universität wien

DISSERTATION | DOCTORAL THESIS

Titel | Title

Transferable Neural Network Wavefunctions

verfasst von | submitted by

Dipl.-Ing. Michael Scherbela BSc

angestrebter akademischer Grad | in partial fulfilment of the requirements for the degree of

Doktor der Naturwissenschaften (Dr.rer.nat.)

Wien | Vienna, 2024

Studienkennzahl lt. Studienblatt | Degree programme code as it appears on the student record sheet:	UA 796 605 405
Dissertationsgebiet lt. Studienblatt | Field of study as it appears on the student record sheet:	Mathematik
Betreut von | Supervisor:	Univ.-Prof. Dr. Philipp Grohs

Zusammenfassung

Wellenfunktionen, die Lösungen der Schrödingergleichung, ermöglichen im Prinzip die Berechnung einer Vielzahl an Eigenschaften beliebiger Materialien. Für die Entdeckung und Erforschung neuer Materialien – wie zum Beispiel Supraleitern, Katalysatoren oder Medikamenten – ist die effiziente Berechnung von Wellenfunktionen daher von großer Bedeutung. Dies ist herausfordernd, da Wellenfunktionen hochdimensionale Objekte sind und zusätzlich hohe Genauigkeiten für praktische Anwendungen notwendig sind. Ein vor wenigen Jahren vorgeschlagener Ansatz ist die Kombination von Variational Monte Carlo (VMC) mit künstlichen neuronalen Netzen. Bei dieser Methode wird die Wellenfunktion durch ein neuronales Netz approximiert, die zugehörige Energie durch Monte Carlo Integration geschätzt und anschließend mittels Gradientenverfahren minimiert. Dank der hohen Expressivität neuronaler Netze erzielt diese Methode herausragende Genauigkeit bei gleichzeitig moderater Abhängigkeit der Rechenzeit von der Systemgröße.

In praktischen Anwendungen, welche oftmals Lösungen der Schrödingergleichung für viele unterschiedliche Moleküle oder Geometrien benötigen, ist dieser Ansatz jedoch zu rechenaufwändig, da für jedes neue System die Wellenfunktion neu optimiert werden muss. Diese Arbeit beschreibt tranferierbare Wellenfunktionen auf Basis neuronaler Netze. Sie ermöglichen die gemeinsame Lösung der Schrödingergleichung für eine Vielzahl an Systemen, sowie den Transfer einer solchen Lösung auf neue Systeme. Die Dissertation gibt einen kurzen Überblick über VMC mit neuronal Netzen und enthält eine Reihe an Publikationen, welche schrittweise transferierbare Wellenfunktionen einführen. Die Publikationen beginnen mit dem Fall einer einzelnen Geometrie, dann unterschiedlichen Geometrien desselben Moleküls und schlussendlich einer Wellenfunktionen welche auf einem Datensatz unterschiedlichster organischer Moleküle trainiert werden kann.

Abstract

Solutions to the Schrödinger Equation, referred to as wavefunctions, allow in principle to predict any property of any molecule or material. Finding solutions efficiently is therefore crucial to computational discovery and understanding of new materials such as drugs, catalysts or superconductors. However due to the large dimensionality of the problem and the high accuracy required for practical applications, designing efficient methods to find approximate solutions is challenging. One recently proposed method is to combine Variational Monte Carlo (VMC) with neural network wavefunctions: This method represents the wavefunction as a neural network, estimates the corresponding energy of the ansatz via Monte Carlo integration, and minimizes this energy via gradient based optimization. Due to the high expressivity of neural networks this ansatz achieves exceptionally high accuracy with moderate scaling of computational costs.

However for practical applications, which often require solutions for many different molecules or geometries, this approach often proves unfeasible since it necessitates an expensive optimization for every new system. This work introduces transferable neural network wavefunctions, capable of simultaneously representing wavefunctions for many distinct molecules. Transferable neural network wavefunctions yield solutions to the Schrödinger Equation for many systems in parallel and even allow to use wavefunctions optimized on a set of training molecules to be applied to new, previously unseen systems. This thesis provides a brief introduction to the field of neural network wavefunction VMC and contains several related publications. These successively builds towards a transferable neural network wavefunction - starting from the single molecule case, via distinct geometries of a single molecule, towards a fully end-to-end machine learned neural network wavefunction applied to a diverse dataset of organic molecules.

To my parents, who supported me throughout this journey.
To my greatest love: Sonja and Simon.

Contents

Chapter 1

Introduction to Neural Network Wavefunctions

1.1 Motivation

Molecules and materials are ubiquitous in our daily lives. Drugs, polymers, semiconductors, catalysts – they are all carefully engineered materials with specific properties that fill important roles in our modern societies. Understanding existing materials and developing new ones is thus of vital importance for future progress. For the last centuries, the discovery of new materials has primarily been driven by experiments, but in the last decades computational physics has emerged as a powerful tool to study materials in silico. This has in principle the potential to lower cost and enable the search for new materials on a much great scale than ever before.

The key challenge for computational methods has already been recognized and succinctly expressed by Paul Dirac in 1929 [1]:

> The underlying physical laws necessary for the mathematical theory of a large part of physics and the whole of chemistry are thus completely known, and the difficulty is only that the exact application of these laws leads to equations much too complicated to be soluble. It therefore becomes desirable that approximate practical methods of applying quantum mechanics should be developed, which can lead to an explanation of the main features of complex atomic systems without too much computation. (Paul Dirac)

On the one hand, except for some typically minor approximations (see section 1.3.1), the Schrödinger Equation (a linear PDE) and its solutions yield in principle a full description of any property of any material. On the other hand, solving the equation – i.e. finding a the ground-state wavefunction for a given molecule – is computationally challenging. Analytical solutions are only known for atoms with a single electron (i.e. a single Hydrogen atom) and thus any system of practical interest must be solved numerically. It has furthermore been shown that for model-Hamiltonians such as the Hubbard model, finding the ground-state wavefunction is a QMA-hard problem [2], making it at least as hard as any NP-complete problem. Given the importance of the problem, a plethora of methods have been developed over the last decades to solve the Schrödinger Equation approximately. Some methods such as the Hartree-Fock (HF)

method are computationally cheap and scale well with system size, but are only accurate for a limited class of systems. Other methods such as Configuration Interaction (CI) and Coupled Cluster (CC) often yield highly-accurate results that are in good agreement with experiments, but scale poorly with system size and are thus limited to small systems. Density Functional Theory (DFT) has emerged as a breakthrough, and has been awarded the Nobel Prize in chemistry in 1998, since it yields surprisingly high accuracy or while scaling well with system size [3]. However, DFT requires the use of an essentially uncontrolled approximation, which can fail for many systems of interest and is thus not universally applicable.

This dissertation and the included papers are concerned with the advancement of an emerging method which might one day be able to combine highly accurate solutions with moderate scaling of computational cost. It is based on combining the well-established method of Variational Monte Carlo (VMC) with the recent advances in Deep-Learning.

1.2 Notation

Throughout this introduction, the following notation is used: Vectors and higher-dimensional tensors are denoted with bold-face symbols. Functions use non-bold symbols irrespective of their output dimension. When the same symbol is used with indices it refers to scalar elements of this tensor or slices along the leading dimensions. For example the tensor $\boldsymbol{A} \in \mathbb{R}^{a \times b \times c}$ is a tensor, $A_{ijk} \in \mathbb{R}$ refers to a scalar element of $\boldsymbol{A}$ and $\boldsymbol{A}_i \in \mathbb{R}^{b \times c}$ refers to a matrix-shaped slice of the tensor $\boldsymbol{A}$.

The operator $\odot$ denotes element wise multiplication of two vectors of identical shapes. For two tensors $\boldsymbol{a} \in \mathbb{R}^{d_1 \times \cdots \times d_a}$ and $\boldsymbol{b} \in \mathbb{R}^{d_1 \times \cdots \times d_b}$, the operation $\boldsymbol{c} = [\boldsymbol{a}|\boldsymbol{b}]$ denotes concatenation of $\boldsymbol{a}$ and $\boldsymbol{b}$ along their last dimension, generally corresponding to some feature-dimension. The resulting tensor $\boldsymbol{c}$ is thus in $\mathbb{R}^{d_1 \times \cdots \times (d_a + d_b)}$.

Indices I, J will generally be used for nuclei, indices i, j, k will generally be used for electrons or orbitals. The notation $j \in \uparrow$ refers to the set of all electron-indices belonging to up-electrons, i.e. $j = 1 \dots n_\uparrow$. Correspondingly $j \in \downarrow$ refers to all spin-down indices, i.e. $j = n_\uparrow + 1 \dots n_{\text{el}}$.

Trainable parameters are denoted with θ.

1.3 The Schrödinger Equation

1.3.1 The Born-Oppenheimer approximation

Throughout this work, we will consider the time-independent Schrödinger equation in the Born-Oppenheimer approximation. A molecule is comprised of heavy nuclei and much lighter electrons. In the Born-Oppenheimer approximation, a molecule is fully specified by the positions $\boldsymbol{R} \in \mathbb{R}^{N_{\text{nuc}} \times 3}$ of the N_{nuc} nuclei, the charges of these nuclei $\boldsymbol{Z} \in \mathbb{N}^{N_{\text{nuc}}}$ and the number of electrons n_{el}. While the positions of the nuclei are given, the electrons do not have deterministic positions $\boldsymbol{r} \in \mathbb{R}^{n_{\text{el}} \times 3}$. Instead the electron positions are randomly distributed according to a probability distribution $p(\boldsymbol{r})$,

$$\boldsymbol{r} \sim p(\boldsymbol{r}) = |\psi(\boldsymbol{r})|^2. \tag{1.1}$$

The function $\psi : \mathbb{R}^{n_{\text{el}} \times 3} \to \mathbb{C}$, the square of which yields the probability distribution, is the so-called wavefunction. The wavefunction ψ is a function of all electron coordinates and its squared absolute value describes the probability of finding the electrons at given positions. Since

$|\psi|^2$ is a probability density, it follows that it must be normalized:

$$\int |\psi|^2 d\boldsymbol{r} = 1, \tag{1.2}$$

leading to the boundary condition that $\psi \to 0$ for any electron coordinate $r_{i\alpha} \to \infty$.

The wavefunction is specified as the solution to the following eigenvalue problem

$$H(\boldsymbol{R}, \boldsymbol{Z})\psi = E\psi, \tag{1.3}$$

where H is the Hamiltonian operator, which depends on the molecule, and the eigenvalues $E \in \mathbb{R}$ corresponds to the energy levels of the system. For a molecule with N_{nuc} nuclei and n_{el} electrons, the Hamiltonian H is given by

$$H = H^{\text{kin}} + H^{\text{el-el}} + H^{\text{el-nuc}} + H^{\text{nuc-nuc}} \tag{1.4}$$

$$H^{\text{kin}} = -\frac{1}{2}\sum_{i=1}^{n_{\text{el}}} \nabla_i^2 = -\frac{1}{2}\sum_{i=1}^{n_{\text{el}}}\sum_{\alpha=1}^{3} \frac{\partial^2}{\partial r_{i\alpha}^2} \tag{1.5}$$

$$H^{\text{el-el}} = \sum_{i=1}^{n_{\text{el}}}\sum_{j=i+1}^{n_{\text{el}}} \frac{1}{|\boldsymbol{r}_i - \boldsymbol{r}_j|} \tag{1.6}$$

$$H^{\text{el-nuc}} = -\sum_{i=1}^{n_{\text{el}}}\sum_{I=1}^{N_{\text{nuc}}} \frac{Z_I}{|\boldsymbol{r}_i - \boldsymbol{R}_I|} \tag{1.7}$$

$$H^{\text{nuc-nuc}} = \sum_{I=1}^{N_{\text{nuc}}}\sum_{J=I+1}^{N_{\text{nuc}}} \frac{Z_I Z_J}{|\boldsymbol{R}_I - \boldsymbol{R}_J|}. \tag{1.8}$$

The term H^{kin}, containing second derivatives with respect to all electron coordinates, corresponds to the kinetic energy of the electrons. The remaining terms correspond to the Coulomb interaction between all particles. Because H is a hermitian operator, its eigenvalues are real and its eigenfunctions form an orthonormal system, i.e.

$$\langle \psi_n, \psi_m \rangle := \int \psi_n^* \psi_m d\boldsymbol{r} = \delta_{nm}. \tag{1.9}$$

The spectrum of H typically consists of two parts: Discrete energy levels with negative eigenvalues, corresponding to bound states, and a continuum of positive eigenvalues, corresponding to unbound states.

