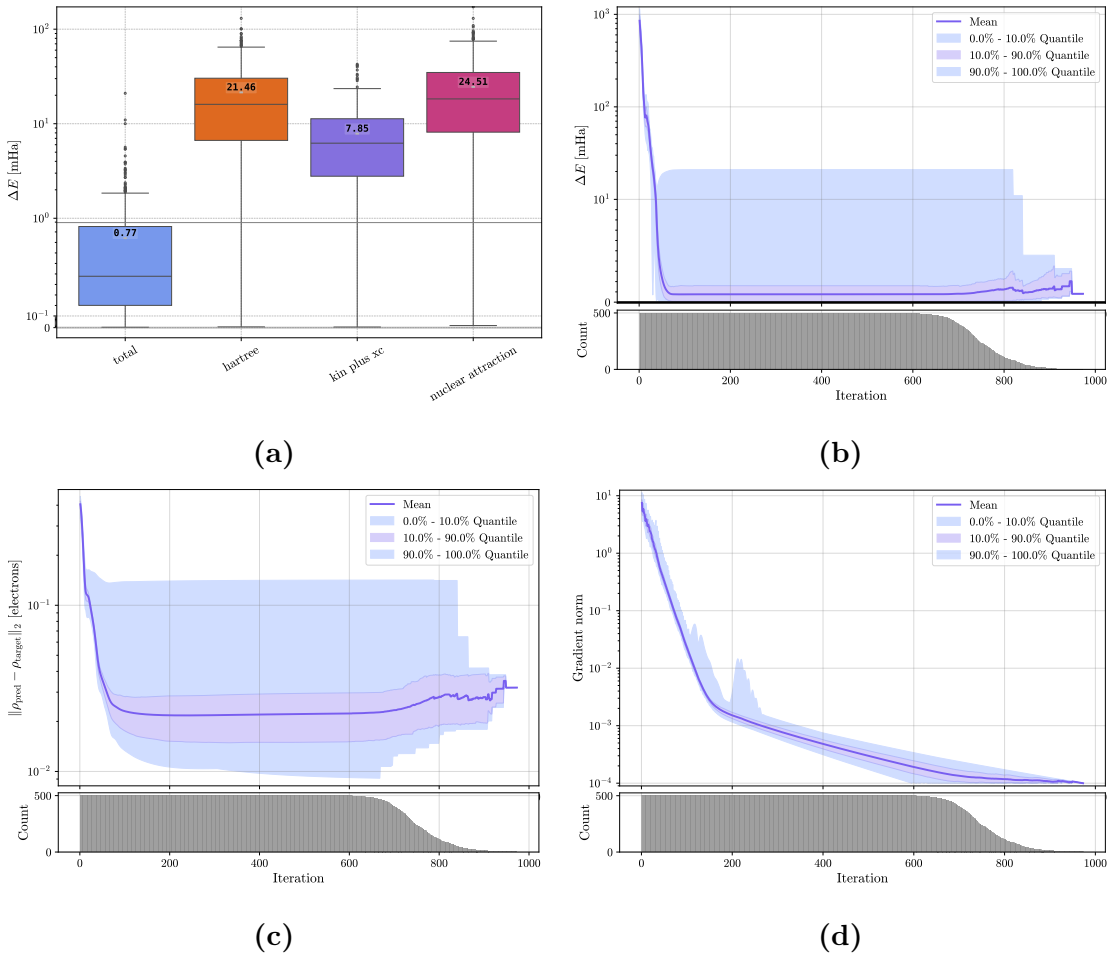


Figure 5.12. Density optimization on 500 QM9 Molecules. The model used here is the best-performing one shown in table 5.2, i.e. a Graphormer trained on SCF iterations 6 to 26 the perturbed data plus the ground states, on the target $(T_S + E_{xc})$. In the box-plot (a), absolute errors of the individual contributions to the energy are shown, medians are marked by vertical bars and means are annotated with numeric values. The summarized optimization trajectories of absolute energy error (b), L2 density error (c), and gradient norm (d) all rapidly decrease in the first 100 iterations. The gradient norm decreases steadily until the convergence threshold of 10^{-4} is reached between 600 and 1000 iterations. Already converged samples are excluded from the plots in later iterations, which leads to the visible jumps in means and quantiles. Figure created by master student Manuel Klockow.

5.8.4 Training target

We train density functionals with three different targets: The residual kinetic energy to its APBE approximation [47], $T_\mathrm{S} - T_{\mathrm{S,\,APBE}}$, the sum of the kinetic energy and exchange-correlation energy $T_\mathrm{S} + E_\mathrm{xc}$, and the total energy E_tot. While we have considered learning the kinetic energy alone as in chapter 4, this leads to suboptimal results in preliminary experiments, most likely to larger gradient scales, which is why we stopped pursuing this target. Learning the residual energies is attractive as it entails a smaller range of values, which in principle facilitates precise predictions. However, the evaluation of $T_{\mathrm{S,\,APBE}}$ becomes expensive for larger systems, since it is evaluated on an integration grid.

We evaluate the accuracy of energy and gradient predictions of Graphormer and Equiformer models on ground state label coefficients $\mathbf{p}^*$ from the QM9 validation set. The energy error is measuerd by the mean absolute error of the predicted energy, and the gradient error by the mean L1 norm of the corresponding deviation of gradient coeffiecients. Results are shown in tables 5.3 and 5.4. For both architectures, the model trained on the total energy E_tot achieves the best accuracies, with the lowest energy and gradient errors. The metrics of the Graphormer for targets $T_\mathrm{S} - T_{\mathrm{S,\,APBE}}$ and $T_\mathrm{S} + E_\mathrm{xc}$ are similar, with the former achieving slightly better results (see table 5.3).

However, the ultimate goal is density optimization, and there we find a different trend: All models trained on targets $T_\mathrm{S} - T_{\mathrm{S,\,APBE}}$ and E_tot fail to converge in density optimization. On the other hand, the Graphormer models trained on the sum of kinetic and exchange-correlation energy $T_\mathrm{S} + E_\mathrm{xc}$ reliably converge, while the models trained on the other targets fail to do so.

5.8.5 Training without natural reparametrization

As the natural reparametrization comes with a cubic time scaling in the system size, we attempt to train a model without it. We use the same settings for our best model, hence train to predict $T_\mathrm{S} + E_\mathrm{xc}$. The resulting validation metrics are worse: The mean absolute energy error at the ground state is 1.46 mHa (0.75 with natural reparametrization), and the corresponding gradient error is 5.24 (2.67 with natural reparametrization).

More importantly, we find that the models trained without natural reparametrization fail to converge in density optimization.

Table 5.3. Graphormer ground state errors. We compare the performance of Graphormer models trained with different targets on the perturbed QM9 dataset. The mean absolute energy errors at the ground state as well as the mean L1 gradient errors of the model are reported.

	$(T_S - T_{S,\,APBE})$	$(T_S + E_{xc})$	E_{tot}
Energy error [mHa]	0.61	0.75	0.84
Gradient error [Ha]	2.49	2.67	0.90

Table 5.4. Equiformer ground state errors. We compare the performance of Equiformer models trained with different targets on the QM9 dataset. Note that the run with target E_{tot} was trained on the unperturbed data (marked with an asteriks). The mean absolute energy errors at the ground state as well as the mean L1 gradient errors of the model are reported.

	$(T_S + E_{xc})$	E_{tot}
Energy error [mHa]	1.703	0.202*
Gradient error [Ha]	5.024	0.068*

5.8.6 Convergence and comparison to M-OFDFT

In table 5.5, we compare our best performing model to the state-of-the-art OF-DFT method M-OFDFT [9] on density optimization on the QM9 test set. Additionally, figure 5.13 shows a radar plot comparing the performance of our model to M-OFDFT on the QM9 test set.

The most striking difference is the convergence behaviour: While none of the M-OFDFT density optimization runs converge, with the final gradient norm consistently above 3×10^{-3}, our model reliably converges to a gradient norm below 1×10^{-4} in the first 1000 iterations. In figures 5.14 and 5.15, we demonstrate this qualitative difference by plotting the gradient norm and energy error during extended density optimziation runs for our best model and the model provided by [9]. If density optimization with the M-OFDFT model is continued for a longer time, many of the runs will diverge completely to energies and densities far from the ground state. In contrast, if we let density optimziation with our model continue even after the gradient norm falls below 1×10^{-4}, it will continue to decrease below 1×10^{-11}, demonstrating convergence to a stationary point in the energy landscape. We computed Hessians of our total energy functional at some of the predicted densities, and found that all of their eigenvalues are positive, confirming that they are indeed minima.

To make sure that our best model was not a fluke, we trained two additional

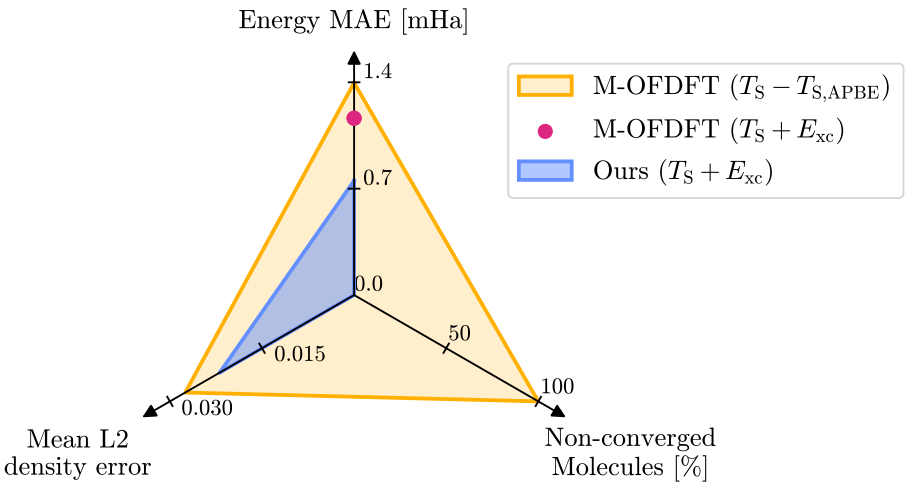


Figure 5.13. Radar plot. We compare our best model (see last row in table 5.2) to M-OFDFT [9] on density optimization on the QM9 dataset. The values for the yellow model are generated with our OF-DFT code, using the model provided by [9]. The magenta dot marks their best energy error, as reported in their paper. Our model, shown in blue, outperforms M-OFDFT in all three metrics, in particular achieving consistent convergence on all molecules of the QM9 test set using our SAD initialization.

models with identical hyperparameters but different random seeds. The performance of those models in density optimization is shown in table 5.5. One of the models is worse, not achieving reliable convergence in density optimization, while the other achieves similar results to the original model.

Using an elaborate "stopping criterion", M-OFDFT is able to pick-out precise ground state energies and densities from the density optimization trajectory. With our model, employing such a criterion is not neccessary, and we can simply stop the optimization after the gradient norm falls below a chosen threshold. Still, our resulting energies and densities are more precise than those obtained by M-OFDFT, see figure 5.13.

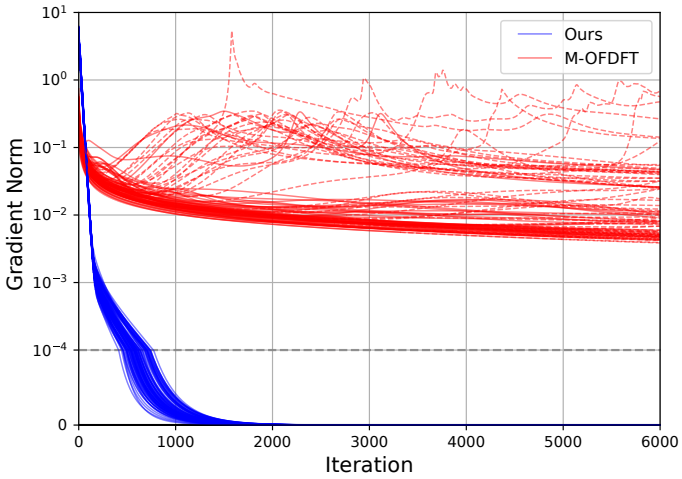


Figure 5.14. Gradient norm during density optimization, comparing M-OFDFT to ours. Shown are 50 exemplary density optimization runs with our best model (blue), and the model provied by [9] using our OF-DFT code (red), for a maximum of 6000 iterations. After the iteration which their "stopping criterion" picks out, their cuves are dashed. While none of the M-OFDFT runs converge (final gradient norm consistently above 3×10^{-3}), all of our runs converge to a gradient norm below 1×10^{-4} in the first 1000 iterations, and to a gradient norm below 1×10^{-11} at the final step. Figure drawn by master student Dominik Geng.

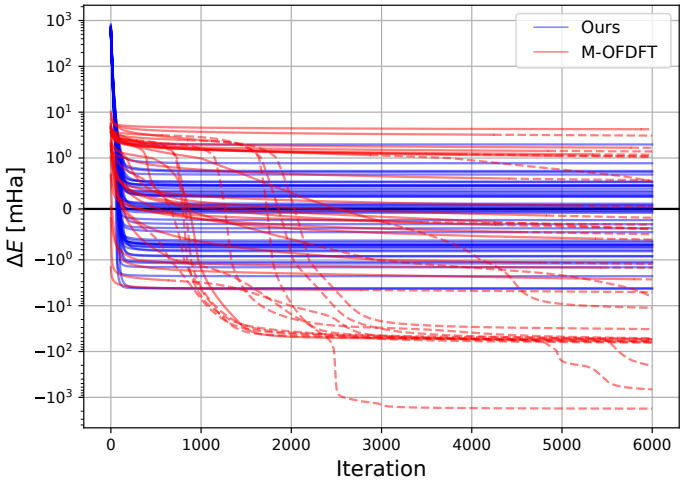


Figure 5.15. Total energy error during density optimization comparing M-OFDFT to ours. Shown are 30 exemplary density optimization runs, comparing our best model (blue) to the model provided by [9] using our OF-DFT code (red), with the same settings as in figure 5.14. While our runs converge to stable final energy errors, many of theirs diverge to huge energy errors. Figure drawn by master student Dominik Geng.