The overall goal of solving the Schrödinger Equation is thus as follows: Given a molecule specified by the positions $\boldsymbol{R}$ and charges $\boldsymbol{Z}$ of the nuclei and the number of electrons n_{el}, find the eigenfunctions ψ_n and corresponding eigenvalues E_n of eq. (1.3).

Besides their positions, electrons have another property named spin, which is a binary variable with its values typically being referred to as spin-up or spin-down. Throughout this work the spin for each electron is assumed to be fixed and the electrons can thus be partitioned into two groups: $n_\uparrow$ spin-up electrons and $n_\downarrow$ spin-down electrons (with $n_\uparrow + n_\downarrow = n_{\text{el}}$). By convention, we choose for the electron indices $i \in [1, n_\uparrow]$ to refer to spin-up electrons and $i \in [n_\uparrow + 1, \dots n_{\text{el}}]$ to refer to the spin-down electrons. Since electrons with the same spin are indistinguishable, the probability density $|\psi|^2$ must not change under permutation of two electrons with the same spin:

$$|\psi(\boldsymbol{r}_1, \dots, \boldsymbol{r}_i, \dots \boldsymbol{r}_j, \dots \boldsymbol{r}_{n_{\text{el}}})|^2 = |\psi(\boldsymbol{r}_1, \dots, \boldsymbol{r}_j, \dots \boldsymbol{r}_i, \dots \boldsymbol{r}_{n_{\text{el}}})|^2 \tag{1.10}$$

This can be satisfied by the wavefunction ψ either being invariant under particle exchange (i.e. ψ being symmetric w.r.t to particle order) or by ψ changing its sign under particle exchange (i.e. ψ being antisymmetric w.r.t to particle order). The former particles are called bosons, the latter are called fermions:

$$\text{bosons:} \qquad \psi(\boldsymbol{r}_1, \ldots, \boldsymbol{r}_i, \ldots \boldsymbol{r}_j, \ldots \boldsymbol{r}_{n_\text{el}}) = +\psi(\boldsymbol{r}_1, \ldots, \boldsymbol{r}_j, \ldots \boldsymbol{r}_i, \ldots \boldsymbol{r}_{n_\text{el}}) \tag{1.11}$$

$$\text{fermions:} \qquad \psi(\boldsymbol{r}_1, \ldots, \boldsymbol{r}_i, \ldots \boldsymbol{r}_j, \ldots \boldsymbol{r}_{n_\text{el}}) = -\psi(\boldsymbol{r}_1, \ldots, \boldsymbol{r}_j, \ldots \boldsymbol{r}_i, \ldots \boldsymbol{r}_{n_\text{el}}) \tag{1.12}$$

Electrons are fermions and a physical wavefunctions ψ must therefore not only satisfy the Schrödinger Equation and its boundary condition, but also fulfil this antisymmetry property.

1.3.2 The variational principle

Of particular interest is the lowest eigenvalue E_0 and its corresponding eigenfunction ψ_0, which are referred to as the ground-state energy and the ground state. The physical motivation for this is that physical systems typically decay to their lowest energy state and the ground-state is thus the most likely state in which a molecule is found in nature. Given any normalized trial wavefunction ψ it holds that

$$\int \psi^* H \psi d\boldsymbol{r} \geq E_0. \tag{1.13}$$

For a rigorous proof the reader is referred to Gustafson et al. [4], while for the case of a confining potential (leading to a purely discrete spectrum) it can be shown by expanding ψ in the basis of eigenfunctions ψ_n:

$$\psi = \sum_n c_n \psi_n \qquad \text{with} \sum_n |c_n|^2 = 1 \tag{1.14}$$

$$\int \psi H \psi d\boldsymbol{r} = \sum_{nm} c_n^* c_m \int \psi_n H \psi_m d\boldsymbol{r} = \tag{1.15}$$

$$= \sum_{nm} c_n^* c_m E_m \int \psi_n \psi_m d\boldsymbol{r} = \tag{1.16}$$

$$= \sum_n |c_n|^2 E_n \geq E_0 \tag{1.17}$$

One way to find the ground-state energy is thus to choose a trial function ψ_θ, parameterized by some parameters θ and minimize the energy of this trial function

$$E_\theta = \int \psi_\theta H \psi_\theta d\boldsymbol{r}. \tag{1.18}$$

When minimizing E_θ one obtains an energy estimator which is guaranteed to be bounded by E_0 from below. When the class of functions parameterized by θ is sufficiently expressive to contain ψ_0 (or a close approximation to it), the true ground-state energy (or a close approximation to it) can be found this way.

If the ground-state is not degenerate, i.e. there is only a single eigenstate ψ_0 with eigenvalue E_0, convergence of the energy also implies convergence of the wavefunction towards this ground-state

wavefunction:

$$E_\psi = \int \psi^* H \psi \tag{1.19}$$

$$= \sum_n |c_n|^2 E_n = |c_0|^2 E_0 + \sum_{n>0} |c_n|^2 E_n \tag{1.20}$$

$$\geq |c_0|^2 E_0 + \sum_{n>0} |c_n|^2 E_1 \tag{1.21}$$

$$= |c_0|^2 E_0 + (1 - |c_0|^2) E_1 \tag{1.22}$$

$$|c_0|^2 \geq 1 - \frac{E_\psi - E_0}{E_1 - E_0} \tag{1.23}$$

This means that as the obtained energy E_ψ of the trial wavefunction approaches the ground-state energy E_0, the square of the overlap with the groundstate $|c_0|^2$ approaches 1. The relevant energy scale for this is the energy gap $E_1 - E_0$ between the ground-state and the next highest energy eigenvalue (called the first excited state).

To implement this scheme in practice three key ingredients are required:

- **Integrator**: An efficient method to evaluate the $3n_{\text{el}}$-dimensional integral in eq. (1.18). This is in general challenging, because of the integral's high dimensionality.
- **Optimizer**: A method to optimize the parameters θ of the trial wavefunction.
- **Trial wavefunction**: A trial wavefunction ψ_θ which is sufficiently expressive to closely match the groundstate wavefunctions of the molecule.

Due to the high dimensionality of the wavefunction, integration is typically done using Monte Carlo integration as outlined in section 1.7 and the entire method is therefore named Variatonal Monte Carlo (VMC). Optimization is typically done using gradient-based methods as outline in section 1.8. For the trial wavefunction, historically purpose-built ansatze containing only a small number of parameters have been used, but recently deep-learning-based ansatze have emerged. These deep-learning based ansatze (also referred to as neural wavefunctions) are attractive because neural networks are efficient universal function approximators and can efficiently express high-dimensional functions. Section 1.5 gives a brief overview of recent developments in the field of neural wavefunctions and section 1.6 discusses ansatze which can be transferred across Hamiltonians of different molecules.

1.4 Conventional Mean-Field Approaches

Before diving into neural network based wavefunctions, it is instructive to briefly review conventional approaches to approximately solve the Schrödinger Equation. A plethora of such methods has been developed over the last decades and a comprehensive review of these methods is beyond the scope of this thesis. This section will focus on the Hartree-Fock (HF) method, since it serves as a starting point for deep-learning based VMC methods. For a thorough derivation and analysis of Hartree-Fock and post-HF methods, the reader is referred to [5].

1.4.1 From product ansatz to Slater determinants

If there was no electron-electron interaction $H^{\text{el-el}}$ and no antisymmetry, the Hamiltonian could be decomposed into a sum of Hamiltonians for each electron

$$H^{\text{non-interac}} = \sum_i h^1(\boldsymbol{r}_i) \tag{1.24}$$

$$h^1(\boldsymbol{r}) = -\frac{1}{2}\nabla^2_{\boldsymbol{r}} - \sum_I \frac{Z_I}{|\boldsymbol{r} - \boldsymbol{R}_I|} + \frac{1}{n_{\text{el}}} \sum_{I>J} \frac{Z_I Z_J}{|\boldsymbol{R}_I - \boldsymbol{R}_J|}, \tag{1.25}$$

which could then be solved by a product ansatz

$$\psi^{\text{non-interac}}(\boldsymbol{r}_1, \ldots, \boldsymbol{r}_{n_{\text{el}}}) = \prod_{i=1}^{n_{\text{el}}} \phi_i(\boldsymbol{r}_i) \tag{1.26}$$

$$h^1 \phi_i = \epsilon_i \phi_i \tag{1.27}$$

$$E^{\text{non-interac}} = \sum_i \epsilon_i. \tag{1.28}$$

Solving the eigenvalue problem eq. (1.27) would be much easier than finding solutions for the full Schrödinger Equations, because ϕ only depends on a single electron coordinate and is thus only a 3-dimensional function, whereas ψ is a $3 \times n_{\text{el}}$-dimensional function.

To address the initial simplifications, two changes are made to this simple product ansatz, which together are known as the Hartree-Fock (HF) method:

- The product ansatz in eq. (1.26) is explicitly antisymmetrized.
- The one-particle Hamiltonian eq. (1.25) is augmented by an additional mean-field potential which captures the *average* interaction of a particle with all other particles: $h^{\text{mean-field}} = h^1 + V^{\text{mean-field}}$.