Table 5.5. Comparison of OF-DFT methods on QM9 density optimization. We compare our models to the state-of-the-art OF-DFT method M-OFDFT [9], as well as to the difference between our labels (KS-DFT with PBE exchange-correlation) and KS-DFT with the R2SCAN functional. We report the mean absolute error in the total energy E_{tot} and the mean per-electron L1 and L2 density errors. The last column shows percentage of samples for which the gradient norm fell below 10^{-4} in density optimziation.

Method	Target	ΔE [mHa]	$\frac{\|\rho-\rho^*\|_1}{N_e}$ [%]	$\frac{\|\rho-\rho^*\|_2}{N_e}$ [%]	conv. [%]
M-OFDFT [9]	$T_S - T_{S,APBE}$	1.397	0.551	0.0419	0
M-OFDFT [9]	$T_S + E_{xc}$	1.163	N/A	N/A	N/A
KS-DFT with R2SCAN	-	0.052	0.064	0.0015	-
Ours, full test set	$T_S + E_{xc}$	0.753	0.336	0.0333	100
Ours, seed 42	$T_S + E_{xc}$	0.702	0.464	0.0316	100
Ours, seed 43	$T_S + E_{xc}$	1.214	1.015	0.2867	61.4

5.9 Discussion

Building upon [9], we have for the first time presented a machine-learned energy functional which achieves chemical accuracy in convergent orbital-free density optimization on the varied organic molecules contained in the QM9 dataset [102, 103]. This was achieved using a supervised training paradigm similar to [9], but with a more diverse training set generated by adapting and improving the external potential sampling scheme introduced in chapter 4. This proved pivotal for proper convergence, as demonstrated by the ablations in section 5.8.3.

For all runs, ground state energy errors on the individual contributions (E_H, $T_S + E_{xc}$, E_{ext}) after density optimization are significantly larger than the total energy error (see e.g. table 5.2), which is to be expected: As the ground state lies at a minimum of the total energy functional, the error of the total energy scales quadratically with small deviations from the true ground state coefficients. The individual contributions do not have a minimum at the ground state and consequently their errors scale linearly in the vicinity of ρ^* and mostly cancel out. Converging to ground state densities which reach chemical accuracy also in these individual contributions would be a significantly more challenging task which we have to leave for future work.

Investigating three different training targets, $T_S - T_{S,\,APBE}$, $T_S + E_{xc}$, and E_{tot} in section 5.8.4, we have found that the $T_S + E_{xc}$ target yields the best results in density optimization, with a mean absolute energy error of 0.70 mHa and a mean

L2 density deviation per electron of 0.032%. Conceptually, $T_S + E_{xc}$ might also be the most appealing target, as it entails exactly the contributions to the total energy for which no tractable analytical expressions are known. $T_S - T_{S,\,APBE}$ has the additional downside of requiring the evaluation of both the APBE kinetic energy and the exchange-correlation functional (in our case PBE[46]) on a quadrature grid, which, while technically scaling linearly with system size, still becomes computationally expensive for larger systems. A possible reason for why the $T_S + E_{xc}$ models perform so well in density optimization might be explained by the only other nonlinear density-dependent part of the functional: The Hartree energy $E_H(\mathbf{p}) = \mathbf{p}^\top \mathbf{C}\mathbf{p}$ with the coulomb matrix $\mathbf{C}$. If $\mathbf{C}$ is positive definite, then even a simple linear approximation of $T_S + E_{xc}$ that accurately captures the gradient at the ground state would ensure a local minimum of the total energy at that state. At this point, the total gradient would vanish, and the Hessian would correspond exactly to $\mathbf{C}$. Of course, we do not learn a linear function, but adding the convex Hartree energy to the learned functional may lead to a similar effect, aiding convergence during density optimization.

Furthermore, we adapted the EquiformerV2 [62] architecture, which has some conceptual advantages over the Graphormer (see section 5.6.2) to the task of learning energy functionals for OF-DFT, but as of now cannot claim that it outperforms the Graphormer architecture on this task. However, to keep training times managable and similar to the Graphormer models, we quite drastically reduced the number of layers and dimensionality of the features in the Equiformer model compared to its original hyperparameter settings. We also used a smaller number of training epochs compared to the Graphormer. Hence, larger Equiformer models trained for longer durations may still outperform the Graphormer models, an avenue we might investigate in future work.

Regarding the bottlenecks in computational scaling of the method, we have made some progress by introducing an accurate superposition of atomic densities (SAD) as an initial guess for the density optimization. This is significantly faster to compute than the minimal atomic orbital (MINAO) guess (which also has to be fitted in the LCAB Ansatz) used in [9]. We showed that it provides just as, if not slightly more accurate initial guesses for density optimization (see sec 5.8.1) However, the natural reparametrization (see section 5.2.1) is still a bottleneck regarding the scaling, as it requires the inversion of the overlap matrix, which scales cubically with system size. Finding a solution to this problem should be one of the prime foci of future work, as, even though the prefactor of its cubic time scaling is relatively small, it will stand in the way of applying the method once a certain size is reached. We see two possible approaches to this problem: Either, one could attempt to simply make den-

sity optimization work in the standard basis, rendering the natural reparametrization unnecessary, or one could try to identify the key features of the natural reparametrization which are crucial for the success of the method. Then, one could try and find an alternative reparametrization, possibly of a local kind, which can be computed in linear time.

Another important avenue for future work is the investigation of the generalization capabilities of the model. On the one hand in regards to the chemical surroundings e.g. represented as solvation models like in chapter 4, and on the other hand towards larger system sizes. Hence, we are currently working on evaluating the performance of the model on the larger systems in the QMUGS [116] dataset, and adapting the architectures to favour generalization over system size by making the individual layers more local. Especially promising in this regard might be adapting the Allegro architecture [117], which is designed to have a fixed and finite receptive field, independent of the number of layers.

Chapter 6

Surrogate models to physical functionals

6.1 Motivation

In the previous chapters, we described our efforts to learn approximations to concrete, physical energy functionals for OF-DFT. The explicit goal was to obtain a machine-learning model that mimics a well-defined physical functional as precisely as possible, ideally on large swathes of chemical space and arbitrary electron densities.

However, taking a step back, one might formulate a higher-level goal: Make OF-DFT work! At a first glance, this may seem equivalent to what we have been doing so far. However, let us entertain the idea that less may be strictly required. Indeed, assume we have an energy functional which satisfies the following three conditions within a certain part of chemical space:

1. Orbital-free density optimization converges,
2. The resulting densities are accurate,
3. The resulting ground state energies are accurate.

If these conditions are met, we could claim that OF-DFT "works": Ground state densities and energies could be computed with the functional, and the functional could be used in any application which relies on these properties. Depending on the application, one might even be willing to relax the last point, e.g. if the properties of interest are computed from the densities alone, such as dipole moments.

In this penultimate chapter, we will focus on achieving these minimal requirements, without necessarily approximating the true energy functional. This is an attractive avenue for several reasons: First and foremost, there may exist such functionals which

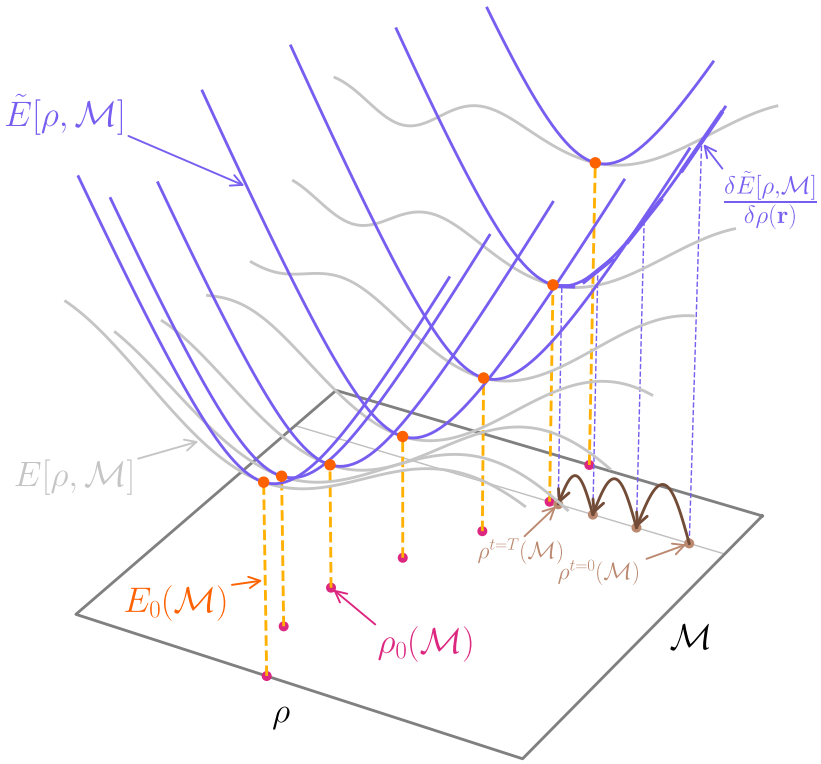


Figure 6.1. Surrogate Energy Landscape. A surrogate energy functional (purple) can be used in place of the true, physical functional (gray) in density optimization (brown). Here, a strong surrogate functional, which assigns the true ground state energy $E_0(\mathcal{M})$ to the ground state densities, is depicted. Compare with figure 1.1.

are easier to learn than the physical energy functional (for a conceptual illustration, see figure 6.1). Secondly, we only require ground state densities (and possibly energies) to train such models, alleviating the need to generate varied densities with energy and gradient labels for training, and allowing any method which provides ground state densities to be used as a training data source.

We will call these functionals *surrogate functionals*, and will define them more precisely in the next section 6.2. Afterward, we will propose loss-functions (section 6.3), methods for choosing training data (section 6.4), as well as certain architectural adjustments (section 6.5) all of which are tailored to the goal of achieving the three conditions outlined above.

The concept of surrogate functionals is closely related to that of Energy-Based Models (EBMs) [118], where a classification or regression task is framed as an energy minimization problem: Just as in OF-DFT, inference is performed via minimization of an energy predicted by a model. However, this minimization is oftentimes performed not via a gradient-based method. Surrogate functionals can be seen as a special case of EBMs, where (at least for strong surrogate functionals, see below) also the value of the energy functional at the minimum matters. We adapt and expand the machinery of EBMs to the specific requirements of OF-DFT, by proposing loss functions and training data which are tailored to the task of orbital-free density optimization.

6.2 Definitions: strong and weak

Let us define more precisely what a surrogate functional is. As the definition depends on the context, in particular the density optimization procedure, we first specify what this encompasses:

Definition 1 *A **density optimization procedure** is a pair formed by an initial guesser that maps molecules $\mathcal{M}$ to initial density coefficients $\mathbf{p}^{(0)}$, and an optimizer, which, given an energy functional E, maps $\mathcal{M}$ and $\mathbf{p}^{(0)}$ to final density coefficients $\mathbf{p}^{(T)}$.*

The density optimization procedure includes the choice of initial guess (e.g. SAD guess), as well as the choice of optimizer (e.g. gradient descent) and all of its parameters (e.g. learning rate, momentum). Now we can further define:

Definition 2 *A **surrogate functional** for a given density optimization procedure is a functional which, when used in place of the true energy functional in density optimization, leads to the true ground state coefficients $\mathbf{p}^{(T)} = \mathbf{p}^*$.*

Hence, in terms of finding the ground state, a surrogate functional is at least as good as the true energy functional, possibly even better: Since the true energy functional might have multiple local minima, successful optimization might depend on carefully chosen initial guesses and optimizers.

Furthermore, we can distinguish between two types of surrogate functionals:

Definition 3 *A **strong surrogate functional** is a surrogate functional which assigns the true ground state energy E_0^* to the true ground state coefficients $\mathbf{p}^*$.*

Hence, a strong surrogate functional can even be used to predict the true ground state energy, and is a full replacement for the true energy functional in the most common applications of OF-DFT.

If we do not explicitly require the ground state energy to be predicted correctly, we sometimes call the surrogate functional a **weak surrogate functional**, and this simpler requirement will be our primary focus in the following sections.

So far, we have only defined surrogate functionals to replace the total energy functional. However, surrogate functionals which replace only parts of the energy functional are also conceivable; pursuing this avenue is left for future work.

Here, we have defined surrogate functionals in the context of the LCAB Ansatz (see section 2.4.2), but note that the concept is equally applicable to other density representations.

6.3 Loss functions

Above we have motivated the use of surrogate functionals and established what they are. Let us now turn to the question of how to train them. In a supervised setting, the goal of reproducing the training labels is straightforward to express as a loss function which simply compares model predictions with ground-truth labels. For surrogate functionals, however, it is less obvious how to use only the ground state coefficients $\mathbf{p}^*$ to provide useful feedback to a model which predicts the energy $E[\mathbf{p}]$ at coefficients $\mathbf{p} \neq \mathbf{p}^*$.

For most optimization procedures, tackling this problem end-to-end via backpropagation through the density optimization procedure is infeasible due to both computational constraints and the fact that a newly initialized model may not even converge during density optimization.

Hence, we propose conditions which facilitate successful density optimization, and then design loss functions which enforce these conditions during training.

A commonality in the resulting losses is that they include some scalar function $L : \mathbb{R} \mapsto \mathbb{R}$. This is used to convert a discrepancy, i.e. degree of failure to fulfill some

condition, to a loss function. It could for instance be the Mean Squared Error (MSE) loss $L_2(x) = \|x\|^2$, the L1 loss $L_1(x) = \|x\|$ or a smooth version thereof, i.e.

$$L_{1,\text{smooth}}(x) = \begin{cases} x^2/(2\beta), & \text{if} \quad \|x\| < \beta \\ \|x\| - \beta/2, & \text{otherwise} \end{cases}, \tag{6.1}$$

for some $\beta > 0$. In the following subsections, we describe several possible loss functions which can be used to train surrogate functionals.