To antisymmetrize eq. (1.26) one can sum over all possible permutations $\boldsymbol{P}$ of n_{el} electrons

$$\psi^{\text{HF}} = \frac{1}{\sqrt{n_{\text{el}}!}} \sum_{\boldsymbol{P}} \sigma(\boldsymbol{P}) \phi_{P_1}(\boldsymbol{r}_1) \phi_{P_2}(\boldsymbol{r}_2) \ldots \phi_{P_{n_{\text{el}}}}(\boldsymbol{r}_{n_{\text{el}}}), \tag{1.29}$$

where $\sigma(\boldsymbol{P}) \in \{-1, +1\}$ denotes the sign of permutation $\boldsymbol{P}$. Because this sum contains all possible permutations of coordinates with their appropriate sign, this ansatz is antisymmetric. When two particles are exchanged, one obtains the original wavefunction by swapping the two corresponding elements in each permutation $\boldsymbol{P}$. This flips the sign $\sigma(\boldsymbol{P})$ yielding the desired antisymmetry. Naively evaluating this ansatz as a sum over all permutations is computationally intractable, since it involves a sum over $n_{\text{el}}!$ terms. This ansatz is however equivalent to the following determinant

$$\psi^{\text{HF}} = \frac{1}{\sqrt{n_{\text{el}}!}} \begin{vmatrix} \phi_1(\boldsymbol{r}_1) & \phi_1(\boldsymbol{r}_2) & \ldots & \phi_1(\boldsymbol{r}_{n_{\text{el}}}) \\ \phi_2(\boldsymbol{r}_1) & \phi_2(\boldsymbol{r}_2) & \ldots & \phi_2(\boldsymbol{r}_{n_{\text{el}}}) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_{n_{\text{el}}}(\boldsymbol{r}_1) & \phi_{n_{\text{el}}}(\boldsymbol{r}_2) & \ldots & \phi_{n_{\text{el}}}(\boldsymbol{r}_{n_{\text{el}}}) \end{vmatrix} := \frac{1}{\sqrt{n_{\text{el}}!}} \det(\boldsymbol{\Phi}), \tag{1.30}$$

which can be evaluated efficiently in $\mathcal{O}({n_{\text{el}}}^3)$ operations using Gaussian elimination. This antisymmetric ansatz is known as a Slater determinant [6] and is in essence an antisymmetrized

version of a product ansatz. The antisymmetrization procedure outlined above is overzealous, since it enforces antisymmetry with respect to exchange of any two electrons, whereas antisymmetry should only be enforced for electrons of the same spin. This is easily remedied by the ansatz

$$\psi^{\mathrm{HF}} = \frac{1}{\sqrt{n_\uparrow! n_\downarrow!}} \begin{vmatrix} \phi_1^\uparrow(\boldsymbol{r}_1) & \dots & \phi_1^\uparrow(\boldsymbol{r}_{n_\uparrow}) \\ \vdots & \ddots & \vdots \\ \phi_{n_\uparrow}^\uparrow(\boldsymbol{r}_1) & \dots & \phi_{n_\uparrow}^\uparrow(\boldsymbol{r}_{n_\uparrow}) \end{vmatrix} \begin{vmatrix} \phi_1^\downarrow(\boldsymbol{r}_{n_\uparrow+1}) & \dots & \phi_1^\downarrow(\boldsymbol{r}_{n_{\mathrm{el}}}) \\ \vdots & \ddots & \vdots \\ \phi_{n_\downarrow}^\downarrow(\boldsymbol{r}_{n_\uparrow+1}) & \dots & \phi_{n_\uparrow}^\downarrow(\boldsymbol{r}_{n_{\mathrm{el}}}) \end{vmatrix} \tag{1.31}$$

$$= \frac{1}{\sqrt{n_\uparrow! n_\downarrow!}} \det(\boldsymbol{\Phi}^\uparrow) \det(\boldsymbol{\Phi}^\downarrow), \tag{1.32}$$

which effectively antisymmetrizes each set of electrons separately.

While using Slater determinants introduces the missing antisymmetrization, the omission of the electron-electron Coulomb repulsion in $H^{\text{non-interac}}$ still needs to be addressed. This is done by adding to h^1 the mean-field operator $V^{\text{mean-field}}$

$$V^{\text{mean-field}} = \sum_j V_j^{\text{Hartree}} + V_j^{\text{exchange}} \tag{1.33}$$

$$V_j^{\text{Hartree}}(\boldsymbol{r}) = \int \frac{|\phi_j(\boldsymbol{r'})|^2}{|\boldsymbol{r} - \boldsymbol{r'}|} \mathrm{d}\boldsymbol{r'} \tag{1.34}$$

$$(V_j^{\text{exchange}} \phi_i)(\boldsymbol{r}) = \phi_j(\boldsymbol{r}) \int \frac{\phi_j^*(\boldsymbol{r'})\phi_i(\boldsymbol{r'})}{|\boldsymbol{r} - \boldsymbol{r'}|}. \tag{1.35}$$

The term $V_j^{\text{Hartree}}(\boldsymbol{r})$ is Coulomb-like potential, but instead of depending explicitly on all electron coordinates, it is a one-particle potential which describes the repulsion of an electron by the average electron density $\sum_j |\phi_j(\boldsymbol{r'})|^2$. The term $V_j^{\text{exchange}}(\boldsymbol{r})$ arises from antisymmetrization as detailed in [5]. The resulting operator for the eigenvalue problem

$$\left[h^1 + V^{\text{mean-field}}(\{\phi_j\}_{j=1\dots n_{\mathrm{el}}})\right] \phi_i = \epsilon_i \phi_i \tag{1.36}$$

now depends on the eigenfunctions $\{\phi_j\}$ and the problem is therefore typically solved iteratively: Guess ϕ, compute the resulting potential $V^{\text{mean-field}}$, solve the eigenvalue problem to obtain an updated set of orbitals ϕ and iterate until the obtained orbitals ϕ and the resulting potential $V^{\text{mean-field}}$ are consistent with each other. This procedure is known as *self consistent field* (SCF) iteration.

The one-particle functions ϕ_i, which are obtained as the eigenfunctions of the mean Hamiltonian are called *orbitals*. Note that the orbitals ϕ in each determinant must be distinct, since the determinant would otherwise be identical to zero. The ground-state wavefunction is thus obtained by finding the n_{el} lowest eigenfunctions of the mean-field Hamiltonian and using them as orbitals in eq. (1.30). A few things to note about the Hartree-Fock solution are:

- The HF-ansatz is no longer a product ansatz (due to the antisymmetrization), and thus the wavefunction (and correspondingly the electron density) can no longer be factorized into functions depending on only a single electron. Due to antisymmetrization, the probability of finding an electron does explicitly depend on the coordinates of the other electrons, an effect known as *exchange interaction*.

- Beyond this exchange interaction the HF-ansatz does not explicitly account for electron-electron interactions and does therefore not yield the exact ground-state wavefunction. The energy difference between the actual ground-state energy and the Hartree-Fock energy is known as *correlation energy*.
- Because the electron-nucleus interaction often dominates the total energy, this missing correlation energy is often only a small fraction of the total energy E_0 (e.g. for a Nitrogen molecule $E_{\text{correlation}} \approx 0.005 E_0$), but $E_{\text{correlation}}$ is often still orders of magnitude larger than the accuracy required to predict experiments [7].

While a single determinant is insufficient to express the exact groundstate wavefunction, the approximation can be systematically improved by using a linear combination of such Slater Determinants as ansatz

$$\psi^{\text{CI}} = \sum_{d=1}^{N_{\text{det}}} \omega_d \det(\Phi_d^{\uparrow}) \det(\Phi_d^{\downarrow}), \tag{1.37}$$

with expansion coefficients ω_d. This approach is known as *Configurational Interaction* (CI).

1.4.2 Basis sets

To represent the orbitals $\phi(\boldsymbol{r})$ in numerical computations, they are typically expanded as linear combinations of basis functions $\hat{b}(\boldsymbol{r})$:

$$\phi_k(\boldsymbol{r}) = \sum_{\nu} \hat{c}_{\nu k} \hat{b}_{\nu}(\boldsymbol{r}), \tag{1.38}$$

with expansion coefficients $\hat{c}$. For molecules the most common type of basis functions are atom-centered basis functions $b(\boldsymbol{r})$, i.e.

$$\phi_k(\boldsymbol{r}) = \sum_{I=1}^{N_{\text{nuc}}} \sum_{\mu=1}^{N_{\text{basis}}} c_{I\mu k}\, b_{\mu}(\boldsymbol{r} - \boldsymbol{R_I}), \tag{1.39}$$

with expansion coefficients $\boldsymbol{c} \in \mathbb{R}^{N_{\text{nuc}} \times N_{\text{basis}} \times N_{\text{orb}}}$. Here μ enumerates the basis functions per atom and the total number of basis functions is given by $N_{\text{nuc}} \times N_{\text{basis}}$. One particularly common choice for these atom centered basis functions are Gaussian Type Orbitals (GTOs), a product of spherical harmonics Y_{lm} and a sum of gaussian functions to model the radial dependency

$$b_{\mu}(\boldsymbol{r}) = Y_{l_\mu m_\mu}\left(\frac{\boldsymbol{r}}{|\boldsymbol{r}|}\right) \sum_{n} \alpha_{\mu n} e^{-\beta_{\mu n}|\boldsymbol{r}|^2}. \tag{1.40}$$

The parameters $\alpha_{\mu n}$ and $\beta_{\mu n}$ are typically kept fixed (only the expansion coefficients $c_{I\mu k}$ are optimized) and are chosen in advance such that the functions b_μ approximate the orbitals ϕ_i of an isolated atom. Many different basis sets have been developed over the years, with their coefficients α and β being implemented in various quantum chemistry codes or available on repositories such as Basis Set Exchange [8].

When expressing the wavefunction as a single Slater Determinant, and using a fixed basis set, the only free parameters are the expansion coefficients $\boldsymbol{c}$. Minimizing the energy with respect to these parameters will yield the lowest possible Hartree-Fock energy achievable within this basis set. The resulting energy exceeds the true ground state energy and the error can be divided into two parts:

- **Basis set error**: Since the orbitals ϕ are expanded in a finite basis, ansatz may not be able to fully approximate the true Hartree-Fock solution. The energy difference between the finite basis energy and the *complete basis set limit* (CBS limit) is known as the basis set error.
- **Correlation energy**: Since a single Slater determinant cannot model arbitrary wavefunctions, there is an approximation error between the CBS Hartree-Fock energy and the true groundstate energy, known as the correlation energy.

1.5 Deep-Learning-based Ansatze for a Single Molecule

Using neural networks as an ansatz we aim to minimize both types of errors. On the one hand we use neural networks as function approximators instead of a linear basis set expansion, to minimize the associated approximation error. On the other hand we will design the wavefunction in such a way that the orbitals ϕ no longer depend on a single electron coordinate $\boldsymbol{r}_i$, but on all coordinates $\boldsymbol{r}$, thus overcoming the limitation of the Hartree-Fock ansatz.