6.3.1 Lower bound loss

Let us describe the probably simplest surrogate loss function, motivated by the variational principle which tells us that the total electronic energy is lowest at the ground state density ρ_0. In other words, a learned functional E which respects this principle should assign any arbitrary density ρ an energy no-lower than $E[\rho_0]$. Expressing densities in the LCAB Ansatz, and making the models parameters $\boldsymbol{\theta}$ explicit, this motivates the *Lower Bound Loss*:

$$\mathcal{L}_{\text{lower-bound}} = L(\max\left(0, E(\mathbf{p}^*; \boldsymbol{\theta}) - E(\mathbf{p}; \boldsymbol{\theta})\right)).$$

Note how $E(\mathbf{p}; \boldsymbol{\theta})$ is compared to $E(\mathbf{p}^*; \theta)$ and not the *true* ground state energy E_0^*. We make this choice because otherwise the loss function would be zero everywhere as long as the network never predicts values below E_0^*. As chosen, $\mathcal{L}_{\text{lower-bound}} \equiv 0$ guarantees that the ground state coefficients $\mathbf{p}^*$ are a global minimum of E.

How useful is this loss function? Certainly it is fully compatible with the true energy functional, which obeys the variational principle. This makes it an attractive objective to use in conjunction with supervised training. However, not every energy functional perfectly fulfilling the lower bound loss is a valid surrogate functional: It may have arbitrarily many local minima in which density optimization might get stuck. Also, a constant energy functional leads to a lower bound loss of zero, and this would be entirely unhelpful during density optimization.

In conclusion, the simple lower bound loss is not useful on its own, but might be attractive to use in conjunction with other losses.

6.3.2 Gradient to ground state loss

The next surrogate loss function which we propose guarantees a better-behaved function. This comes with the cost of it not being fully compatible with the true energy functional, i.e. the loss would be nonzero if the model perfectly reproduced it.

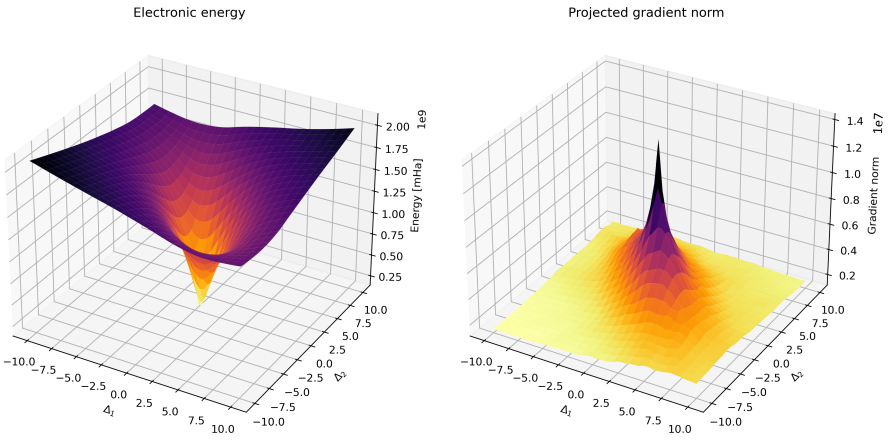


Figure 6.2. 2D Slice of energy and gradient-norm surface trained with gradient-to-ground-state loss alone. These plots show the energy surface (left) and gradient norm (right) of a preliminary model trained only with the gradient-to-ground-state loss (eq. 6.3) on a 2D slice of the input space, spanned by the directions from the ground state to the two penultimate SCF iterations. At least restricted to this slice, there is a minimum close to the ground state, however the gradient and coefficient scales are huge, hindering density optimization.

If the negative gradients $-\nabla E(\mathbf{p};\boldsymbol{\theta})$ would always point directly towards the ground state, density optimization would be straightforward. Hence, one could propse a loss

$$\mathcal{L} = L(1 - \cos\mathrm{sim}(\boldsymbol{\nabla}_{\mathbf{p}} E(\mathbf{p};\boldsymbol{\theta}),\, \mathbf{p} - \mathbf{p}^*)) \tag{6.2}$$

which is zero only if the gradient $\boldsymbol{\nabla}_{\mathbf{p}} E(\mathbf{p};\boldsymbol{\theta})$ points directly away from $\mathbf{p}^*$, such that the cosine similarity is 1. However, this would drastically restrict the shape of the energy surface and would certainly be incompatible with the true energy functional.

This motivates a similar but softer version, where the direction of the gradient is allowed to differ from $\mathbf{p} - \mathbf{p}^*$ by some maximum angle, or, put differently, the cosine similarity is only required to be larger than some $\Delta \in (0, 1)$:

$$\mathcal{L}_{\text{gradient-to-ground-state}} = L(\max(0, \Delta - \cos\mathrm{sim}(\boldsymbol{\nabla}_{\mathbf{p}} E(\mathbf{p};\boldsymbol{\theta}),\, \mathbf{p} - \mathbf{p}^*)))\,. \tag{6.3}$$

If this loss is zero for all possible input densities, the energy surface will have only a single minimum: Following the flow generated by $\boldsymbol{\nabla}_{\mathbf{p}} E(\mathbf{p};\boldsymbol{\theta})$ would continuously move densities towards the ground state.

Figure 6.2 shows the energy surface and gradient norm of a preliminary model trained with this loss function alone, on a 2D slice of the input space. This demonstrates one possible disadvantage of the loss function: It does not restrict the norm or

the predicted gradient at all, as it only acts on its direction via the cosine similarity. When the gradient-to-ground-state loss is used on its own this leads to vastly varying gradient scales which may be problematic during density optimization, at least if simple gradient descent is used there. Also, the loss is undefined if $\boldsymbol{\nabla}_{\mathbf{p}} E(\mathbf{p};\boldsymbol{\theta})$ is zero, and gets numerically unstable for very small gradient predictions. Luckily, both of these disadvantages can be mitigated by keeping the norm of the gradients in-check, e.g. by facilitating the loss that we describe next.

6.3.3 Gradient norm range loss

This loss is one way to keep the gradients of the model in a specified range, without dictating them precisely. Given minimum and maximum gradient norms $g_{\min}$ and $g_{\max}$, one can write

$$\mathcal{L} = L(\max(0,\, g_{\min} - \|\boldsymbol{\nabla}_{\mathbf{p}} E(\mathbf{p};\boldsymbol{\theta})\|,\, \|\boldsymbol{\nabla}_{\mathbf{p}} E(\mathbf{p};\boldsymbol{\theta})\| - g_{\max}))\,. \tag{6.4}$$

At the ground state, the gradients should be zero, and increase for densities further away from it, so in our experiments with the *gradient-norm-range loss*, we choose to scale the gradient range with the distance to the ground state coefficients $\mathbf{p}^*$:

$$\mathcal{L}_{\text{gradient-range}} = L(\max(0, \gamma_{\min}\|\mathbf{p}-\mathbf{p}^*\| - \|\boldsymbol{\nabla}_{\mathbf{p}} E(\mathbf{p};\boldsymbol{\theta})\|, \|\boldsymbol{\nabla}_{\mathbf{p}} E(\mathbf{p};\boldsymbol{\theta})\| - \gamma_{\max}\|\mathbf{p}-\mathbf{p}^*\|))\,, \tag{6.5}$$

with minimum and maximum distance-gradient scalings $\gamma_{\min}$ and $\gamma_{\max}$.

6.3.4 Gradient descent improvement loss

If one plans to use gradient descent in density optimization, one can design a loss function which directly optimizes for the performance of this optimizer. One way to do this is to require that the model predicts gradients which lead to steps towards the true ground state $\mathbf{p}^*$ when used in a gradient descent step. Multiple measures for this are conceivable, but a simple one is to require that each step reduces the distance to the ground state by at least a certain factor $\beta < 1$:

$$\mathcal{L}_{\text{gradient-descent-improvement}} = \max\left(0, \|\underbrace{\mathbf{p} - \lambda \boldsymbol{\nabla}_{\mathbf{p}} E(\mathbf{p})}_{\text{coeffs after step}} - \mathbf{p}^*\| - \beta\|\mathbf{p}-\mathbf{p}^*\|\right). \tag{6.6}$$

A great advantage of this loss is that if it is zero for all densities, density optimization with gradient descent will not only converge to the ground state but is guaranteed to do so quickly: If the distance of the initial guess to the ground state is d, the distance after n steps is at most $d\beta^n$. A possible disadvantage is that the loss is more restrictive

than the losses described above, and might not be compatible with the true energy functional for all but very conservative, i.e. high, choices of β.

For future work, on could adapt this loss to more sophisticated density optimization algorithms, e.g. gradient descent with momentum or line-search methods.

6.4 Surrogate training data

An ideal machine-learned energy functional would perfectly generalize over all of chemical space and all possible input density coefficients $\mathbf{p}$. In practice however, a trade-off between the extent of inputs which the model generalizes over and the accuracy of the model will likely be necessary. In other words, given constraints in model capacity and computational effort, it is unrealistic to aim at training a model which is highly accurate for every input density. Hence, one should decide which kinds of densities the model should excel on, and adjust the training data accordingly, as it primarily determines where the model works well.

In the previous chapters 4 and 5, the main consideration in this regard was to generate sufficiently *varied* training data, as without tricks, data from KS-DFT is quite restricted, leading models trained on it to fail in density optimization.

Here however, we find ourselves in the opposite situation: The surrogate loss functions introduced above (see section 6.3) are applicable to *any* set of coefficients $\mathbf{p} \in \mathbb{R}^n$, as long as the ground state coefficients $\mathbf{p}^*$ of the molecule are known. As we cannot densely sample $\mathbb{R}^n$ during training, we have to decide on a subset.

To aid this decision, let us revisit the definition of surrogate functionals and thereby our ultimate goal: successful density optimization. To achieve this, the minimal set of densities the model has to work well on is comprised of the density optimization path between the initial guess and the final converged coefficients, which should ideally be close to the true ground state. Because we hope to achieve a certain degree of robustness regarding choice of initial guess and optimizer used in density optimization, we extend this requirement to some neighborhood of this path.

A simple subset to aim for is a ball around the ground state coefficients, since we can assume that e.g. a learned initial guess will be quite close to the ground state coefficients already. This gives rise to the first way of generating training data for surrogate functionals: Sampling densities around the ground state.

6.4.1 Sampling densities around the ground state

During density optimization with a surrogate functional, the distance to the ground state coefficients should continuously decrease, and a high accuracy is increasingly more important as the distance to the ground state shrinks.

Therefore, simply sampling training coefficients from gaussians centered around the ground state coefficients might be a suboptimal choice, as the density of high-dimensional gaussians (we are in the high-dimensional case, as the coefficients have hundreds of components) is concentrated on a spherical shell. This would lead to almost constant distance to the ground state and few samples substantially closer to it.

Therefore, we propose to sample densities differently: First, sample the magnitude of the perturbation from the ground state $r \in \mathbb{R}$ according to some distribution R, then its direction d uniformly from the sphere:

$$\mathbf{p} = \mathbf{p}^* + \Delta\mathbf{p}\,, \quad \Delta\mathbf{p} = d \cdot r \tag{6.7}$$

$$\text{with} \quad d \sim U(\mathcal{S}^n)\,, \quad r \sim R\,. \tag{6.8}$$

For R, many distributions with non-zero probability between zero and the expected distance of the initial guess to the ground state would be conceivable choices; We use a Gaussian distribution with mean and standard deviation of 0.05, truncated at zero (i.e. assigning zero probability to all $r < 0$).

Some densities corresponding to coefficients sampled in this way may be unphysical, e.g. negative. But this is no flaw, rather an advantage as the learned functional should therefore be robust to such inputs, which might be especially important if the non-negativity is not enforced during density optimization. This is advantageous compared to supervised training, where the model is only ever exposed to valid inputs and hence might behave unpredictably on invalid ones.

6.4.2 Train-time density optimization via caching

Motivation

Above, we argued that it is sufficient if a surrogate functional minimizes the loss in the vicinity of the ground state and described a method to generate training samples from this vicinity. Does a functional trained on these samples with surrogate loss functions lead to successful density optimization? It might be possible, but in general (and in practice) there is a failure mode which can hinder it: When choosing and configuring the surrogate losses, one aim was to not restrict the shape of the energy

surface too much. This might allow the model to satisfy the condition enforced by the loss function on almost all sampled coefficients in unexpected and undesirable ways.

Let us demonstrate this subtle but important point with an example: If we would train a model with the "gradient to ground state" loss (see 6.3.2), we do not require the gradient to point exactly away from the ground state, but allow the angle to differ by some margin. Now assume that some components of $\mathbf{p}^*$ are easy to estimate (e.g. coefficients mainly describing core electrons), while others are much more challenging (e.g. diffuse basis functions). Then, the model might quickly learn to perfectly point the negative gradient towards the ground state in the "easy" dimensions, while predicting zero gradient in the "difficult" ones. If half of the dimensions are "easy" and half are "difficult", this would lead to an expected cosine similarity of $1/2$ if training densities are sampled isotropically around the ground state as proposed above. This would, depending on the chosen value of Δ in the loss (see eq. 6.3), lead to zero loss for most of the sampled densities. The situation would be even more extreme if a smaller set of the dimensions are "difficult" to predict. In density optimization, such a model might converge to the correct values in the "easy" dimensions, but fail completely in the "difficult" ones. A similar example can be made for the "gradient descent improvement" loss, described in section 6.3.4.

Hence, we propose to adapt the sampling of training densities towards the goal of density optimization: If density optimization is already performed during training, and the loss is ensured to be fulfilled in each step of these train-time density optimization trajectories, the failure mode is ruled out. Moreover, this allows to sample data exactly in the way we described as optimal above: Along density optimization trajectories.

Method

The most straightforward way to implement train-time density optimization might be the following: Load the density coefficients (or sample them on the fly) as usual, then conduct k density optimization steps, evaluate the model and losses on the results. Unfortunately, this quickly becomes infeasible for moderately large k.

Therefore, we conduct only one step of density optimization per training iteration, but *cache* the resulting coefficients for the next time the molecule is loaded in a training batch. Then, the structure of a training iteration is:

1. Load a training batch $\{(\mathcal{M}_i, \mathbf{p}_i, \mathbf{p}_i^*)\}_{i=1\dots B}$
2. Load coefficients from cache: For $i = 1 \dots B$: If $\mathcal{M}_i$ is in the cache C, overwrite the loaded coefficients with the ones from the cache, $\mathbf{p}_i \mapsto C[\mathcal{M}_i]$
3. Evaluate the model, yielding energies $E(\mathbf{p}_i; \theta)$ and gradients $\nabla_{\mathbf{p}} E(\mathbf{p}_i; \theta)$

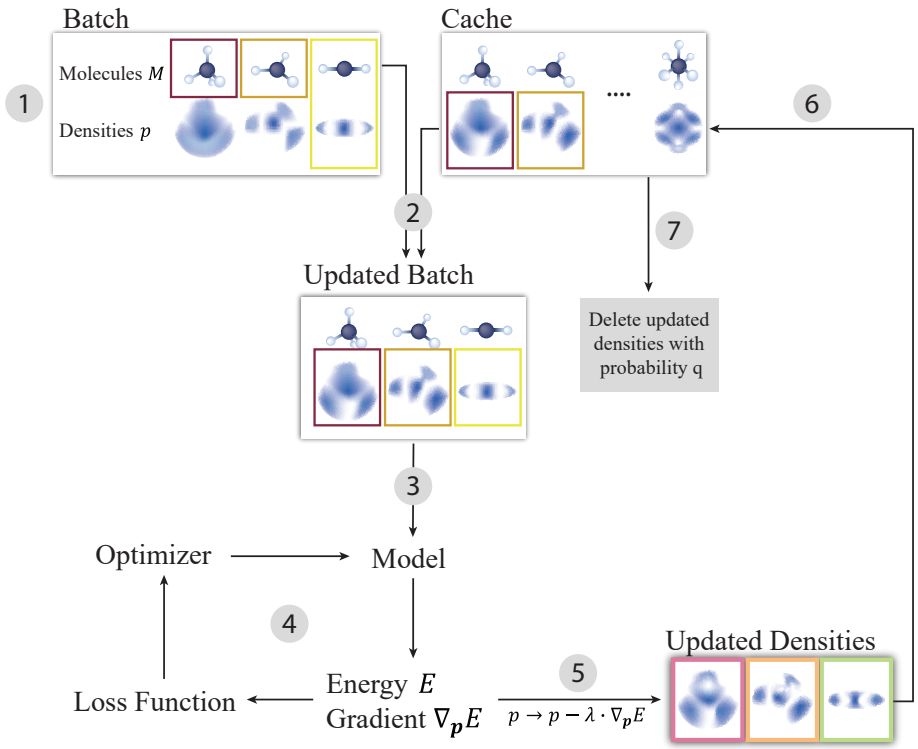


Figure 6.3. Train-time density optimization via caching. A training batch is loaded (1), updated with densities of all molecules in the batch which are present in the cache (2), passed through the model (3) yielding energies and gradients which are both used to evaluate the losses and update model parameters (4) as well as utilized in a density optimization step (5). The updated densities are written to the cache (6), before finally each molecule of the present batch is discarded from the cache with probability q (7).

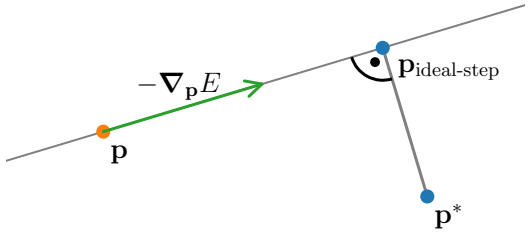


Figure 6.4. Ideal density optimization step. The ideal step chooses the point $\mathbf{p}_{\text{ideal-step}}$ on the ray anchored at $\mathbf{p}$ in direction $\nabla_{\mathbf{p}}$

4. Evaluate losses, backpropagate, and update model parameters θ

5. Make a density optimization step: $\mathbf{p}_i \mapsto \mathbf{p}_i - \lambda \nabla_{\mathbf{p}} E(\mathbf{p}_i; \theta)$

6. Save the updated coefficients to the cache: $C[\mathcal{M}_i] \mapsto \mathbf{p}_i$

7. Discard molecules in the batch from the cache with probability q_{reset}.

This procedure is also visualized in figure 6.3. Hence, each time a certain training molecule is loaded, another density optimization step is conducted on its density coefficients. Over multiple training epochs, these steps thus accumulate to possibly very long trajectories, without having to do more than a single density optimization step per training iteration.

The probabilistic reset in the last step prevents the model from overfitting to the states which density optimization converges to, as it guarantees that some percentage of training samples come straight from the originally chosen distribution around the ground state (see 6.4.1). In our experiments, we set q_{reset} to a value of 1%.

Note that we do not project the gradients (see section 5.3) prior to the density optimization step, hence allow for non-normalized densities during training.

A major strength of this approach is how directly it adapts the training to the final goal of density optimization: While density optimization is conducted across epochs with different states of the model, a convergent density optimization at train time of this type is much more indicative of generalization to density optimization on test geometries than simply fitting a fixed dataset of energies and gradients in the supervised case.

Ideal density optimization steps

Above, we described the density optimization step as a simple gradient descent step. However, at train-time we know the true ground state coefficients $\mathbf{p}^*$, and can hence speed up the optimization by making ideal steps in the sense of picking the step size λ such that the distance to the ground state is minimized:

$$\lambda = \underset{\lambda, \lambda > 0}{\arg\min} \left\| \mathbf{p} - \lambda \nabla_{\mathbf{p}} E(\mathbf{p}; \theta) - \mathbf{p}^* \right\|_2 . \tag{6.9}$$

This amounts to moving along the ray from $\mathbf{p}_i$ in the direction of the negative gradient until the difference vector to the ground state is orthogonal to the gradient (see figure 6.4), and is a simple one-dimensional optimization problem whose closed-form solution is

$$\lambda = \begin{cases} \frac{\boldsymbol{\nabla} E(\mathbf{p};\theta) \cdot (\mathbf{p} - \mathbf{p}^*)}{\|\boldsymbol{\nabla} E(\mathbf{p};\theta)\|^2}, & \text{if} \quad \boldsymbol{\nabla} E(\mathbf{p};\theta) \cdot (\mathbf{p} - \mathbf{p}^*) > 0, \\ 0, & \text{otherwise.} \end{cases} \tag{6.10}$$

Above we omitted the index i for clarity, but the ideal learning rates are calculated for each molecule in the batch individually.

This approach has two potential advantages: Fist, it takes fewer epochs to reach the same distance to the ground state, and thereby challenging densities are sampled more often. Second, it might lead to a more stable training process, as the densities are always moved towards the ground state, and potential spikes in the gradient prediction do not lead to large steps away from the ground state.

On the other hand, the ideal density optimization steps are of course not available at test time, and the model might not generalize as well to density optimization steps with a fixed step size.

Implementation of caching

The coefficient cache C could be implemented in a number of ways. For huge datasets it might be required to save the cached coefficients to disk. For MD17 and QM9 however, it is feasible to keep the cached coefficients not only in RAM, but even in GPU memory.

The loading from and saving to the cache is then performed in the main thread, and not in the individual data loading workers. This may not be ideal, as it is less parallelized, but still allows for decent performance (as the data is cached in GPU memory) while being much simpler to implement.

6.5 Surrogate architectures

While we mostly follow the architectural choices described in section 5.6, we make some adjustments for the training of surrogate functionals, which we describe in the following.

6.5.1 Dimension-wise rescaling

The dimension-wise rescaling module was introduced by [9] in order to optimize variational fitting of label energies and gradients. Since we do not aim to reproduce these labels when training surrogate functionals, this motivation no longer applies. In fact, we have found that the scaling of input coefficients can be disadvantageous when training surrogate models, as coefficients with small prefactors are sometimes completely ignored by trained networks.

Hence, we deactivate the per-coefficient scaling in the dimension-wise rescaling module, and only subtract the coefficient means while scaling with a fixed scalar factor a, which is an additional hyperparameter:

$$\mathbf{p} \mapsto a \cdot (\mathbf{p} - \bar{\mathbf{p}}) \,. \tag{6.11}$$

In our experiments, we use a value of 10 for the prefactor a.

6.5.2 Atomic reference module

Similar to dimension-wise rescaling, the atomic reference module is not necessary for training (particularly weak) surrogate functionals, hence we deactivate it.

In some experiments, however, we replace it with an isotropic parabola around $\bar{\mathbf{p}}$ (the superposition of non-normalized atomic densities, as introduced in section 5.4):

$$E_{\text{parabola}}(\mathbf{p}) = \alpha \|\mathbf{p} - \bar{\mathbf{p}}\|_2^2 \,, \tag{6.12}$$

with some prefactor $\alpha \in \mathbb{R}^+$.

This *Parabolic AtomRef* essentially gives the model a head start, by adding a simple functional with a minimum at $\bar{\mathbf{p}}$, and allows the machine-learned part of the functional to focus on moving this minimum to $\mathbf{p}^*$.

While the scalar factor α could be easily converted to a learnable parameter, we leave investigating this option to future work.

6.6 Results

When not stated otherwise, we utilize a downsized Equiformer (see [62] and section 5.6.2) architecture, since we have found it to perform well compared to the Graphormer in preliminary experiments. The precise hyperparameters are given in the appendix, see table B.2. A comparison for the training of surrogate models with the Graphormer architecture (see section 5.6.1) is given in section 6.6.3. When not stated otherwise, we work in the natural reparametrization basis (see section 5.2.1), which lead to the best results in 5.

Most experiments described in this section were conducted by bachelor student Mats Kothe whom I gave detailed instructions.

6.6.1 Data-driven Hyperparameter Choices

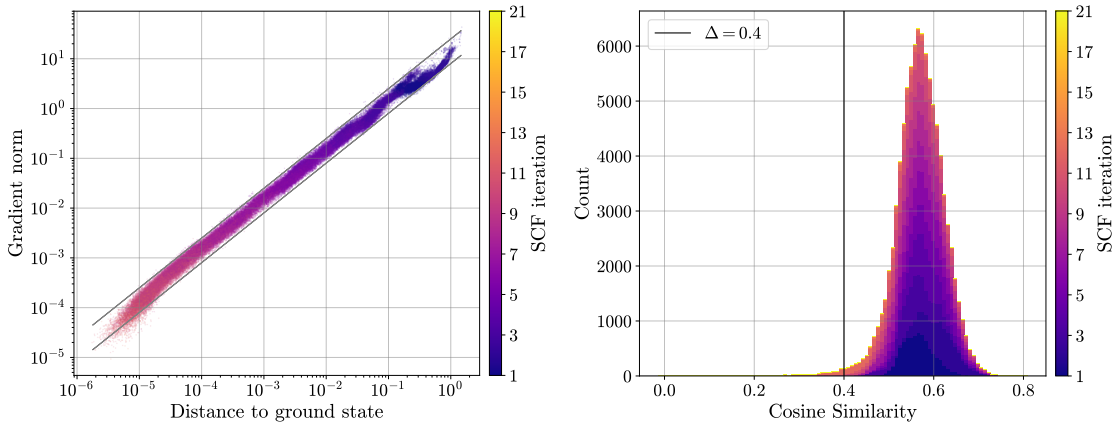


Figure 6.5. Data-driven choice of surrogate loss hyperparameters. In (a) we plot the norm of the gradient label against the distance to the ground state for samples from the unperturbed QM9 dataset. The gray lines represent the boundaries of the gradient range we restrict our surrogate functionals to when using the gradient-norm-range loss (eq. 6.5). In (b) we show a histogram over cosine similarities between inverted gradient labels and directions to the ground state, motivating our choice of $\Delta = 0.4$ for the gradient-to-ground-state-loss (eq. 6.3).

The loss functions and architectural adjustments introduced above give rise to a number of additional hyperparameters.

One approach to choose them is to use a data-driven method, where we estimate which hyperparameters would be appropriate, e.g. would lead to zero loss on the annotated training data.

Table 6.1. Density optimization with surrogate functionals trained on MD17. All models are trained with the gradient-to-ground-state loss (eq. 6.3) and, all but the first two additionally with the gradient-norm-range loss (eq. 6.5). The best-performing model is trained with $\alpha = 1$ in the parabolic atom reference module (eq. 6.12). Evaluations are conducted on the full MD17 test set.

Run	LR λ	converged [%]	$\frac{\|\rho-\rho^*\|_2}{N_e}$ [%]
$\alpha = 0$, no grad. scale	3.00×10^{-5}	0.00	4.07×10^{-2}
$\alpha = 1$, no grad. scale	3.00×10^{-5}	0.00	2.46×10^{-1}
$\alpha = 0$	5.00×10^{-3}	0.00	6.04×10^{-2}
$\alpha = 0.1$	5.00×10^{-3}	0.10	5.15×10^{-2}
$\alpha = 1$	5.00×10^{-3}	**100.00**	$\mathbf{4.44 \times 10^{-3}}$

Figure 6.5 (a) shows the distribution of the cosine similarity between ground-truth gradients and vectors pointing from the ground state $\mathbf{p}^*$ to the training densities $\mathbf{p}$. Based on this, we choose 0.4 as Δ parameter for the gradient-to-ground-state loss (see equation 6.3), as for almost all samples from the MD17 and QM9 datasets, the cosine similarities lie above this value.

We proceed similarly for the gradient-norm-range loss (see equation 6.5), and select $\gamma_{\text{min}} = 8$ and $\gamma_{\text{max}} = 20$ based on the distributions of the ratios of gradient norms over ground state distances, illustrated in figure 6.5 (b), as most of the ratios lie in this range.