1.5.1 Overall structure

The majority of existing neural wavefunction architectures roughly follows these steps to compute the wavefunction $\psi(\boldsymbol{r}_1, \dots, \boldsymbol{r}_{n_\text{el}})$:

1. **Input features**: Transform the raw inputs $\boldsymbol{r}_i$, $\boldsymbol{R}_I$, spins σ_i and nuclear charges Z_I, by computing simple input features. These typically include single-particle features $\boldsymbol{x}_i$, $\boldsymbol{X}_I$ for electrons and nuclei, and pairwise-features $\boldsymbol{p}_{ij}$, $\boldsymbol{P}_{iI}$ for electron-electron pairs and electron-nuclei pairs.

$$\boldsymbol{x}_i = f^\text{el}(\boldsymbol{r}_i, \sigma_i, \boldsymbol{R}, \boldsymbol{Z}) \tag{1.41}$$
$$\boldsymbol{X}_I = f^\text{nuc}(\boldsymbol{R}_I, Z_I, \boldsymbol{R}, \boldsymbol{Z}), \tag{1.42}$$
$$\boldsymbol{p}_{ij} = f^\text{el-el}(\boldsymbol{r}_i, \sigma_i, \boldsymbol{r}_j, \sigma_j) \tag{1.43}$$
$$\boldsymbol{P}_{iJ} = f^\text{el-nuc}(\boldsymbol{r}_i, \sigma_i, \boldsymbol{R}_J, Z_J) \tag{1.44}$$

2. **Embedding**: Compute a high-dimensional embedding $\boldsymbol{h}_i$ for each electron i. These embeddings not only depend on the input features of electron i but also on all other electrons j and nuclei J. The function h must be invariant under permutation of two electrons $j, k \neq i$ of the same spin. It therefore can only depend on the *multiset* $\{\boldsymbol{x}\}$ of all other spin-up (spin-down) electrons.

$$\boldsymbol{h}_i = h\left(\boldsymbol{x}_i,\ \{(\boldsymbol{x}_j, \boldsymbol{p}_{ij})\}_{j\in\uparrow},\ \{(\boldsymbol{x}_j, \boldsymbol{p}_{ij})\}_{j\in\downarrow},\ \boldsymbol{X}, \boldsymbol{P}_i\right) \tag{1.45}$$

3. **Orbitals**: Compute orbital functions $\phi_k(\boldsymbol{h_i})$ from the embeddings $\boldsymbol{h}_i$ for each orbital $k = 1, \dots, n_\text{el}$, and each determinant d.

$$\Phi_{dik} = \phi_{kd}(\boldsymbol{h}_i) \tag{1.46}$$

4. **Slater determinant**: Compute ψ as a sum of Slater determinants of these orbitals Φ_{dik}. Optionally augment it with a function J that is invariant under permutation of electrons with the same spin. This function is known as a *Jastrow factor*.

$$\psi = J\big(\{\boldsymbol{h}_i\}_{i\in\uparrow}, \{\boldsymbol{h}_i\}_{i\in\downarrow}\big) \sum_d \det(\Phi_d) \tag{1.47}$$

All functions in these 4 steps can have trainable parameters θ, which are subsequently optimized to minimize the energy of the ansatz. The motivation for this ansatz becomes apparent when expressing the Hartree-Fock wavefunction (see section 1.4.1) in this framework:

1. **Input features**: Use only relative electron-nucleus coordinates and the nuclear charges as input features
$$\boldsymbol{P}_{iJ} = \boldsymbol{r}_i - \boldsymbol{R}_{\boldsymbol{J}} \tag{1.48}$$

2. **Embedding**: Use the embeddings $\boldsymbol{h}_i$ to evaluate all atom-centered basis functions b_μ, i.e. each element of $\boldsymbol{h}_i \in \mathbb{R}^{n_{\text{el}} \times (N_{\text{nuc}} N_{\text{basis}})}$ corresponds to one basis-function. Do not consider any electron-electron interactions.
$$h_{i,(J\mu)} = b_\mu(\boldsymbol{P}_{iJ}) \tag{1.49}$$

3. **Orbitals**: Evaluate the orbitals ϕ as a linear combination of these basis functions with expansion coefficients $c_{(J\mu),k}$.
$$\Phi_{ik} = \sum_{J\mu} c_{(J\mu),k}\, h_{i,(J\mu)} \tag{1.50}$$

4. **Slater determinant**: Express the wavefunction as single Slater determinant. Do not use a Jastrow factor
$$\psi = \det(\Phi) \tag{1.51}$$

The overall framework thus provides a natural generalization of the Hartree-Fock ansatz, by allowing electron-electron interactions when computing $\boldsymbol{h}_i$, including an arbitrary permutation invariant function J, and including multiple determinants.

There are ample design choices when implementing this ansatz, but two restrictions apply:

- The embeddings $\boldsymbol{h}_i$ must be permutation equivariant functions of the electron coordinates. Swapping electron $i \leftrightarrow j$ must lead to a swap of embeddings $\boldsymbol{h}_i \leftrightarrow \boldsymbol{h}_j$, when i, j refer to electrons of the same spin. This ensures that a swap of electrons leads to a swap of columns in the Slater determinant, thus enforcing antisymmetry.
- The wavefunction must obey the boundary conditions (e.g. decay to zero for electrons far away from the nuclei). This condition is typically enforced at the level of orbitals ϕ_k.

In the following, some deep-learning building blocks as well as a few proposed choices for each of the four steps are discussed.

1.5.2 Deep-learning building blocks

Most deep-learning ansatze are obtained by combining simple, parameterized building blocks into expressive functions.

Multi layer perceptron A Multi Layer Perceptron (MLP) forms the most basic building block of most deep-learning architectures and consists of L layers, alternating between affine

transformations and elementwise non-linear functions. Given an input $\boldsymbol{x}^0 \in \mathbb{R}^{d_0}$, the output of each subsequent layer $l = 1 \dots L$ is computed as

$$a_n^l = \sum_m W_{nm}^l x_m^{l-1} + b_n^l \qquad \text{(Affine transformation)} \tag{1.52}$$

$$x_n^l = \sigma(a_n^l) \qquad \text{(Elementwise nonlinearity)} \tag{1.53}$$

$$\text{MLP}\left(\boldsymbol{x}^0\right) := \boldsymbol{x}^L, \tag{1.54}$$

with trainable weights $\boldsymbol{W}^l \in \mathbb{R}^{d_l \times d_{l-1}}$, $\boldsymbol{b}^l \in \mathbb{R}^{d_l}$ for every layer l and a nonlinear function σ, referred to as *activation function*.

Activation functions Common choices for this activation function include:

$$\sigma(x) = \text{ReLU}(x) = \begin{cases} 0 & x < 0 \\ x & x \geq 0 \end{cases} \tag{1.55}$$

$$\sigma(x) = \tanh(x) \tag{1.56}$$

$$\sigma(x) = \text{SiLU}(x) = \frac{x}{1 + e^{-x}} \tag{1.57}$$

Because the nonlinearity can limit the range of possible output values, depending on the use-case the nonlinearity is often not applied in the last layer of the MLP. Because the wavefunction must be twice differentiable (to allow evaluation of the kinetic energy $\nabla^2\psi$) w.r.t. the electron coordinates, the ReLU activation cannot be used for neural wavefunctions and a smooth variant such as SiLU must be used instead.

It can be shown that an MLP with an activation function that is not a polynomial in the limit of infinite depth L and layer widths d_l is a universal function approximator [9], i.e. any continuous function can be approximated by it.

1.5.3 Input features

Pairwise features 3D-difference vectors $\boldsymbol{r}_{ij} = \boldsymbol{r}_i - \boldsymbol{r}_j$ and distances $r_{ij} = |\boldsymbol{r}_{ij}|$ form natural inter-particle features $\boldsymbol{x}_{ij}$. While including the distance r_{ij} in addition to $\boldsymbol{r}_{ij}$ may seem redundant, it serves two purposes: First, it provides an input feature that has a discontinuous derivative when two particles coincide, in which case also the wavefunction ψ has discontinuous derivatives. Including this discontinuous input feature thus allows modelling the discontinuous wavefunction with smooth functions h, and ϕ. Second, many interactions strongly depend on the distance between particles, so that including it as a feature provides valuable information to the model in an accessible manner. Some models [7, 10, 11] use r_{ij} directly as input to the model, others encode r_{ij} [12–14] by evaluating several radial basis functions and concatenating the outputs

$$x_{ij,\nu} = \exp\left(-\frac{(r_{ij} - r_\nu)^2}{\sigma_\nu^2}\right). \tag{1.58}$$

While some architectures [7, 12] only used the inter-particle distance $|\boldsymbol{r}_{ij}|$ and did not use the corresponding difference vector $\boldsymbol{r}_{ij}$, this has been recognized to be too restrictive.

To avoid input features with excessively large values (in particular for large molecules with large inter-particle distances), Glehn et al. [15] proposed to logarithmically scale the inter-particle

distances and differences:

$$\widetilde{r}_{ij} = \log\left(1 + r_{ij}\right) \tag{1.59}$$

$$\widetilde{\boldsymbol{r}}_{ij} = \boldsymbol{r}_{ij}\frac{\widetilde{r}_{ij}}{r_{ij}} \tag{1.60}$$

$$\tag{1.61}$$

Single particle features To initialize the electron features $\boldsymbol{x}_i$, typically all electron-nuclei difference vectors are being aggregated. FermiNet [10] concatenates these features, while PESNet [11] pools a nonlinear transformation of these pairwise features to obtain invariance w.r.t. permutation of nuclei.

$$\boldsymbol{x}_i^{\text{concat}} = \left[\left(\boldsymbol{r}_i - \boldsymbol{R}_1\right) | \ldots | \left(\boldsymbol{r}_i - \boldsymbol{R}_{N_{\text{nuc}}}\right)\right] \tag{1.62}$$

$$\boldsymbol{x}_i^{\text{pooled}} = \sum_J \text{MLP}\left(\boldsymbol{r}_i - \boldsymbol{R}_J\right) \tag{1.63}$$

Nuclear charge Z is typically one-hot encoded, and spin can be encoded as a single feature with values ± 1.

1.5.4 Embedding

The role of the embedding network is to take simple input features $\boldsymbol{x}_i$ and $\boldsymbol{p}_{ij}$ and compute embeddings $\boldsymbol{h}_i$ that form good basis functions for the subsequent many-body orbitals. To do this, the embedding network must on the one hand be able to incorporate information from all other electrons j, and on the other hand be able to represent arbitrary functions of a single electron i. These two requirements are typically addressed by interleaving two kinds of computation over multiple rounds l: A function f that gathers information from other electrons and a function g that acts only on a single electron (typically implemented as single Dense layer or an MLP). Most embedding networks thus follow the following structure:

$$\boldsymbol{h}_i^0 = \boldsymbol{x}_i \qquad \text{Initialization} \tag{1.64}$$

$$\boldsymbol{m}_i^{\uparrow,l} = \sum_{j\in\uparrow} f^{\uparrow}(\boldsymbol{h}_i^{l-1}, \boldsymbol{h}_j^{l-1}, \boldsymbol{p}_{ij}) \qquad \text{Gather information across electrons} \tag{1.65}$$

$$\boldsymbol{m}_i^{\downarrow,l} = \sum_{j\in\downarrow} f^{\downarrow}(\boldsymbol{h}_i^{l-1}, \boldsymbol{h}_j^{l-1}, \boldsymbol{p}_{ij})$$

$$\boldsymbol{h}_i^l = g\left(\boldsymbol{h}_i^{l-1}, \boldsymbol{m}_i^{\uparrow,l}, \boldsymbol{m}_i^{\downarrow,l}\right) \qquad \text{Single-electron computation} \tag{1.66}$$

After iterating eq. (1.65) and eq. (1.66) for $l = 1 \ldots L$, the final embeddings are given by the output of the last layer, i.e. $\boldsymbol{h}_i = \boldsymbol{h}_i^L$. A few design considerations are worth discussing:

- The message $\boldsymbol{m}_i$ in eq. (1.65) is a sum over all particles of a given spin. Since the sum is a permutation invariant operation, the resulting message is invariant under permutation of two electrons of the same spin. Therefore the resulting embeddings $\boldsymbol{h}_i$ are permutation equivariant, i.e. swapping two electrons of the same spin, leads to a simple swap of their respective embeddings. This property is required to enforce permutation antisymmetry in the subsequent orbital construction.

- Not all parts of the network have the same impact on computational cost. While the functions $f(\boldsymbol{h}_i, \boldsymbol{h}_j, \boldsymbol{p}_{ij})$ in eq. (1.65) are evaluated for every pair of electrons – and thus have computational cost scaling as $\mathcal{O}({n_\text{el}}^2)$ – the function g in eq. (1.66) is only evaluated for every electron, thus scaling as $\mathcal{O}(n_\text{el})$. This difference in scaling is typically reflected by the fact that most architectures use wide (and thus costly) MLPs for the one-electron computations (eq. (1.66)), and computationally cheaper functions for $f^\uparrow$ and $f^\downarrow$ (eq. (1.65)).
- Some architectures differentiate between messages from up- and down-electrons (as denoted in eq. (1.65)), while others differentiate between messages from spin-parallel or spin-antiparallel electrons. The latter choice enforces invariance w.r.t. exchanging all spin-up particles with spin-down particles and has been shown to be advantageous for closed-shell systems [16].