While the distributions of the ground-truth labels of the physical energy functional are by no means a perfect indicator of the optimal hyperparameters for the surrogate losses, they at least allow us to make a more informed choice than random guessing. Additionally, this choice might become more important when combining surrogate losses with supervised training, which we plan on pursuing in future work.

6.6.2 Training on MD17

As a first step, we train weak surrogate models on a dataset consisting of the ethanol molecules from the MD17 database [100, 101]. The dataset consists of 10 000 geometries sampled along a molecular dynamics trajectory, for which we compute the ground state coefficients using KS-DFT, as described in section 5.1. 8000 geometries are used for training, 1000 for validation, and 1000 for testing.

We train the models with a sum of the gradient-to-ground-state loss (eq. 6.3) and, except for the first two, additionally with the gradient-norm-range loss (eq. 6.5). Following our choice in chapter 5 we use L1 versions of both losses, i.e. take $L(x) = |x|$.

in both eq. 6.3 and 6.5.

Preliminary experiments without density optimization at train time (see section 6.4.2) showed no promise in density optimization (only few initial steps were towards the ground state, before diverging completely). Therefore, we always activate it in all subsequent experiments which we present here. When training with the gradient-to-ground-state loss, we use ideal density optimization steps (see section 6.4.2) to update coefficients during train time density optimization.

Density optimization is usually conducted with gradient descent with a learning rate of $\lambda = 5 \times 10^{-3}$ for a maximum of 1000 steps, stopping early if the gradient norm falls below 1×10^{-4}. When training without the gradient-norm-range loss, gradient norms are not restricted, and the learning rate commonly has to be lowered to prevent jumps far away from the ground state. For each run, we progressively lower the learning rate until this is no longer observed.

Table 6.1 summarizes the results of the density optimization experiments on the test geometries of the MD17 dataset. The only model (last row) which achieves a convergence rate of 100% is trained with the parabolic atom reference model (see eq.6.12). Therefore, we conclude it is highly beneficial. Furthermore, the gradient-norm-range loss seems crucial for successful density optimization with gradient descent, as the first two models, which are trained without it, do not converge at all.

In the next section, we will transfer the parameters of the best-performing model to the much more challenging QM9 dataset, consisting of bigger molecules of various compositions, and evaluate its performance there.

6.6.3 Training on QM9

The best-performing hyperparameter configuration from the MD17 experiments served us as a starting point for training on the QM9 dataset. We find that the same hyperparameters work exceptionally well on QM9 too, yielding an energy functional which leads to convergent density optimization in fewer than 1000 iterations on nearly all, 99.978%, of the test geometries. The non-converged molecules are not excluded from the evaluation of the mean L2 density error, which is 6.81×10^{-3} per electron, thereby outperforming our best supervised model by more than a factor of 3 (compare with table 5.5). Density optimization typically converges within 300 iterations, as shown in figure 6.6.

In table 6.2, we show the results of training with different variations on this successful hyperparameter choice. Both lowering or increasing the prefactor α of the parabolic reference module deteriorated the performance, as did training without the natural reparametrization basis. Employing a cosine learning rate scheduler, which

Table 6.2. Density optimization with surrogate functionals trained on QM9. The first row is trained with the same configuration as the best-performing model on MD17, in the rows below individual hyperparameters are varied, as described in the first entry of each row. Mean per-electron density errors which outperform our supervised results from chapter 5 are underlined. Evaluations are conducted on the full QM9 test set.

Run	LR λ	converged [%]	$\frac{\|\rho-\rho^*\|_2}{N_e}$ [%]
Best settings on MD17	5.00×10^{-3}	**99.978**	$\mathbf{6.81 \times 10^{-3}}$
$\alpha = 10$	5.00×10^{-3}	99.910	$\underline{9.10 \times 10^{-3}}$
$\alpha = 0.1$	1.00×10^{-3}	0.000	1.70×10^{-2}
w/o natrep	5.00×10^{-5}	0.000	5.49×10^{-2}
cosine scheduler	5.00×10^{-3}	99.970	$\underline{9.16 \times 10^{-3}}$
grad. impr. loss	5.00×10^{-3}	88.140	1.86×10^{-1}
grad. impr. loss, no grad scale	5.00×10^{-3}	99.955	$\underline{9.18 \times 10^{-3}}$

we found to beneficial for supervised training, did not improve the results but also did not harm them catastrophically.

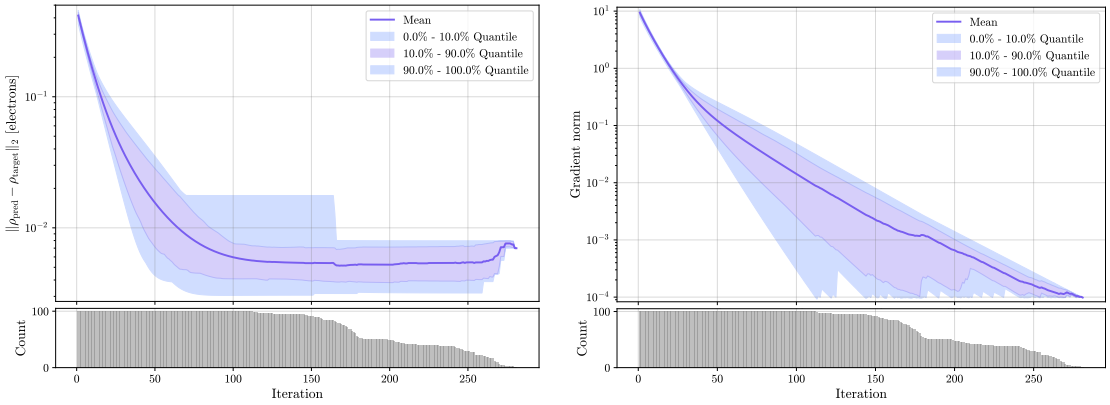
Finally, we trained models with the gradient descent improvement loss (see section 6.3.4). We set the minimum improvement factor β to 0.9, and the learning rate λ to 0.022. The gradient-to-ground-state loss was activated, and we trained one model with and one without the gradient-norm-range loss, the latter performing significantly better than the former as seen in the last two rows of table 6.2.

Surrogate functionals with the Graphormer Architecture

Since we found the Graphormer to work well in supervised training, we also trained surrogate functionals with this architecture: One with the best performing hyperparameters using the gradient-to-ground-state loss and the gradient-norm-range loss, and one with the gradient descent improvement loss. However, both models perform worse in density optimization, never converging to a gradient norm below 1×10^{-4} on the QM9 test set, and also do not achieve low density error.

Training strong surrogate functionals

To provide a proof of concept for the training of strong surrogate functionals, which also aim to predict the correct ground state energy, we train a model where we additionally add a supervised L1 energy loss to ground state densities with a relative weight of 0.2. The model converges on 99.918% of test molecules to an average per-electron



(a) **(b)**

Figure 6.6. Density optimization with a surrogate model on QM9. The model used here is the best-performing one shown in table 6.2, i.e. an Equiformer trained using the gradient-to-ground-state loss (see eq. 6.3) and the gradient-range loss (see eq. 6.5). The L2 density error (a) and the gradient norm (b) demonstrate the rapid convergence in fewer than 300 iterations. Compare with figure 5.12 of our best supervised model.

L1 density error of 8.62×10^{-5}, thereby only slightly worse than the best-performing model without the energy loss. However, the achieved energy error of 88.1 mHa is far from chemical accuracy and cannot compete with our supervised models.

6.7 Discussion

In this chapter we have introduced the concept of surrogate functionals, an application of the energy-based model framework [118], to the problem of solving for the electronic ground state density via orbital-free density optimization.

We have demonstrated that a carefully chosen set of loss functions together with a train-time density optimization scheme (see section 6.4.2) can lead to successful density optimization with a surrogate functional, on ethanol geometries from MD17 as well as on the challenging QM9 dataset. Indeed, either combining the gradient-to-ground-state loss (eq. 6.3) with the gradient-norm-range loss (eq. 6.5), or utilizing the gradient descent improvement loss (eq. 6.6) allowed us to train surrogate models which outperform our best supervised models in terms of density error after density optimization by a wide margin, more than a factor of three as measured by the mean L2 density error per electron.

A puzzling observation is that in our experiments, even for surrogate models, transforming the atomic basis via natural reparametrization (see section 5.2.1) seems

to be crucial for successful density optimization. This is surprising, as the main reason for introducing it, reducing the gradient scale, does not apply to surrogate models, because ground-truth gradients are not used during training. We do not have a satisfying explanation for this observation. The way in which natural reparametrization transforms the ground state coefficients seems to be beneficial for a model learning to predict them.

In terms of strong surrogate functionals, which additionally should map the ground state density to the correct energy, we present an initial proof of concept, but find that the energy error is far from chemical accuracy and cannot compete with our supervised models (see 6.6.3). Training more accurate strong surrogate functionals is an interesting avenue for future work, and many approaches to improve upon our initial foray into this direction are conceivable. Furthermore, one could try and combine supervised and surrogate losses which are compatible with the physical energy functional in a joint training scheme. Alternatively, one could fine-tune directly supervised models with surrogate losses after supervised training. In particular, fine-tuning a model on larger systems where generating a sufficient amount of varied energy and gradient labels may be prohibitively expensive could be of interest.

Indeed, a major benefit of surrogate functionals over directly supervised models, which we have not exploited in this chapter, is the fact that only the ground state density is needed for training. This opens the door to data-generation methods other than KS-DFT, even if they do not yield energy or gradient labels.

Chapter 7

Contributions and outlook

In this thesis[1], we have presented the development of machine-learned energy functionals for orbital-free density functional theory (OF-DFT). We have shown that it is possible to learn the kinetic energy functional from data, and that the resulting machine-learned energy functional can be used to optimize electron densities with chemical accuracy.

In chapter 4, we introduced KineticNet, the first deep neural network architecture which was successfully trained to reproduce the kinetic energy with chemical accuracy across a number of small molecules, demonstrating generalization over both input densities and geometries, and reproducing chemical bonding in orbital-free density optimization in two electron systems. Careful and well-motivated design choices for the model architecture (section 4.2) required expert knowledge from both quantum chemistry and machine learning. Paired with a novel scheme of generating labels for the kinetic energy and gradient which augmented the width of training distribution (see section 4.3), they allowed us to achieve this result. It served as a crucial proof of principle for machine-learned OF-DFT, and laid the foundation for the subsequent work on larger-scale systems.

With this goal in mind, in chapter 5, we transition from representing electronic densities on a quadrature grid to the more efficient representation in terms of the LCAB Ansatz (see section 2.4.2) developed independently but published before us in the seminal work [9]. Relative to their work, we introduce a number of key enhance-

[1]Significant work not presented in this thesis has been done by the author on the evaluation protocol for an imaging-based test for the presence of the Sars-CoV-2 virus. At a time when antigen tests were not yet available, this work [119] was utilized in the study "Prevalence of SARS-CoV-2 infection in children and their parents in Southwest Germany" [120], leading to the conclusion that during the time-frame and the surveyed region of the study, children aged 1-10 were no particular drivers of the pandemic, an important result for decision makers who soon after decided to reopen schools and kindergartens.

ments. Most notable among them is an improved version of the external potential sampling scheme that we introduced in chapter 4 (see section 5.5). Ultimately, we outperform [9] in all metrics on the QM9 dataset, while addressing their key shortcoming: For the first time, we present a machine-learned energy functional that can be used in properly convergent density optimization with chemical accuracy across a wide range of molecular systems, while setting a new state of the art in OF-DFT regarding ground state energy and density prediction.

Finally, in chapter 6, we introduce the concept of surrogate functionals, which aim to replace the exact, physical energy functional in density optimization without attempting to perfectly mimic it. Such a functional might be significantly easier to learn, while still allowing to find the ground state of a system in a variational manner, which is the baseline task of DFT. We combine surrogate loss functions (section 6.3) with a dynamic sampling procedure which allows performing density optimization at train time (section 6.4.2). We successfully train an energy functional which further improves upon the best accuracy of ground state densities resulting from variational density optimization in more orthodox OF-DFT published so far. By relaxing the requirements on functional approximations, it opens up new possibilities for the development of machine-learned energy functionals.

Outlook

In this thesis, we have demonstrated significant progress towards making machine learned orbital-free Density Functional Theory on molecular systems a reality. Nevertheless, in future work a number of key challenges needs to be overcome in order for it to be adapted by practitioners: The potential of linear scaling has to be realized and generalizability of the method has to be improved. Furthermore, the learned functionals are limited by the accuracy of Kohn-Sham calculations which we treated as ground-truth here.

Regarding the first point, the cubic scaling of the natural reparametrization used in chapters 5 and 6 is the first hurdle that we aim to overcome in future work. We believe that this is also the main obstacle, as modifying the architectures which we currently use such that evaluation of our learned functionals scales linearly is relatively straightforward. We could do this by introducing distance cutoffs, and the Hartree term can also be approximated accurately in linear time due to the nearsightedness of electronic matter, at least for gapped systems.

Secondly, improving generalizability of the learned functionals can be approached from two sides, and we believe both are equally important: On the machine-learning side, architectures could be further adapted to the task at hand. Incorporating exact

relations, such as the spatial scaling relation for the non-interacting kinetic energy, or known asymptotic behaviors of density functionals, directly into the models could prove highly beneficial. This approach has a long tradition in classical density functional approximations, and while challenging, we believe it is not infeasible to achieve similar integration for machine-learned functionals. On the other side, we believe that further improving data generation schemes is crucial. Generating datasets for more varied molecular geometries including larger systems is straightforward, but was out of scope for this thesis. Generating varied densities has proven critical for the results presented here. Modifying parameters of the method we introduced, e.g. adapting the way we sample external potential modifications to better cover the density coefficient space, could be a promising step forwards.

Regarding the precision of ground-truth data, utilizing larger basis-sets and more precise exchange-correlation functionals in KS-DFT is of course an option. In the longer term, one might strive for more accurate but still compact representations of the electron density, for example by employing orbital types other than Gaussian. Once research pushes beyond KS-DFT accuracy, data generation will become even more expensive and supplying prior knowledge to machine learning methods is known to pay off particularly in the resulting low-data regime. Also, approaches like surrogate functionals, whose training only requires ground-state densities, may prove useful, especially when combined with pre-training on lower accuracy but more abundant data.

Given the immense popularity of Kohn-Sham DFT, rightfully regarded as the workhorse of quantum chemistry, we believe that once its orbital-free counterpart outperforms it on a single important practical application by a sufficiently large margin, research in the area might accelerate and receive much greater attention from the machine-learning and computational chemistry and materials science communities.

Machine-learned OF-DFT will, of course, have to compete with alternative machine-learning approaches, such as direct prediction of ground-state energies. While it is difficult to predict which methods will prevail, we believe it will neither be those which forego all physical knowledge and rely solely on big datasets, nor those which aim to change as little as possible in existing quantum chemical codes, but rather the methods somewhere in between, such as those introduced in this thesis.

Modern machine learning is still a relatively young field, and its application to computational chemistry even more so. We believe that many breakthroughs in the intersection of these fields are yet to happen, and that these are most likely when expertise from both fields is tightly integrated with each other.

It is an exciting age for applications of machine learning in quantum chemistry,

and this thesis underlines the potential of machine-learned energy functionals to revolutionize the way we perform electronic structure calculations.

Chapter 8

Appendix

A Learning a transferable kinetic energy functional on quadrature grids

A.1 Data generation

As mentioned in section 4.3, to generate the training data for our model we slightly perturbed the external potential of our molecules to sample a diverse set of densities as solution of Kohn-Sham-DFT and thereby calculate our targets. To achieve this we used the pyscf package as code base and implemented our own restricted Kohn Sham class which takes an additional Matrix and adds it to the external potential matrix in the Hamiltonian of the SCF procedure.
For the sampling of those perturbation matrices the following approach was adopted:

1. (Relative) entries of the perturbation matrix are drawn from some random distribution and ensure a symmetric matrix

2. the norm of the perturbation matrix is drawn from some distribution and the matrix is normalized accordingly

The distributions are chosen to ensure that a majority of data points are somewhat close to the ground state. The reasoning being that for accurate convergence and ground state values the machine learning model should make precise predictions in this part of density space while far away from the solution a rough estimate of the kinetic energy and potential should be enough to guide the OF-DFT solver in the correct direction. Details regarding those distributions in the different basis sets are given in table A.1. All data sets have been calculated using at the BLYP/cc-pVDZ level of theory. The distribution of the kinetic and total energies for the H2O dataset are shown in figure A.1.

Table A.1. Data sets used for training KineticNet. The norm of the perturbation matrix is sampled from a normal distribution with a minimum norm of 0.005 and the mean and standard deviation given in the table. The relative entries of these matrices are sampled from a normal distribution with mean and standard deviation given in the table.

Data set	Norm		Matrix elements	
	μ	σ	μ	σ
HF	0.25	0.05	0.0	0.2
Ne_2	0.25	0.05	0.0	0.2
H_2	0.25	0.05	0.0	0.2
$H_3{}^+$	0.25	0.05	0.0	0.2
He	0.25	0.05	0.0	0.2
H_2O	0.00	0.10	0.0	0.2

A.2 OF-DFT implementation

Our OF-DFT implementation is based on the pyscf package, which we use to compute all the required integrals. Density fitting as implemented in the pyscf package is used for the calculation of the Coulomb matrix.

A.3 Correspondence between KS and OF Ansatz for two electrons

Recall the equation for the electron density in terms of the coefficients c_ν in our OF approach:

$$\rho(\mathbf{r}) = \Big(\sum_\nu c_\nu \chi_\nu(\mathbf{r}) \Big)^2 \quad . \tag{8.1}$$

In KS-DFT one can write the electron density in the basis of atomic basis functions using the molecular orbital coefficients $m_{i\nu}$:

$$\rho(\mathbf{r}) = \sum_i \Big(\sum_\nu m_{i\nu} \chi_\nu(\mathbf{r}) \Big)^2 \quad . \tag{8.2}$$

For two electron systems, there is only one molecular orbital, hence the first sum disappears. Thus, the two expressions for the density are equal for $m_{1\nu} = c_\nu$.

A.4 Atomic contributions

The atomic contributions have been calculated using either restricted or restricted open-shell KS-DFT at the BLYP/cc-pVDZ level of theory. An “atomic” initial guess

was used, as implemented in the pyscf package. The convergence tolerance was set to 10^{-4} and a grid level of 2 was used. Symmetry was employed to remove directional bias. The usage of symmetry ensures separation w.r.t. angular momentum. This allows the following procedure for spherical symmetrization of p type orbitals: First the MO coefficients and energies are averaged weighted by their occupation. Next the electrons in p-orbitals are evenly distributed over all three p-orbitals. As this procedure has only been implemented for p type orbitals only elements up to Neon can be used.

A.5 Model hyperparameters

We use identical hyperparameters for our model for the kinetic energy and for the kinetic potential. Our models employ $L = 5$ atom-atom interaction layers. We choose the number of features per order l such that approximately the same number of floats are used for each l. For the encoder and decoder, we use features of type (40, 14, 8, 6, 4) (i.e. 40 scalars, 14 vectors, 8 $l = 2$ tensors and so on), for all layers in-between features of type (101, 34, 20, 14, 11). We use a radial basis consisting of 32 functions for the encoder and decoder, and 16 functions for the atom-atom interaction layers.

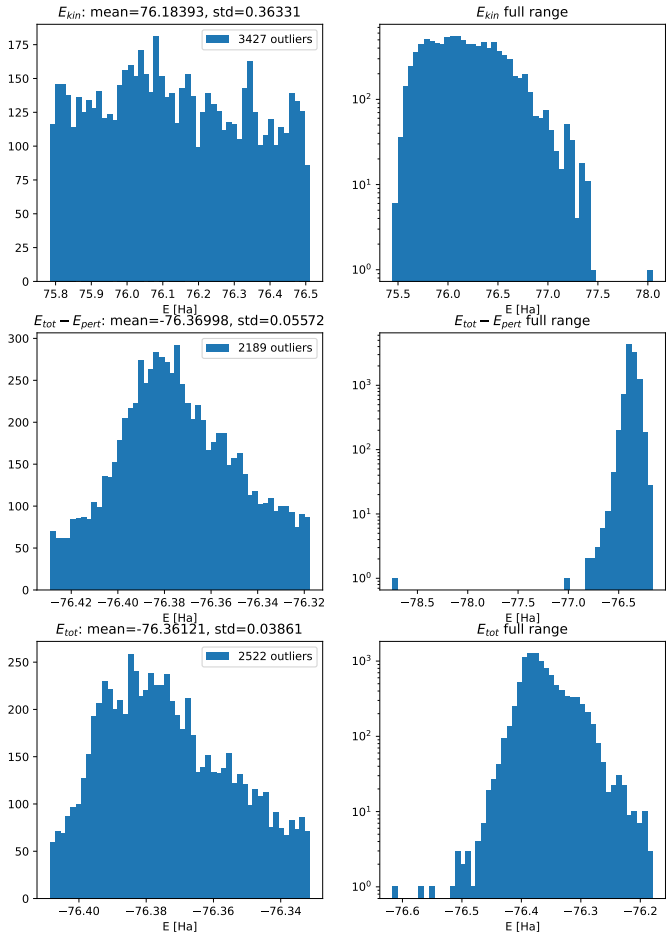


Figure A.1. Distribution of the kinetic energy E_{kin}, the total energy without the contribution from our perturbation to the external potential $E_{\text{tot}} - E_{\text{pert}}$ and the total energy in the H_2O data set.

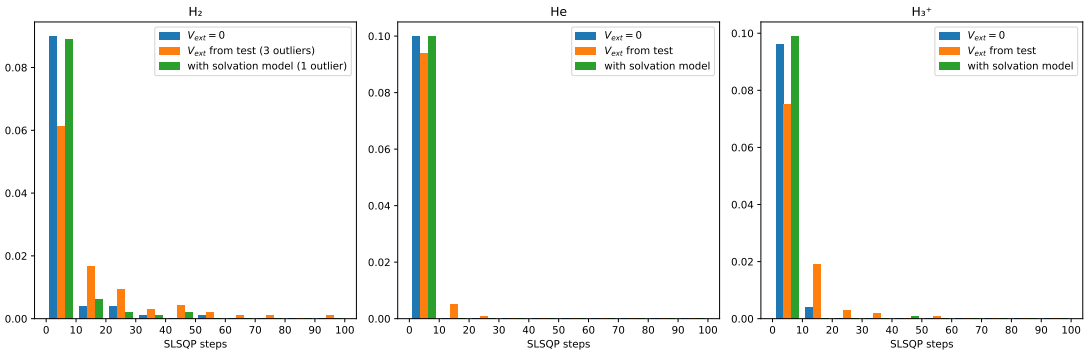


Figure A.2. Distribution of steps until convergence for the three two-electron systems and different density optimization modes.

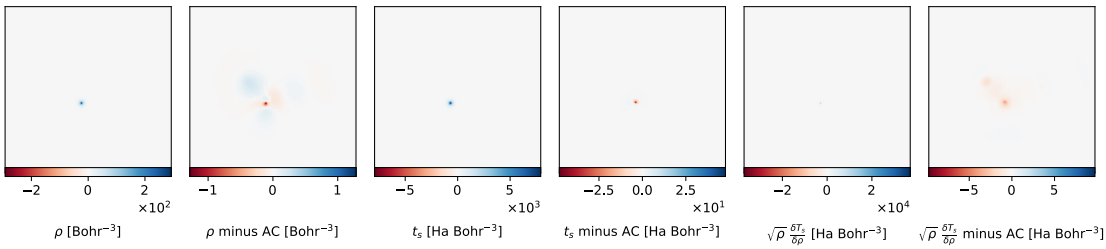


Figure A.3. Effectiveness of subtracting atomic contributions demonstrated for H_2O. For electron density, kinetic energy density as well as our target for the kinetic potential, subtracting the ACs decreases the value range by at least two orders of magnitude.

B Equiformer hyperparameters

In table B.2 we list the hyperparameters used for the EquiformerV2 architecture in chapters 5 and 6.

Hyperparameter	**Value in 5.8**	**Value in 6.6**
optimizer	AdamW	AdamW
weight decay	1×10^{-10}	1×10^{-10}
learning rate	1×10^{-3}	1×10^{-3}
learning rate scheduler	Cosine	Linear
batch size	32	90
epochs	25	45
max_neighbors	20	20
max_radius	12.0	12.0
max_num_elements	90	90
num_layers	6	4
sphere_channels	32	16
attn_hidden_channels	32	16
num_heads	4	4
attn_alpha_channels	32	16
attn_value_channels	8	8
ffn_hidden_channels	64	32
lmax_list	[5]	[5]
mmax_list	[5]	[5]
grid_resolution	18	18
num_sphere_samples	64	64
edge_channels	64	32
use_atom_edge_embedding	True	True
share_atom_edge_embedding	False	False
distance_function	“gaussian”	“gaussian”
num_distance_basis	128	64
attn_activation	“silu”	“silu”
use_s2_act_attn	False	False
use_attn_renorm	True	True
ffn_activation	“silu”	“silu”
use_gate_act	False	False
use_grid_mlp	True	True
use_sep_s2_act	True	True
alpha_drop	0.0	0.0
drop_path_rate	0.0	0.0
proj_drop	0.0	0.0

Table B.2. EquiformerV2 Hyperparameter choices. A detailed explanation of hyperparameters in the lower part of the table can be found in the paper introducing the architecture [62].

Authors Bibliography

This bibliography contains the publications which the author contributed to during the course of his PhD.

Chapter 4 of this thesis is based on this publication:

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. *The Journal of Chemical Physics*, 159(14), 2023

The following publications are not used in this thesis:

Constantin Pape, Roman Remme, Adrian Wolny, Sylvia Olberg, Steffen Wolf, Lorenzo Cerrone, Mirko Cortese, Severina Klaus, Bojana Lucic, Stephanie Ullrich, et al. Microscopy-based assay for semi-quantitative detection of SARS-CoV-2 specific antibodies in human sera: A semi-quantitative, high throughput, microscopy-based assay expands existing approaches to measure SARS-CoV-2 specific antibody levels in human sera. *Bioessays*, 43(3):2000257, 2021
(shared first authorship with Constantin Pape)

Burkhard Tönshoff, Barbara Müller, Roland Elling, Hanna Renk, Peter Meissner, Hartmut Hengel, Sven F. Garbade, Meinhard Kieser, Kathrin Jeltsch, Jürgen Grulich-Henn, Julia Euler, Maximilian Stich, Kristine Chobanyan-Jürgens, Maria Zernickel, Aleš Janda, Lena Wölfle, Thomas Stamminger, Thomas Iftner, Tina Ganzenmueller, Christian Schmitt, Tessa Görne, Vibor Laketa, Sylvia Olberg, Anna Plaszczyca, Mirko Cortese, Ralf Bartenschlager, Constantin Pape, Roman Remme, Daniela Huzly, Marcus Panning, Sebastian Weigang, Sebastian Giese, Kevin Ciminski, Jakob Ankerhold, Georg Kochs, Martin Schwemmle, Rupert Handgretinger, Charlotte M. Niemeyer, Corinna Engel, Winfried V. Kern, Georg Friedrich Hoffmann, Axel R. Franz, Philipp Henneke, Klaus-Michael Debatin, and Hans-Georg Kräusslich. Prevalence of SARS-

CoV-2 infection in children and their parents in southwest Germany. *JAMA pediatrics*, 175(6):586–593, 2021

Henning Arlt, Xuewu Sui, Brayden Folger, Carson Adams, Xiao Chen, Roman Remme, Fred A Hamprecht, Frank DiMaio, Maofu Liao, Joel M Goodman, et al. Seipin forms a flexible cage at lipid droplet formation sites. *Nature structural & molecular biology*, 29(3):194–202, 2022

Peter Lippmann, Gerrit Gerhartz, Roman Remme, and Fred A Hamprecht. Tensor frames–how to make any message passing network equivariant. *arXiv preprint arXiv:2405.15389*, 2024

Bibliography

[1] P. Hohenberg and W. Kohn. Inhomogeneous electron gas. *Phys. Rev.*, 136(3): B864–B871, 1964. ISSN 0031-899X. doi: 10.1103/PhysRev.136.B864. URL `https://link.aps.org/doi/10.1103/PhysRev.136.B864`.