Given this very general framework, the key difference between the various embedding architectures therefore lies in the message functions $f^\uparrow$, $f^\downarrow$, with a few common choices outlined below.

Hartree-Fock If no information from other electrons is gathered (e.g. $f^\uparrow \equiv f^\downarrow \equiv \mathbf{0}$), the final embedding $\boldsymbol{h}_i$ only depends on the input features of that electron $\boldsymbol{x}_i$. The network cannot capture any correlation effects and thus the best possible accuracy is Hartree-Fock.

FermiNet In FermiNet the message-function f simply concatenates the feature vectors of embedding $\boldsymbol{h}_j$ and an MLP of $\boldsymbol{p}_{ij}$.

$$f^\uparrow(h_i, \boldsymbol{h}_j, \boldsymbol{p}_{ij}) = \left[\boldsymbol{h}_j \mid \text{MLP}^\uparrow\left(\boldsymbol{p}_{ij}\right)\right] \tag{1.67}$$

$$f^\downarrow(h_i, \boldsymbol{h}_j, \boldsymbol{p}_{ij}) = \left[\boldsymbol{h}_j \mid \text{MLP}^\downarrow\left(\boldsymbol{p}_{ij}\right)\right] \tag{1.68}$$

FermiNet clearly improves upon a simple non-interacting embedding, by including information about all other electron embeddings as well as their relative positions in every layer. Note however that in FermiNet all electron embeddings $\boldsymbol{h}_j$ contribute equally to the message $\boldsymbol{m}_i$, irrespective of the distance between electron i and j. This runs against physical intuition, which suggests that electrons at large separations would have a smaller influence.

Graph convolutional Neural Networks In a Graph Convolutional neural network (GCN) [17], the contributions of each electron j to the message $\boldsymbol{m}_i$ are weighted by their relative geometric positions encoded in $\boldsymbol{p}_{ij}$. This weighting is commonly achieved using an elementwise product along the feature dimension:

$$f(h_i, \boldsymbol{h}_j, \boldsymbol{p}_{ij}) = \text{MLP}\left(\boldsymbol{h}_j\right) \odot \text{MLP}\left(\boldsymbol{p}_{ij}\right). \tag{1.69}$$

This allows the network in particular to put higher weight on closer neighboring electrons than electrons which are far apart. Some approaches [14] enforce this prior knowledge, by multiplying the MLP $\left(\boldsymbol{p}_{ij}\right)$ with functions that explicitly decay as a function of the distance between electrons i and j.

Self-attention based Neither in FermiNet nor the GCN embedding does the message $\boldsymbol{m}_i$ explicitly depend on the message receiver $\boldsymbol{h}_i$, but instead only depends on the message sender $\boldsymbol{h}_j$ and the pairwise features $\boldsymbol{p}_{ij}$. Self-attention is an approach where the weighting of each message j is computed as an inner product between a query vector (derived from the receiving embedding

$\boldsymbol{h}_i$) and a key vector (derived from the sending embedding $\boldsymbol{h}_j$).

$$\boldsymbol{q}_i = \boldsymbol{W}^{\mathrm{q}}\boldsymbol{h}_i \qquad \boldsymbol{k}_j = \boldsymbol{W}^{\mathrm{k}}\boldsymbol{h}_j \qquad \boldsymbol{v}_j = \boldsymbol{W}^{\mathrm{q}}\boldsymbol{h}_j \tag{1.70}$$

$$\tilde{w}_{ij} = \exp\left(\frac{\langle \boldsymbol{q}_i, \boldsymbol{k}_j\rangle}{\sqrt{d_{\mathrm{emb}}}}\right) \quad w_{ij} = \frac{\tilde{w}_{ij'}}{\sum_j \tilde{w}_{ij'}} \tag{1.71}$$

$$f(\boldsymbol{h_i}, \boldsymbol{h}_j, \boldsymbol{p}_{ij}) = w_{ij}\boldsymbol{v}_j \tag{1.72}$$

The message $\boldsymbol{m}_i$ explicitly depends on the embedding for electron i and j, but no longer explicitly depends on their pairwise features $\boldsymbol{p}_{ij}$. This geometric information must be inferred from the inner product of $\boldsymbol{q}_i$ and $\boldsymbol{k}_j$, and thus requires that the single-electron input features contain information about their absolute positions.

1.5.5 Orbitals

Given permutation equivariant embeddings $\boldsymbol{h}_i$, for each electron, the orbital part of the network computes N_{det} Slater matrices $\boldsymbol{\Phi}_d$. Each matrix $\boldsymbol{\Phi}_d$ is a square $n_{\mathrm{el}} \times n_{\mathrm{el}}$ matrix, where one dimension enumerates electrons (indexed by i) and one enumerates orbitals (indexed by k):

$$\Phi_{dik} = \phi_{kd}(\boldsymbol{h}_i). \tag{1.73}$$

The orbital function ϕ typically has the form

$$\phi_{kd} = \langle \boldsymbol{W}_{kd}, \boldsymbol{h}_i\rangle\, \widetilde{\varphi}(\boldsymbol{r}_i), \tag{1.74}$$

where $\boldsymbol{W}$ is a trainable tensor $\boldsymbol{W} \in \mathbb{R}^{N_{\mathrm{orb}} \times N_{\mathrm{det}} \times D_{\mathrm{emb}}}$, and $\widetilde{\varphi}$ is an envelope function enforcing that $\phi \to 0$ as $\boldsymbol{r}_i \to \infty$. For molecules the envelope function is typically expressed as a sum over nuclei, leading to

$$\widetilde{\varphi}_{kd}(\boldsymbol{r}_i) = \sum_{J=1}^{N_{\mathrm{nuc}}} \varphi_{kdJ}(\boldsymbol{r}_i - \boldsymbol{R}_j), \tag{1.75}$$

and putting it all together leads to

$$\Phi_{dik} = \langle \boldsymbol{W}_{kd}, \boldsymbol{h}_i\rangle \sum_{J=1}^{N_{\mathrm{nuc}}} \varphi_{kdJ}(\boldsymbol{r}_i - \boldsymbol{R}_j). \tag{1.76}$$

The most common choice for the envelope function are simple *exponential envelopes*

$$\varphi_{kdJ} = \pi_{kdJ}\, e^{-\alpha_{kdJ}|\boldsymbol{r}_i - \boldsymbol{R}_j|}, \tag{1.77}$$

with trainable parameters π_{kdJ} and α_{kdJ}. An alternative is using the single-particle orbitals from a Hartree-Fock calculation

$$\varphi_{kdJ}(\boldsymbol{r}_{iJ}) = \varphi^{\mathrm{HF}}_{kdJ}(\boldsymbol{r}_{iJ}) = \sum_{\mu=1}^{N_{\mathrm{basis}}} c_{J\mu k}\, b_\mu(\boldsymbol{r}_{iJ}), \tag{1.78}$$

where b_μ are atom-centered basis functions and $c_{J\mu k}$ are the expansion coefficients of orbital k in this basis (see section 1.4.2).

Although one might think that using Hartree-Fock orbitals as envelopes provides a useful prior and good starting point for optimization, the exponential envelopes are not only simpler to implement

but also lead to substantially more accurate results [7]. Even though using the HF-envelopes directly can decrease accuracy – and several groups that originally used them [12, 13], replaced them in later work with exponential envelopes [7, 18] – there is still information in the HF-orbitals which can be used:

First, different HF-orbitals typically have different length-scales: Some orbitals (known by chemists as core orbitals) are tightly localized on an atom, whereas other orbitals (known by chemists as valence orbitals) are somewhat delocalized. This can be quantified and used to initialize the exponents α of the exponential envelopes, using large values for α to initialize strongly localized core orbitals and small values to initialize delocalized valence electrons. Numerical experiments show that this initialization accelerates wavefunction optimization [7], in particular for heavy atoms where the length-scale of core orbitals differs by an order of magnitude from the length-scale of the valence electrons.

Second, one can use the expansion coefficients of an HF-orbital as a descriptor of that orbital. This can be useful when designing a transferable wavefunction as outlined in section 1.6.

1.5.6 Slater determinant and Jastrow Factor

The total wavefunction ψ is assembled as a determinant of the orbitals ϕ (which ensures antisymmetry) and optionally a permutation invariant function J known as a *Jastrow factor*

$$\psi = J\left(\{\boldsymbol{h}_i\}_{i\in\uparrow}, \{\boldsymbol{h}_i\}_{i\in\downarrow}\right) \sum_d \det(\boldsymbol{\Phi}_d). \tag{1.79}$$

The function J must be invariant under permutations of electrons with the same spin, to ensure that the antisymmetry enforced by the determinant is not violated. It is also common to use a Jastrow-factor that does not alter the sign of ψ, by enforcing $J > 0$ via $J = \exp(\hat{J})$.

The Jastrow factor generally serves two purposes: Increasing expressivity of the wavefunction ansatz and enforcing the Kato cusp conditions [19]. The first can be achieved by a permutation invariant pooling of the embeddings $\boldsymbol{h}_i$, e.g. as

$$J = \exp\left(\sum_{i\in\uparrow} \mathrm{MLP}\left(\boldsymbol{h}_i\right) + \sum_{i\in\downarrow} \mathrm{MLP}\left(\boldsymbol{h}_i\right)\right) \tag{1.80}$$

The latter refers to a property of the wavefunction known as cusps: The local energy $E^{\mathrm{loc}} = \frac{H\psi}{\psi}$ of the groundstate wavefunction is constant (with $E^{\mathrm{loc}} = E_0$), but the individual terms in the Hamiltonian are not. In particular the potential energy terms in eq. (1.19) diverge whenever the distance between two particles approaches zero. To obtain a constant local energy, the kinetic energy – given by the curvature of the wavefunction – must diverge with opposite sign, leading to discontinuous first derivatives of the wavefunction ψ whenever two particles coincide. These cusps of high electron density (when an electron approaches a nucleus) or low electron density (when an electron approaches another electron) can be represented by an ansatz that has discontinuous derivatives at distance $r_{ij} = 0$. A choice proposed by [12] is:

$$J = \exp\left(\sum_{i<j} \frac{a}{b + r_{ij}}\right) \tag{1.81}$$

with trainable parameters a, b.

1.6 Transferable Deep-Learning-based Ansatze

1.6.1 Motivation

So far we have discussed ansatze which can model the wavefunction for one specific system, i.e. one specific molecule in one specific geometry, i.e. a single input $\boldsymbol{R}$ and $\boldsymbol{Z}$. Many practical applications however, require energies (or other observables) not only for one specific system, but for many different molecules or geometries. One example of such applications is the evaluation of a Potential Energy Surface (PES), which corresponds to the groundstate energy as a function of nuclear coordinates $E_0(\boldsymbol{R})$. Other examples include optimizing the structure of a molecule, which corresponds to minimizing this energy as a function of nuclear coordinates $\boldsymbol{R}^{\text{opt}} = \arg\min_{\boldsymbol{R}} E_0(\boldsymbol{R})$, or compiling a dataset of high-accuracy energies for training of a downstream model. For each new system (given via $\boldsymbol{R}$, $\boldsymbol{Z}$, n_{el}), we must in principle find a new set of optimal parameters $\theta(\boldsymbol{R}, \boldsymbol{Z}, n_{\text{el}})$ to model its groundstate wavefunction. Given that optimizing the wavefunction for even just single system can require hundreds of GPU-hours, this puts many interesting applications out of computational reach. This need for repeated optimization stands in stark contrast to many other machine learning applications, where models are typically trained once (at potentially large computational cost), and subsequently allow for fast inference on new problem instances.

It is therefore desirable to find ansatze and methods which can efficiently represent not only the wavefunction for a single system, but for many different systems at once. Parameters for such a wavefunction which have been obtained on some representative systems can hopefully be *transferred* to new systems. Transferring parameters should ideally fully negate the need for optimization on the new system, or at least substantially accelerate optimization using the transferred parameters as a starting point.

Because solving the Schrödinger Equation – even for model systems such as the Hubbard model – is NP-hard [2], finding an accurate wavefunction, which is transferrable to arbitrary systems, is a near hopeless task. However, it may be possible to design and train a transferable wavefunction which is accurate for a relevant subset of systems, e.g. for organic molecules. There are multiple reasons to believe that such as transferable neural wavefunction may be feasible. Firstly, many molecules consist of recurring motives (known in chemistry as functional groups). It is therefore plausible that a wavefunction which has learned to represent the electronic structure of these groups, could also approximately represent the full wavefunction of a molecule comprised of these groups. Secondly, the intuition behind the embeddings $\boldsymbol{h}_i$ is to give a description of the electron i and its interaction with the surrounding environment (of nuclei and other electrons). Since the interaction terms of the Schrödinger Equation are identical across molecules, it is plausible to hope that these embeddings can generalize across different molecules.

This section outlines some of the challenges associated with designing such transferrable wavefunctions, discusses potential approaches, and finally introduces the *Transferable Atomic Orbitals* [20] as one proposed solution.

1.6.2 Design goals

A transferable neural wavefunction should be both highly expressive, i.e. able to accurately represent any molecular wavefunction, and generalize well across molecules, i.e. yield a good representation of the groundstate wavefunction for a previously unseen molecules. Several of the architectures discussed in section 1.5 are highly expressive, i.e. yield low energies when optimized for a single molecule, but they typically do not generalize across molecules. The following properties are desirable to allow efficient generalization:

- **Constant parameter count**: For some architectures the number of trainable parameters explicitly depends on the considered molecule. For example, FermiNet [10] uses a concatenation of all electron-nuclei pairs as input feature. The dimensionality of this feature scales with the number of nuclei and so also the dimensionality of weight matrices of MLPs acting on this feature depend on the number of nuclei. Such an ansatz can fundamentally not transfer across molecules of different size.
- **Permutation symmetry**: All ansatze enforce antisymmetry of the wavefunction with respect to permutation of electrons. Many [10, 15] do however ignore the invariance of the wavefunction with respect to permutation of the nuclei with the same nuclear charge Z. Enforcing this symmetry ensures that identical molecules, where indistinguishable atoms have been permuted, yields the same energy.
- **Translation and rotation symmetry**: The energy of a molecule is invariant under translation or rotation of the molecule. Most ansatze enforce translation invariance (by encoding all inputs as translation invariant differences between two particles), but none enforce rotational invariance.
- **Locality and size extensivity**: When a molecules consists of two subsystems, which are so far separated that the interactions between the subsystems can be neglected, the wavefunction should factorize into the product of the two subsystem-wavefunctions. By enforcing this property, the wavefunction for a large system can be composed from representations of local subsystem and thus efficiently transfer to larger molecules.

1.6.3 Challenges to transferability and potential approaches

When considering purely the aspect of parameter shapes, many components of neural network wavefunction ansatze are in principle transferable, i.e. the number of parameters does not explicitly depend on the system size. When breaking down a typical ansatz into the four steps of input features, embeddings, orbitals, and Slater determinants (section 1.5.1), the embedding and Slater determinants typically pose no issues:

When evaluating the embedding, the pairwise functions $f^{\uparrow}$, $f^{\downarrow}$ eq. (1.65) and the single-particle function g eq. (1.66) can have the same structure independent of system size. When evaluating the embedding for a larger molecule, the functions will be evaluated more often (for each particle or each pair of particles), but the dimensions of the trainable parameters can be identical. The same holds true for computing the final wavefunction ψ via the Slater determinant and Jastrow factor. The determinant has no trainable parameters and the Jastrow factor is typically just a sum over functions of a single electron eq. (1.80) or pairs of electrons eq. (1.81).

The input features and orbitals however do typically have parameter dimensions which do explicitly depend on the system size. For example, when concatenating electron-nucleus pairs to obtain the initial input features for the electrons (eq. (1.62)), the input feature vector for each electron has dimension $3N_{\text{nuc}}$ and thus explicitly depends on system size. This issue can easily be circumvented by using pooled input features eq. (1.63), simultaneously restoring invariance of the ansatz with respect to permutation of nuclei.

While the issue of input features is easily resolved, the most challenging aspect of a transferable ansatz is the generation of the Slater matrix. Recall that the elements of the Slater matrix $\boldsymbol{\Phi}$ are typically constructed from linear projection of the embeddings $\boldsymbol{h}_i$ and an envelope function given

as a sum over nuclei

$$\Phi_{ik} = \langle \boldsymbol{W}_k, \boldsymbol{h}_i \rangle \sum_{J=1}^{N_{\text{nuc}}} \pi_{kJ} e^{-\alpha_{kJ}|\boldsymbol{r}_i - \boldsymbol{R_J}|}. \tag{1.82}$$

Here Φ_{ik} denotes the element of the Slater matrix corresponding to electron i and orbital k, and $\boldsymbol{h}_i \in \mathbb{R}^{D_{\text{emb}}}$ denotes the high-dimensional embedding of electron i. The tensors $\boldsymbol{W} \in \mathbb{R}^{N_{\text{orb}} \times D_{\text{emb}}}$ and $\boldsymbol{\pi}, \boldsymbol{\alpha} \in \mathbb{R}^{N_{\text{orb}} \times N_{\text{nuc}}}$ are trainable parameters. Since the number of orbitals N_{orb} must be equal to n_{el} to obtain a square Slater matrix, all three parameters explicitly depend on the number of electrons and/or the number of nuclei. The following briefly discusses potential options to address this challenge.

Separate orbitals for each system The most naive approach is to simply not tackle this challenge and accept that the model cannot generalize across systems of different sizes. This was first proposed by [13] (section 2.2), which uses a separate set of orbital parameters for each new system. Since the neural network embedding contains most parameters, this strategy still allows to transfer $\approx 95\%$ of parameters across systems, yielding substantial speedups when computing energies for many geometries of a molecule. The approach is somewhat generalized by Gao et al. [11, 16], which is still limited to a single molecule, but can represent arbitrary geometries with a single set of parameters.

Auto-regressive generation of orbitals Instead of generating all orbitals at once (potentially requiring parameters scaling with the number of orbitals), one could also generate them autoregressively. In this approach outputs (in this case orbitals or orbital parameters) are predicted sequentially, with each output being conditioned on all previous outputs. This paradigm has been highly successful in language modelling [21, 22] and has also been applied to neural network wavefunctions in second quantization [23]. A key drawback of this approach is the potentially slow evaluation of the approach due to the inherently sequential nature of output generation, which would require N_{orb} sequential passes to generate N_{orb} orbitals. To the author's knowledge this approach has not yet been applied to neural wavefunctions in first quantization.

Orbital features as additional inputs An alternative approach is to not parameterize orbital dependant parameters like $\boldsymbol{W}_k$, $\boldsymbol{\pi}_k, \boldsymbol{\alpha}_k$ directly, but to instead define them as trainable functions of some additional orbital descriptor $\boldsymbol{c}_k$.

$$\boldsymbol{W}_k := f(\boldsymbol{c}_k) \quad \boldsymbol{\pi}_k := g(\boldsymbol{c}_k) \quad \boldsymbol{\alpha}_k := h(\boldsymbol{c}_k) \tag{1.83}$$

There are several options to generate orbital descriptors $\boldsymbol{c}_k$. Gao et al. [14] proposed an ad-hoc construction based on chemical bonding rules to specify bonds and corresponding orbitals. This yields good results for some molecules but appears to generalize poorly to new systems.

A key contribution of this thesis is to instead use orbitals from a computationally cheap mean-field calculations (e.g. Hartree-Fock in a small basis set) to generate orbital features $\boldsymbol{c}_k$. By using the basis expansion coefficients of the orbitals as descriptors $\boldsymbol{c}_k$, one can compute the orbital-dependant parameters via eq. (1.83). This approach is discussed in more detail in section 1.6.4 and forms the basis of the papers summarized in sections 2.3 to 2.5.

Determinant-free antisymmetrization Finally one could in principle also avoid Slater determinants entirely (which require square matrices) and instead use a different antisymmetrization procedure. Several methods have been proposed which bring the additional benefit of offering

computationally better scaling than the $\mathcal{O}({n_\mathrm{el}}^3)$ of the determinant. Han et al. [24] essentially use a wavefunction of the form (spin omitted for the sake of simplicity)

$$\psi = \prod_{1 \leq i < j \leq n_\mathrm{el}} \left[f(\boldsymbol{r}_i, \boldsymbol{r}_j) - f(\boldsymbol{r}_j, \boldsymbol{r}_i) \right], \tag{1.84}$$

with a trainable pairwise function f. This construction is antisymmetric, can be easily extended to incorporate arbitrary correlation by using functions of the embeddings $\boldsymbol{h}_i$, rather than the coordinates $\boldsymbol{r}_i$ and has no parameters that depend explicitly on the system size. Unfortunately results by Han et al. achieve only very low accuracy, with errors exceeding hundreds of milli-Hartrees for single atoms like Boron, while determinant-based antisymmetrization schemes achieve sub-milli-Hartree accuracy for these systems [10]. Richter-Powell et al. [25] propose an approach based on sorting, yielding in principle a favorable scaling of $\mathcal{O}(n_\mathrm{el} \log n_\mathrm{el})$, but with similar errors in excess of hundreds of milli-Hartrees for single atoms. Given their low accuracy, none of these approaches have been applied to transferable wavefunctions.

A more promising approach is a pre-print by Gao et al. [26], which uses Pfaffians for antisymmetrization. While this approach introduces some additional complexity it avoids the need for generating exactly as many orbitals as electrons and appears to yield highly accurate energies.