[2] Wenhui Mi, Kai Luo, SB Trickey, and Michele Pavanello. Orbital-free density functional theory: An attractive electronic structure method for large-scale first-principles simulations. *Chemical Reviews*, 123(21):12039–12104, 2023.

[3] W. Kohn and L. J. Sham. Self-consistent equations including exchange and correlation effects. 140(4):A1133–A1138. doi: 10.1103/PhysRev.140.A1133. URL `https://link.aps.org/doi/10.1103/PhysRev.140.A1133`. Publisher: American Physical Society.

[4] Robert O Jones. Density functional theory: Its origins, rise to prominence, and future. *Reviews of modern physics*, 87(3):897–923, 2015.

[5] Michael M Bronstein, Joan Bruna, Yann LeCun, Arthur Szlam, and Pierre Vandergheynst. Geometric deep learning: going beyond euclidean data. *IEEE Signal Processing Magazine*, 34(4):18–42, 2017.

[6] Kevin Ryczko, Sebastian J Wetzel, Roger G Melko, and Isaac Tamblyn. Toward orbital-free density functional theory with small data sets and deep learning. *Journal of Chemical Theory and Computation*, 18(2):1122–1128, 2022.

[7] 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. *Physical Chemistry Chemical Physics*, 21(1): 378–395, 2019.

[8] 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.

[9] He Zhang, Siyuan Liu, Jiacheng You, Chang Liu, Shuxin Zheng, Ziheng Lu, Tong Wang, Nanning Zheng, and Bin Shao. M-ofdft: Overcoming the barrier of orbital-free density functional theory for molecular systems using deep learning. *arXiv e-prints*, pages arXiv–2309, 2023.

[10] R.G. Parr and Y. Weitao. *Density-Functional Theory of Atoms and Molecules*. International Series of Monographs on Chemistry. Oxford University Press, 1994. ISBN 9780195357738.

[11] Tosio Kato. On the eigenfunctions of many-particle systems in quantum mechanics. *Communications on Pure and Applied Mathematics*, 10(2):151–177, 1957.

[12] John P Perdew and Alex Zunger. Self-interaction correction to density-functional approximations for many-electron systems. *Physical review B*, 23 (10):5048, 1981.

[13] John P. Perdew, Kieron Burke, and Matthias Ernzerhof. Generalized gradient approximation made simple. 77(18):3865–3868. doi: 10.1103/PhysRevLett.77.3865. URL `https://link.aps.org/doi/10.1103/PhysRevLett.77.3865`. Publisher: American Physical Society.

[14] Axel D. Becke. Density-functional thermochemistry. iii. the role of exact exchange. *The Journal of Chemical Physics*, 98(7):5648–5652, 04 1993. ISSN 0021-9606. doi: 10.1063/1.464913. URL `https://doi.org/10.1063/1.464913`.

[15] Lars Goerigk and Stefan Grimme. Double-hybrid density functionals. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 4(6):576–600, 2014.

[16] Mel Levy. Universal variational functionals of electron densities, first-order density matrices, and natural spin-orbitals and solution of the v-representability problem. 76(12):6062–6065. doi: 10.1073/pnas.76.12.6062. URL `https://www.pnas.org/doi/10.1073/pnas.76.12.6062`. Publisher: Proceedings of the National Academy of Sciences.

[17] Elliott H Lieb. Density functionals for coulomb systems. *International Journal of Quantum Chemistry*, 24(3):243–277, 1983.

[18] J. Almlöf, K. Faegri Jr., and K. Korsell. Principles for a direct SCF approach to LICAO–MOab-initio calculations. 3(3):385–399. ISSN 1096-987X. doi: 10.1002/jcc.540030314. URL `https:`

//onlinelibrary.wiley.com/doi/abs/10.1002/jcc.540030314. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/jcc.540030314.

[19] J. H. Van Lenthe, R. Zwaans, H. J. J. Van Dam, and M. F. Guest. Starting SCF calculations by superposition of atomic densities. 27(8):926–932. ISSN 1096-987X. doi: 10.1002/jcc.20393. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/jcc.20393. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/jcc.20393.

[20] Péter Pulay. Convergence acceleration of iterative sequences. the case of scf iteration. 73(2):393–398, . ISSN 0009-2614. doi: 10.1016/0009-2614(80)80396-4. URL https://www.sciencedirect.com/science/article/pii/0009261480803964.

[21] John A Pople and David L Beveridge. Molecular orbital theory. *C0., NY*, 1970.

[22] J Stephen Binkley, John A Pople, and Warren J Hehre. Self-consistent molecular orbital methods. 21. small split-valence basis sets for first-row elements. *Journal of the American Chemical Society*, 102(3):939–947, 1980.

[23] Warren J Hehre, Robert F Stewart, and John A Pople. Self-consistent molecular-orbital methods. i. use of gaussian expansions of slater-type atomic orbitals. *The Journal of Chemical Physics*, 51(6):2657–2664, 1969.

[24] Praveen C Hariharan and John A Pople. The influence of polarization functions on molecular orbital hydrogenation energies. *Theoretica chimica acta*, 28:213–222, 1973.

[25] RBJS Krishnan, J Stephen Binkley, Rolf Seeger, and John A Pople. Self-consistent molecular orbital methods. xx. a basis set for correlated wave functions. *The Journal of chemical physics*, 72(1):650–654, 1980.

[26] Michelle M Francl, William J Pietro, Warren J Hehre, J Stephen Binkley, Mark S Gordon, Douglas J DeFrees, and John A Pople. Self-consistent molecular orbital methods. xxiii. a polarization-type basis set for second-row elements. *The Journal of Chemical Physics*, 77(7):3654–3665, 1982.

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

[28] Rick A Kendall, Thom H Dunning Jr, and Robert J Harrison. Electron affinities of the first-row atoms revisited. systematic basis sets and wave functions. *The Journal of chemical physics*, 96(9):6796–6806, 1992.

[29] P Jeffrey Hay and Willard R Wadt. Ab initio effective core potentials for molecular calculations. potentials for the transition metal atoms sc to hg. *The Journal of chemical physics*, 82(1):270–283, 1985.

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

[31] Florian Weigend and Reinhart Ahlrichs. Balanced basis sets of split valence, triple zeta valence and quadruple zeta valence quality for h to rn: Design and assessment of accuracy. *Physical Chemistry Chemical Physics*, 7(18):3297–3305, 2005.

[32] Benjamin P. Pritchard, Doaa Altarawy, Brett Didier, Tara D. Gibsom, and Theresa L. Windus. A new basis set exchange: An open, up-to-date resource for the molecular sciences community. *J. Chem. Inf. Model.*, 59:4814–4820, 2019. doi: 10.1021/acs.jcim.9b00725.

[33] Qiming Sun, Xing Zhang, Samragni Banerjee, Peng Bao, Marc Barbry, Nick S. Blunt, Nikolay A. Bogdanov, George H. Booth, Jia Chen, Zhi-Hao Cui, Janus J. Eriksen, Yang Gao, Sheng Guo, Jan Hermann, Matthew R. Hermes, Kevin Koh, Peter Koval, Susi Lehtola, Zhendong Li, Junzi Liu, Narbe Mardirossian, James D. McClain, Mario Motta, Bastien Mussard, Hung Q. Pham, Artem Pulkin, Wirawan Purwanto, Paul J. Robinson, Enrico Ronca, Elvira R. Sayfutyarova, Maximilian Scheurer, Henry F. Schurkus, James E. T. Smith, Chong Sun, Shi-Ning Sun, Shiv Upadhyay, Lucas K. Wagner, Xiao Wang, Alec White, James Daniel Whitfield, Mark J. Williamson, Sebastian Wouters, Jun Yang, Jason M. Yu, Tianyu Zhu, Timothy C. Berkelbach, Sandeep Sharma, Alexander Yu. Sokolov, and Garnet Kin-Lic Chan. Recent developments in the PySCF program package. 153(2):024109, . ISSN 0021-9606. doi: 10.1063/5.0006074. URL `https://doi.org/10.1063/5.0006074`.

[34] 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, Sebastian Wouters, and Garnet Kin-Lic Chan. PySCF: the python-based simulations of chemistry framework. 8(1):e1340, . ISSN 1759-0884. doi: 10.1002/wcms.1340. URL

https://onlinelibrary.wiley.com/doi/abs/10.1002/wcms.1340. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/wcms.1340.

[35] Qiming Sun. Libcint: An efficient general integral library for gaussian basis functions. 36(22):1664–1671. ISSN 1096-987X. doi: 10.1002/jcc.23981. URL https://onlinelibrary.wiley.com/doi/abs/10.1002/jcc.23981. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/jcc.23981.

[36] John C Slater. Atomic shielding constants. *Physical review*, 36(1):57, 1930.

[37] Robert S Mulliken. Electronic population analysis on lcao–mo molecular wave functions. i. *The Journal of chemical physics*, 23(10):1833–1840, 1955.

[38] Hans L Skriver. *The LMTO method: muffin-tin orbitals and electronic structure*, volume 41. Springer Science & Business Media, 2012.

[39] José M Soler, Emilio Artacho, Julian D Gale, Alberto García, Javier Junquera, Pablo Ordejón, and Daniel Sánchez-Portal. The siesta method for ab initio order-n materials simulation. *Journal of Physics: Condensed Matter*, 14(11): 2745, 2002.

[40] Jürg Hutter, Marcella Iannuzzi, Florian Schiffmann, and Joost VandeVondele. cp2k: atomistic simulations of condensed matter systems. *Wiley Interdisciplinary Reviews: Computational Molecular Science*, 4(1):15–25, 2014.

[41] DR Bowler, R Choudhury, MJ Gillan, and T Miyazaki. Recent progress with large-scale ab initio calculations: the conquest code. *physica status solidi (b)*, 243(5):989–1000, 2006.

[42] Chris-Kriton Skylaris, Peter D Haynes, Arash A Mostofi, and Mike C Payne. Introducing onetep: Linear-scaling density functional simulations on parallel computers. *The Journal of chemical physics*, 122(8), 2005.

[43] 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.

[44] Enrico Fermi. Statistical method to determine some properties of atoms. *Rend. Accad. Naz. Lincei*, 6(602-607):5, 1927.

[45] CF v Weizsäcker. Zur theorie der kernmassen. *Zeitschrift für Physik*, 96(7): 431–458, 1935.

[46] John P Perdew. Generalized gradient approximation for the fermion kinetic energy as a functional of the density. *Physics Letters A*, 165(1):79–82, 1992.

[47] Lucian A. Constantin, E. Fabiano, S. Laricchia, and F. Della Sala. Semiclassical neutral atom as a reference system in density functional theory. 106(18):186406. doi: 10.1103/PhysRevLett.106.186406. URL `https://link.aps.org/doi/10.1103/PhysRevLett.106.186406`. Publisher: American Physical Society.

[48] Kai Luo, Valentin V Karasiev, and SB Trickey. A simple generalized gradient approximation for the noninteracting kinetic energy density functional. *Physical Review B*, 98(4):041111, 2018.

[49] Houlong Zhuang, Mohan Chen, and Emily A Carter. Elastic and thermodynamic properties of complex mg-al intermetallic compounds via orbital-free density functional theory. *Physical Review Applied*, 5(6):064021, 2016.

[50] YH Ding, Alexander James White, SX Hu, Ondrej Certik, and Lee A Collins. Ab initio studies on the stopping power of warm dense matter with time-dependent orbital-free density functional theory. *Physical Review Letters*, 121(14):145001, 2018.

[51] 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.

[52] R Ahlrichs, M Bär, HP Baron, R Bauernschmitt, S Böcker, M Ehrig, K Eichkorn, S Elliott, F Furche, F Haase, et al. Turbomole version 5, theoretical chemistry group, university of karlsruhe, 2002;(b) o. treutler, r. ahlrichs. *J. Chem. Phys*, 102:346, 1995.

[53] Vyacheslav Ivanovich Lebedev. Quadratures on a sphere. *USSR Computational Mathematics and Mathematical Physics*, 16(2):10–24, 1976.

[54] Jerry L Whitten. Coulombic potential energy integrals and approximations. *The Journal of Chemical Physics*, 58(10):4496–4501, 1973.

[55] Brett I Dunlap, JWD Connolly, and JR Sabin. On some approximations in applications of xα theory. *The Journal of Chemical Physics*, 71(8):3396–3402, 1979.

[56] O Vahtras, J Almlöf, and MW Feyereisen. Integral approximations for lcao-scf calculations. *Chemical Physics Letters*, 213(5-6):514–518, 1993.

[57] Alexandre Duval, Simon V Mathis, Chaitanya K Joshi, Victor Schmidt, Santiago Miret, Fragkiskos D Malliaros, Taco Cohen, Pietro Lio, Yoshua Bengio, and Michael Bronstein. A hitchhiker's guide to geometric gnns for 3d atomic systems. *arXiv preprint arXiv:2312.07511*, 2023.

[58] Christoph Goller and Andreas Kuchler. Learning task-dependent distributed representations by backpropagation through structure. In *Proceedings of international conference on neural networks (ICNN'96)*, volume 1, pages 347–352. IEEE, 1996.

[59] Franco Scarselli, Marco Gori, Ah Chung Tsoi, Markus Hagenbuchner, and Gabriele Monfardini. The graph neural network model. *IEEE transactions on neural networks*, 20(1):61–80, 2008.

[60] A Vaswani. Attention is all you need. *Advances in Neural Information Processing Systems*, 2017.

[61] 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? URL `https://openreview.net/forum?id=OeWooOxFwDa`.