1.6.4 Transferable atomic orbitals

To construct the transferable orbitals, we start with canonical mean-field orbitals $\phi_k(\boldsymbol{r})$, expanded in an atom centered basis set

$$\phi_k^\mathrm{HF}(\boldsymbol{r}) = \sum_{J=1}^{N_\mathrm{nuc}} \sum_{\mu=1}^{N_\mathrm{basis}} c_{kJ\mu}\, b_\mu(\boldsymbol{r} - \boldsymbol{R_J}), \tag{1.85}$$

with fixed 3-dimensional basis functions $b : \mathbb{R}^3 \to \mathbb{R}$ and the tensor of expansion coefficients $\boldsymbol{c} \in \mathbb{R}^{N_\mathrm{orb} \times N_\mathrm{nuc} \times N_\mathrm{basis}}$. Note that μ enumerates all unique basis functions (and the sum over J yields all basis functions throughout the molecule), whereas most quantum chemistry codes typically merge the indices J, μ into a single index enumerating all basis functions. The expansion coefficients $\boldsymbol{c}_{kJ}$ yield a natural descriptor of the mean field orbital k around the nucleus J. To obtain correlated transferable orbitals, we use these descriptors as inputs to predict the orbital-dependent parameters in the final layer of the neural network wavefunction. Starting from the FermiNet-style orbitals

$$\Phi_{ik} = \langle \boldsymbol{W}_k, \boldsymbol{h}_i \rangle \sum_{I=1}^{N_\mathrm{nuc}} \pi_{kJ} e^{-\alpha_{kJ} |\boldsymbol{r}_i - \boldsymbol{R_J}|}, \tag{1.86}$$

we first extend the matrix $\boldsymbol{W} \in \mathbb{R}^{N_\mathrm{orb} \times D_\mathrm{emb}}$ with an additional dimension corresponding to the nuclei. Introducing this extended matrix $\hat{\boldsymbol{W}} \in \mathbb{R}^{N_\mathrm{orb} \times N_\mathrm{nuc} \times D_\mathrm{emb}}$ and moving it into the sum yields

$$\Phi_{ik} = \sum_{J=1}^{N_\mathrm{nuc}} \left\langle \hat{\boldsymbol{W}}_{kJ}, \boldsymbol{h}_i \right\rangle \pi_{kJ} e^{-\alpha_{kJ} |\boldsymbol{r}_i - \boldsymbol{R_J}|}. \tag{1.87}$$

The factor π_{kJ} can be absorbed into the $\hat{\boldsymbol{W}}_{kJ}$, simplifying to

$$\Phi_{ik} = \sum_{J=1}^{N_\mathrm{nuc}} \left\langle \hat{\boldsymbol{W}}_{kJ}, \boldsymbol{h}_i \right\rangle e^{-\alpha_{kJ} |\boldsymbol{r}_i - \boldsymbol{R_J}|}. \tag{1.88}$$

To obtain transferable orbitals, all that needs to be done at this point is to exchange the trainable parameters $\hat{\boldsymbol{W}}_{kJ}, \boldsymbol{\alpha}$ with functions that predict these parameters from the orbital descriptors $\boldsymbol{c}_{kJ}$.

$$\Phi_{ik} = \sum_{J=1}^{N_{\text{nuc}}} \langle f(\boldsymbol{c}_{kJ}), \boldsymbol{h}_i \rangle \, e^{-g(\boldsymbol{c}_{kJ})|\boldsymbol{r}_i - \boldsymbol{R}_J|}, \tag{1.89}$$

with trainable functions $f : \mathbb{R}^{N_{\text{basis}}} \to \mathbb{R}^{D_{\text{emb}}}, g : \mathbb{R}^{N_{\text{basis}}} \to \mathbb{R}^{+}$, which in practice are implemented as MLPs. Because these transferable orbitals are based on atom-wise features, they have been dubbed *Transferable Atomic Orbitals (TAOs)*.

Orbital localization for improved generalization While using the expansion coefficients $\boldsymbol{c}_{kJ}$ of the canonical Hartree-Fock orbitals in eq. (1.89) works, generalization can be further improved by using localized orbitals. Any wavefunction given by a Slater determinant is invariant under multiplication of the Slater matrix with another matrix U that satisfies $\det[U] = 1$,

$$\psi = \det[\Phi] = \det[\Phi] \det[U] = \det[\Phi U]. \tag{1.90}$$

The Hartree-Fock orbitals (and correspondingly the expansion coefficients $\boldsymbol{c}$) can therefore be linearly combined to a new set of orbitals

$$\hat{\boldsymbol{c}}_k = \sum_{k'} \boldsymbol{c}_k U_{kk'}, \tag{1.91}$$

which represents exactly the same wavefunction. Because the only constraint is $\det[U] = 1$, U can be chosen to optimize an additional property of the orbitals. One particular appealing property is to choose orbitals φ_k^{loc}, which are *maximally localized*. Localized orbitals are useful for a transferable neural network wavefunction, because it will ensure that orbitals for larger systems are qualitatively similar to the orbitals found in smaller systems (see fig. 1.1).

Several metrics have been proposed to measure the delocalization of the orbitals, which can then be minimized by a suitable choice of U. A natural choice is the sum of spatial variance of each orbital φ_k^{loc}, as proposed by Foster et al. [27]:

$$\Omega = \sum_{k=1}^{N_{\text{orb}}} \int |\varphi_k^{\text{loc}}(\boldsymbol{r})|^2 \, |\boldsymbol{r} - \bar{\boldsymbol{r}}_k|^2 \, d\boldsymbol{r} \tag{1.92}$$

$$\bar{\boldsymbol{r}}_k = \int |\varphi_k^{\text{loc}}(\boldsymbol{r})|^2 \, \boldsymbol{r} \, d\boldsymbol{r} \tag{1.93}$$

Several other localization losses have been proposed, many of which are based on localizing the atomic charge attributable to each orbital. This has originally been proposed by Pipek et al. [28] (using potentially ill-defined Mulliken-charges) and has been extended to several other well-defined charge metrics [29]. Orbital localization is computationally cheap and readily implemented in quantum chemistry software packages such as PySCF [30].

The canonical Hartree-Fock orbitals are eigenfunctions of the Fock operator and typically delocalized, i.e. they are nonzero across the entire molecule. This is a consequence of the kinetic energy operator, penalizing large curvature required to build local orbitals. The localized orbitals on the other hand are typically centered on a single site and decay rapidly as a function of distance from this orbital center. It can in fact be shown that for insulators, there exist orbitals which decay exponentially as a function of the distance from the orbital center [31]. Figure 1.1a shows this difference on the example of Hydrogen chains. While the canonical orbitals are spread across

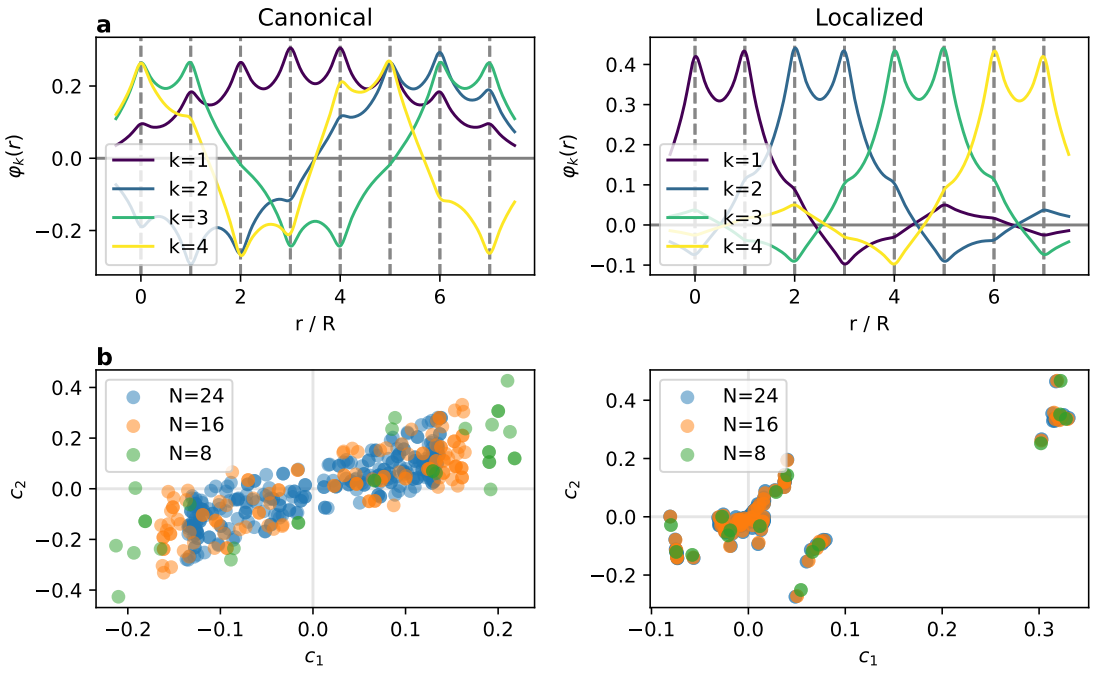

Figure 1.1: **Canonical vs. localized orbitals for the Hydrogen chain**: **a)** Value of the 4 occupied orbitals for a Hydrogen chain of length 8 along the chain. Dashed lines mark atom positions. While the canonical orbitals are non-zero along the entire chain, the localized orbitals decay rapidly from their centers, essentially corresponding to chemical bonds between 2 neighboring atoms. **b)** Scatter-plot of the expansion coefficients $\boldsymbol{c}_{kJ}$ for each orbital k and atom J for chains with varying number of atoms N. In the case of localized orbitals (right panel), the coefficients for large chains are mostly identical to the features found in smaller chain, enabling efficient generalization. For the canonical orbitals (left panel) coefficients present in short chains do not re-appear in longer chains, rendering generalization more difficult.

the entire chain, the localized orbitals have most of their contribution centered on one pair of Hydrogen atoms each, essentially corresponding to chemical bonds. Localized orbitals also lead to expansion coefficients which are consistent across atom positions and system sizes (see fig. 1.1b), aiding generalization across system sizes.

Non-uniqueness of orbital descriptor One issue of using the expansion coefficients $\boldsymbol{c}_k$ of mean-field orbitals as descriptors is that they are not unique. Methods such as Hartree-Fock iteratively solve a generalized eigenvalue problem and the coefficients $\boldsymbol{c}_k$ are the eigenvectors corresponding to the N_{orb} lowest eigenvalues ϵ_k. If eigenvalues are degenerate then any linear combination of the eigenvectors in the degenerate subspace is also an eigenvector and thus a potential solution obtained by the mean-field method. Even if the eigenvalues are not degenerate there is an ambiguity in the sign of the eigenvector: For every eigenvector $\boldsymbol{c}_k$, also $-\boldsymbol{c}_k$ is an eigenvector to the same eigenvalue and it is essentially random whether the mean-field method yields $\boldsymbol{c}_k$ or $-\boldsymbol{c}_k$. This means that for the same molecule given by $\boldsymbol{R}, \boldsymbol{Z}$ different orbital descriptors $\boldsymbol{c}$ may be obtained, yielding different wavefunctions and energies.

There are several potential strategies to overcome this issue: One could canonicalize the orbitals, e.g. by projecting $\boldsymbol{c}$ on a fixed vector $\boldsymbol{v}$ and choosing the sign of $\boldsymbol{c}$ such that $\langle \boldsymbol{c}, \boldsymbol{v} \rangle > 0$. For example by choosing $\boldsymbol{v}_{kI\nu} = 1$ we enforce that the mean of $\boldsymbol{c} > 0$. However this approach is ill-defined for $\boldsymbol{c} \perp \boldsymbol{v}$ and yields discontinuous features around $\boldsymbol{c} \perp \boldsymbol{v}$. In practice, for localized orbitals, canonicalization works well and is empirically stable.

Alternatively one could construct features from $\boldsymbol{c}_k$, which contain all information but are invariant w.r.t. the sign of $\boldsymbol{c}_k$. One example of such an invariant representation proposed by [32] is to use an outer product $\boldsymbol{c}_k \boldsymbol{c}_k^T$. While this works in principle it also substantially increases the dimension of the input feature, potentially increasing computational cost.