[62] Yi-Lun Liao, Brandon Wood, Abhishek Das, and Tess Smidt. EquiformerV2: Improved equivariant transformer for scaling to higher-degree representations. URL `http://arxiv.org/abs/2306.12059`.

[63] Simon Batzner, Albert Musaelian, Lixin Sun, Mario Geiger, Jonathan P Mailoa, Mordechai Kornbluth, Nicola Molinari, Tess E Smidt, and Boris Kozinsky. E (3)-equivariant graph neural networks for data-efficient and accurate interatomic potentials. *Nature communications*, 13(1):2453, 2022.

[64] Maurice Weiler, Fred A. Hamprecht, and Martin Storath. Learning steerable filters for rotation equivariant CNNs. pages 849–858. URL `https://openaccess.thecvf.com/content_cvpr_2018/html/Weiler_Learning_Steerable_Filters_CVPR_2018_paper.html`.

[65] Nathaniel Thomas, Tess Smidt, Steven Kearnes, Lusann Yang, Li Li, Kai Kohlhoff, and Patrick Riley. Tensor field networks: Rotation-and translation-equivariant neural networks for 3d point clouds. *arXiv preprint arXiv:1802.08219*, 2018.

[66] Maurice Weiler, Mario Geiger, Max Welling, Wouter Boomsma, and Taco S Cohen. 3d steerable cnns: Learning rotationally equivariant features in volumetric data. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL `https://proceedings.neurips.cc/paper_files/paper/2018/file/488e4104520c6aab692863cc1dba45af-Paper.pdf`.

[67] Mario Geiger and Tess Smidt. e3nn: Euclidean neural networks. *arXiv preprint arXiv:2207.09453*, 2022.

[68] Peter Lippmann, Gerrit Gerhartz, Roman Remme, and Fred A Hamprecht. Tensor frames–how to make any message passing network equivariant. *arXiv preprint arXiv:2405.15389*, 2024.

[69] Nadav Dym, Hannah Lawrence, and Jonathan W Siegel. Equivariant frames and the impossibility of continuous canonicalization. *arXiv preprint arXiv:2402.16077*, 2024.

[70] 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. *The Journal of Chemical Physics*, 159(14), 2023.

[71] Robert G Parr and Weitao Yang. Density-functional theory of the electronic structure of molecules. *Annual review of physical chemistry*, 46(1):701–728, 1995.

[72] James Kirkpatrick, Brendan McMorrow, David H. P. Turban, Alexander L. Gaunt, James S. Spencer, Alexander G. D. G. Matthews, Annette Obika, Louis Thiry, Meire Fortunato, David Pfau, Lara Román Castellanos, Stig Petersen, Alexander W. R. Nelson, Pushmeet Kohli, Paula Mori-Sánchez, Demis Hassabis, and Aron J. Cohen. Pushing the frontiers of density functionals by solving the fractional electron problem. *Science*, 374(6573):1385–1389, Dec 2021. doi: 10.1126/science.abj6511.

[73] 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.

[74] Kyle Bystrom and Boris Kozinsky. Cider: An expressive, nonlocal feature set for machine learning density functionals with exact constraints. *Journal of Chemical Theory and Computation*, 18(4):2180–2192, 2022.

[75] Sebastian Dick and Marivi Fernandez-Serra. Machine learning accurate exchange and correlation functionals of the electronic density. *Nature communications*, 11(1):1–10, 2020.

[76] Narbe Mardirossian and Martin Head-Gordon. ωb97x-v: A 10-parameter, range-separated hybrid, generalized gradient approximation density functional with nonlocal correlation, designed by a survival-of-the-fittest strategy. *Physical Chemistry Chemical Physics*, 16(21):9904–9924, 2014.

[77] Jonathan Schmidt, Carlos L Benavides-Riveros, and Miguel AL Marques. Machine learning the physical nonlocal exchange–correlation functional of density-functional theory. *The journal of physical chemistry letters*, 10(20):6425–6431, 2019.

[78] 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.

[79] H Saidaoui, S Kais, S Rashkeev, and FH Alharbi. Direct scheme calculation of the kinetic energy functional derivative using machine learning. *arXiv preprint arXiv:2003.00876*, 2020.

[80] S Alireza Ghasemi and Thomas D Kühne. Artificial neural networks for the kinetic energy functional of non-interacting fermions. *The Journal of Chemical Physics*, 154(7):074107, 2021.

[81] 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): 241705, 2018.

[82] 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.

[83] Andrea Grisafi, David M Wilkins, Gábor Csányi, and Michele Ceriotti. Symmetry-adapted machine learning for tensorial properties of atomistic systems. *Physical review letters*, 120(3):036002, 2018.

[84] Maurice Weiler, Mario Geiger, Max Welling, Wouter Boomsma, and Taco S Cohen. 3d steerable cnns: Learning rotationally equivariant features in volumetric data. *Advances in Neural Information Processing Systems*, 31, 2018.

[85] Axel D Becke. Density-functional exchange-energy approximation with correct asymptotic behavior. *Physical review A*, 38(6):3098, 1988.

[86] Chengteh Lee, Weitao Yang, and Robert G Parr. Development of the colle-salvetti correlation-energy formula into a functional of the electron density. *Physical review B*, 37(2):785, 1988.

[87] Thom H. Dunning. Gaussian basis sets for use in correlated molecular calculations. i. the atoms boron through neon and hydrogen. *J. Chem. Phys.*, 90: 1007–1023, 1989. doi: 10.1063/1.456153.

[88] David E. Woon and Thom H. Dunning. Gaussian basis sets for use in correlated molecular calculations. iv. calculation of static electrical response properties. *J. Chem. Phys.*, 100:2975–2988, 1994. doi: 10.1063/1.466439.

[89] Rollin A King and Nicholas C Handy. Kinetic energy functionals from the kohn–sham potential. *Physical Chemistry Chemical Physics*, 2(22):5049–5056, 2000.

[90] 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.

[91] Qiming Sun, Xing Zhang, Samragni Banerjee, Peng Bao, Marc Barbry, Nick S Blunt, Nikolay A Bogdanov, George H Booth, Jia Chen, Zhi-Hao Cui, et al. Recent developments in the pyscf program package. *The Journal of chemical physics*, 153(2):024109, 2020.

[92] Garnet Kin-Lic Chan, Aron J. Cohen, and Nicholas C. Handy. Thomas–Fermi–Dirac–von Weizsäcker models in finite systems. *J. Chem. Phys.*, 114(2):631, 2001. ISSN 00219606. doi: 10.1063/1.1321308. URL `https://pubs.aip.org/aip/jcp/article/114/2/631-638/184186`.

[93] Matthew S. Ryley, Michael Withnall, Tom J. P. Irons, Trygve Helgaker, and Andrew M. Teale. Robust all-electron optimization in orbital-free density-functional theory using the trust-region image method. *J. Phys. Chem. A*,

125(1):459–475, 2021. ISSN 1089-5639. doi: 10.1021/acs.jpca.0c09502. URL https://doi.org/10.1021/acs.jpca.0c09502. Publisher: American Chemical Society.

[94] D. Kraft. *A Software Package for Sequential Quadratic Programming*. Deutsche Forschungs- und Versuchsanstalt für Luft- und Raumfahrt Köln: Forschungsbericht. Wiss. Berichtswesen d. DFVLR, 1988.

[95] Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, C J Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E. A. Quintero, Charles R. Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272, 2020. doi: 10.1038/s41592-019-0686-2.

[96] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[97] Eric Cancès, Yvon Maday, and Benjamin Stamm. Domain decomposition for implicit solvation models. *The Journal of chemical physics*, 139(5):054111, 2013.

[98] Filippo Lipparini, Benjamin Stamm, Eric Cances, Yvon Maday, and Benedetta Mennucci. Fast domain decomposition algorithm for continuum solvation models: Energy and first derivatives. *Journal of chemical theory and computation*, 9(8):3637–3648, 2013.

[99] Filippo Lipparini, Giovanni Scalmani, Louis Lagardère, Benjamin Stamm, Eric Cancès, Yvon Maday, Jean-Philip Piquemal, Michael J Frisch, and Benedetta Mennucci. Quantum, classical, and hybrid qm/mm calculations in solution: General implementation of the ddcosmo linear scaling strategy. *The Journal of chemical physics*, 141(18):184108, 2014.

[100] 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.

[101] 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.

[102] Raghunathan Ramakrishnan, Pavlo O. Dral, Matthias Rupp, and O. Anatole von Lilienfeld. Quantum chemistry structures and properties of 134 kilo molecules. 1(1):140022. ISSN 2052-4463. doi: 10.1038/sdata.2014.22. URL `https://www.nature.com/articles/sdata201422`. Publisher: Nature Publishing Group.

[103] Lars Ruddigkeit, Ruud van Deursen, Lorenz C. Blum, and Jean-Louis Reymond. Enumeration of 166 billion organic small molecules in the chemical universe database GDB-17. 52(11):2864–2875. ISSN 1549-9596. doi: 10.1021/ci300415d. URL `https://doi.org/10.1021/ci300415d`. Publisher: American Chemical Society.

[104] P. Pulay. Ab initio calculation of force constants and equilibrium geometries in polyatomic molecules. 17(2):197–204, . ISSN 0026-8976. doi: 10.1080/00268976900100941. URL `https://doi.org/10.1080/00268976900100941`.

[105] R. Ditchfield, W. J. Hehre, and J. A. Pople. Self-consistent molecular-orbital methods. IX. an extended gaussian-type basis for molecular-orbital studies of organic molecules. 54(2):724–728. ISSN 0021-9606. doi: 10.1063/1.1674902. URL `https://doi.org/10.1063/1.1674902`.

[106] Michael J. Frisch, John A. Pople, and J. Stephen Binkley. Self-consistent molecular orbital methods 25. supplementary functions for gaussian basis sets. 80(7):3265–3269. ISSN 0021-9606. doi: 10.1063/1.447079. URL `https://doi.org/10.1063/1.447079`.

[107] W. J. Hehre, R. Ditchfield, and J. A. Pople. Self—consistent molecular orbital methods. XII. further extensions of gaussian—type basis sets for use in molecular orbital studies of organic molecules. 56(5):2257–2261. ISSN 0021-9606. doi: 10.1063/1.1677527. URL `https://doi.org/10.1063/1.1677527`.

[108] R. Krishnan, J. S. Binkley, R. Seeger, and J. A. Pople. Self-consistent molecular orbital methods. XX. a basis set for correlated wave functions. 72:650–654. ISSN 0021-9606. doi: 10.1063/1.438955. URL `https://ui.adsabs.harvard.edu/abs/1980JChPh..72..650K`. Publisher: AIP ADS Bibcode: 1980JChPh..72..650K.

[109] Per-Olov Löwdin. On the non-orthogonality problem connected with the use of atomic wave functions in the theory of molecules and crystals. 18(3):365–375. ISSN 0021-9606. doi: 10.1063/1.1747632. URL https://doi.org/10.1063/1.1747632.

[110] Matthias Fey and Jan E. Lenssen. Fast graph representation learning with PyTorch Geometric. In *ICLR Workshop on Representation Learning on Graphs and Manifolds*, 2019.

[111] William Falcon and The PyTorch Lightning team. PyTorch Lightning, March 2019. URL https://github.com/Lightning-AI/lightning.

[112] Ilya Loshchilov and Frank Hutter. SGDR: Stochastic gradient descent with warm restarts. In *International Conference on Learning Representations*, 2017.

[113] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. URL http://arxiv.org/abs/1711.05101.

[114] Ilya Sutskever, James Martens, George Dahl, and Geoffrey Hinton. On the importance of initialization and momentum in deep learning. In *Proceedings of the 30th International Conference on Machine Learning*, pages 1139–1147. PMLR. URL https://proceedings.mlr.press/v28/sutskever13.html. ISSN: 1938-7228.

[115] Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc. URL https://proceedings.neurips.cc/paper/2019/hash/bdbca288fee7f92f2bfa9f7012727740-Abstract.html.

[116] Clemens Isert, Kenneth Atz, José Jiménez-Luna, and Gisbert Schneider. QMugs, quantum mechanical properties of drug-like molecules. 9(1):273. ISSN 2052-4463. doi: 10.1038/s41597-022-01390-7. URL https://www.nature.com/articles/s41597-022-01390-7. Publisher: Nature Publishing Group.

[117] 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. 14(1):579. ISSN 2041-1723.

doi: 10.1038/s41467-023-36329-y. URL `https://www.nature.com/articles/s41467-023-36329-y`. Publisher: Nature Publishing Group.

[118] Yann LeCun, Sumit Chopra, Raia Hadsell, M Ranzato, Fujie Huang, et al. A tutorial on energy-based learning. *Predicting structured data*, 1(0), 2006.

[119] Constantin Pape, Roman Remme, Adrian Wolny, Sylvia Olberg, Steffen Wolf, Lorenzo Cerrone, Mirko Cortese, Severina Klaus, Bojana Lucic, Stephanie Ullrich, et al. Microscopy-based assay for semi-quantitative detection of SARS-CoV-2 specific antibodies in human sera: A semi-quantitative, high throughput, microscopy-based assay expands existing approaches to measure SARS-CoV-2 specific antibody levels in human sera. *Bioessays*, 43(3):2000257, 2021.

[120] Burkhard Tönshoff, Barbara Müller, Roland Elling, Hanna Renk, Peter Meissner, Hartmut Hengel, Sven F Garbade, Meinhard Kieser, Kathrin Jeltsch, Jürgen Grulich-Henn, et al. Prevalence of SARS-CoV-2 infection in children and their parents in southwest Germany. *JAMA pediatrics*, 175(6):586–593, 2021.

[121] Henning Arlt, Xuewu Sui, Brayden Folger, Carson Adams, Xiao Chen, Roman Remme, Fred A Hamprecht, Frank DiMaio, Maofu Liao, Joel M Goodman, et al. Seipin forms a flexible cage at lipid droplet formation sites. *Nature structural & molecular biology*, 29(3):194–202, 2022.