Lastly, one can choose the functions f, g in eq. (1.89) to be symmetric with respect to the sign of the input $\boldsymbol{c}_{kI}$. In particular by choosing g to be even, i.e. $g(-\boldsymbol{c}_{kI}) \equiv g(\boldsymbol{c}_{kI})$ and f to be odd, i.e. $f(-\boldsymbol{c}_{kI}) \equiv -f(\boldsymbol{c}_{kI})$, the resulting orbitals become equivariant to the sign of $\boldsymbol{c}_k$. This leads to energies that are invariant to the sign of $\boldsymbol{c}$ (thus rendering the non-unique sign of $\boldsymbol{c}$ irrelevant) and simplifies pretraining of the orbitals against a mean-field reference. To obtain an even or odd function from an arbitrary function f, one can simply evaluate it on the original input as well as the input with the opposite sign:

$$f^{\text{even}}(x) := f(x) + f(-x) \qquad f^{\text{odd}}(x) := f(x) - f(-x) \tag{1.94}$$

Properties Comparing the TAOS against the desired properties of a transferable ansatz listed in section 1.6.2, they meet many of them. The ansatz has a constant parameter count, independent of system size. The wavefunction is invariant under permutation of atoms as long as the electron-embeddings $\boldsymbol{h}_i$ are permutation invariant, which is easily achieved using message passing neural networks.

When using an odd function f to predict the backflow-weights the wavefunction is also local and size extensive. If a system is comprised of two independent subsystems A, B, which are so far separated that interactions between the two systems can be ignored, the total (mean-field) Hamiltonian H_{AB} corresponds to the sum of the Hamiltonians for each of the subsystems. In matrix-form this means

$$H^{AB} = \begin{pmatrix} H^A & 0 \\ 0 & H^B \end{pmatrix}, \tag{1.95}$$

and the eigenvectors are of the form

$$\boldsymbol{c}_k^{AB} = \begin{pmatrix} \boldsymbol{c}_{k_A} \\ 0 \end{pmatrix} \quad k = k_A = 1 \dots N_{\text{orb}}^A \tag{1.96}$$

$$\boldsymbol{c}_k^{AB} = \begin{pmatrix} 0 \\ \boldsymbol{c}_{k_B} \end{pmatrix} \quad k = N_{\text{orb}}^A + k_B, k_B = 1 \dots N_{\text{orb}}^B, \tag{1.97}$$

i.e. the eigenvectors are only non-zero on one of the two subsystems and correspond to the eigenvectors of the subsystem Hamiltonians. If the orbital features $\boldsymbol{c}_k$ are only non-zero on one of the subsystems, then also the odd function f is non-zero only on atoms belonging to this subsystem. The Slater Matrix Φ_{ik} is therefore block-diagonal

$$\Phi = \begin{pmatrix} \Phi^A & 0 \\ 0 & \Phi^B \end{pmatrix}, \tag{1.98}$$

and the wavefunction therefore factorizes into the product of the wavefunctions for the two subsystems.

The only desirable symmetry not captured by this ansatz is the rotational symmetry. While rotation of a molecule leads to a predictable transformation of the basis coefficients $\boldsymbol{c}$, the subsequent functions f, g currently do not exploit this symmetry.

Variations Throughout the publications that form the core of this thesis [20, 33, 34] several variations of the TAOs have been used, which are discussed in greater depth in the particular publications. Possible variations include:

- **Orbital GNN**: Instead of using the orbital features $\boldsymbol{c}_{kI}$ directly, it can be advantageous to allow exchange of information between different nuclei J for each orbital k. This can be achieved by a graph neural network (GNN) acting independently on each orbital k, by using layers of the form

$$\boldsymbol{c}_{kI}^{l+1} = \sum_J \mathrm{MLP}\left(\boldsymbol{R}_I - \boldsymbol{R}_J\right) \odot \mathrm{MLP}\left(\boldsymbol{c}_{kJ}^{l}\right). \tag{1.99}$$

- **Additional orbital features**: Beyond the orbital expansion coefficients $\boldsymbol{c}_k$, also other orbital specific features can be included in the feature vector. In particular orbital energy ϵ_k or the mean position $\boldsymbol{R}_k^{\mathrm{orb}}$ of the orbital can be concatenated to $\boldsymbol{c}_{kI}$ to augment the orbital descriptor.

- **Electron-nucleus embeddings**: Since almost all terms in eq. (1.89) are given as a sum over nuclei, it is somewhat natural to also extend the electron embeddings to not represent a single electron $\boldsymbol{h}_i$, but instead represent an interaction $\boldsymbol{h}_{iJ}$ between electron i and nucleus J. This generalization is particular natural for embeddings such as Moon [14] where the final electron embedding is given by a sum over such $\boldsymbol{h}_{iJ}$ embeddings. Simply omitting this sum and instead combining it with the sum of eq. (1.89) yields the electron-nucleus TAOs eq. (1.100)

$$\Phi_{ik} = \sum_{J=1}^{N_{\mathrm{nuc}}} \langle f(\boldsymbol{c}_{kJ}), \boldsymbol{h}_{iJ} \rangle \, e^{-g(\boldsymbol{c}_{kJ})|\boldsymbol{r}_i - \boldsymbol{R}_J|}. \tag{1.100}$$

  For periodic systems this formulation is more expressive than the TAOs of eq. (1.89) as shown in the appendix of [34].

1.7 Sampling

1.7.1 From integration to sampling

To compute the expectation value of some operator O for a given unnormalized wavefunction ψ, the following integral must be evaluated:

$$\langle O \rangle_\psi = \frac{1}{\int \psi^*(\boldsymbol{r})\psi(\boldsymbol{r}) d\boldsymbol{r}} \int \psi^*(\boldsymbol{r})(O\psi)(\boldsymbol{r}) d\boldsymbol{r} \tag{1.101}$$

Common examples for O are the Hamiltonian operator H – the expectation value of which corresponds to the energy – or the position operator $\hat{X} := \frac{1}{n_{\mathrm{el}}} \sum_i \boldsymbol{r}_i$, which yields the mean electron position. These integrals are in general hard to evaluate, because there is no analytic closed form and numerical integration is hard due to the high dimensionality of the integrand (being $3 \times n_{\mathrm{el}}$).

The key trick of Monte Carlo integration is to simply divide and multiply by ψ in the second integral, yielding the following expression (the dependence of ψ on $\boldsymbol{r}$ has been dropped for clarity):

$$\langle O \rangle_\psi = \frac{1}{\int \psi^* \psi d\boldsymbol{r}} \int \psi^* O \psi d\boldsymbol{r} \tag{1.102}$$

$$= \frac{1}{\int \psi^* \psi d\boldsymbol{r}} \int \psi^* \psi \frac{O\psi}{\psi} d\boldsymbol{r} \tag{1.103}$$

$$= \int p(\boldsymbol{r}) \frac{O\psi}{\psi} d\boldsymbol{r} \tag{1.104}$$

$$= \left\langle \frac{O\psi}{\psi} \right\rangle_{\boldsymbol{r} \sim p} \tag{1.105}$$

with

$$p(\boldsymbol{r}) := \frac{|\psi(\boldsymbol{r})|^2}{\int |\psi(\boldsymbol{r'})|^2 d\boldsymbol{r'}} \tag{1.106}$$

Since $p(\boldsymbol{r})$ is positive (because of $|\cdot|^2$) and its integral is 1 (because of the normalization in the denominator), it corresponds to a probability density. Equation (1.104) is the definition of the expectation value for $\frac{O\psi}{\psi}$, given that the samples $\boldsymbol{r}$ are distributed according to the probability density $p(\boldsymbol{r})$.

Therefore, estimating the original integral eq. (1.101) is equivalent to estimating the expectation value in eq. (1.105). To estimate this expectation value, one samples N_s iid electron configurations $\boldsymbol{r}$ distributed according to the probability distribution p (which in turn is proportional to $|\psi(\boldsymbol{r})|^2$) and computes the mean of $\frac{O\psi}{\psi}$:

$$\langle O \rangle_\psi \approx \frac{1}{N_\mathrm{s}} \sum_{n=1}^{N_\mathrm{s}} \frac{(O\psi)(\boldsymbol{r}_n)}{\psi(\boldsymbol{r}_n)} \tag{1.107}$$

Note that the index n here does not run over the number of electrons, but over the number of samples. Each $\boldsymbol{r}_n$ is a full set of electron positions in $\mathbb{R}^{n_\mathrm{el} \times 3}$.

According to the Central Limit Theorem the variance of eq. (1.107) approaches $\frac{1}{N_\mathrm{s}} \mathrm{var}(\frac{O\psi}{\psi})$ as $N_\mathrm{s} \to \infty$. The expected approximation error does therefore not depend on the dimensionality of the problem and can in principle be systematically improved by drawing more samples, but only converges at the relatively slow rate of $\frac{1}{\sqrt{N_\mathrm{s}}}$ that is typical for Monte Carlo Methods. Note however that if $\frac{O\psi}{\psi}$ has low variance, even a small number of samples yields an accurate estimate of $\langle O \rangle_\psi$. One particularly important case of this energy estimation: As ψ approaches an eigenstate (e.g. the ground-state ψ_0), $\frac{H\psi}{\psi}$ approaches the corresponding energy eigenvalue, which is a constant and thus has zero-variance. Therefore as optimization of a VMC wavefunction progresses, not only does the energy of the wavefunction ansatz decrease, also the uncertainty of this energy decreases as well.

1.7.2 Sampling using Markov Chain Monte Carlo

Equation (1.107) shifts the problem of integration to a problem of sampling: How can we draw samples $\boldsymbol{r}$ that are distributed according to some given high-dimensional probability distribution $p(\boldsymbol{r})$? While drawing samples is easy for some specific distributions (e.g. samples from a uniform distribution or a multivariate Gaussian distribution), sampling from an arbitrary distribution is non-trivial.

The Metropolis-Hastings algorithm A simple, but effective algorithm to draw a sample from an arbitrary probability distribution (algorithm 1) was proposed by Metropolis et al.[1] and generalized by Hastings [36]. This algorithm produces a Markov chain of samples $\boldsymbol{r}_n$ with

Algorithm 1 Metropolis-Hastings sampling

Require: Probability density $p(\boldsymbol{r})$, proposal distribution $q(\boldsymbol{r}_p|\boldsymbol{r}_n)$, initial configurations $\boldsymbol{r}_0$, number of steps N

```
for n = 0 to N − 1 do
    r_p ∼ q(r_p|r_n)                                  ▷ Propose new configuration r_p
    a = min(1, p(r_p)q(r_n|r_p) / p(r_n)q(r_p|r_n))   ▷ Compute acceptance probability a
    if a ≥ RandomUniform(0, 1) then
        r_{n+1} ← r_p                                 ▷ Accept the proposal with probabilty a
    else
        r_{n+1} ← r_n                                 ▷ Reject the proposal with probabilty 1 − a
    end if
end for
return r_N        ▷ In the limit of N → ∞, r_N is distributed according to p(r)
```

stationary distribution p. Thus, given a suitable proposal function q, the algorithm produces a sample $\boldsymbol{r}_N$ that is distributed according to p in the limit of $N \to \infty$. Intuitively, over multiple steps n the sample tends to move towards high-probability regions because proposals towards higher probability are always accepted, while proposals towards lower-probability regions are often rejected. Because there is some chance to accept proposals towards low-probability regions, the algorithm does not only return the value with highest probability, but a distribution of samples. Figure 1.2a depicts the convergence of the distribution of samples towards the target distribution p for a simple 1D example. The approach of sampling using a Markov Chain is generally known as Markov Chain Monte Carlo (MCMC). The Metropolis-Hastings (MH) algorithm is one particular algorithm to generate stochastic transitions between states,which ensure that the Markov Chain converges to the targeted stationary distribution.

Sketch of proof of correctness A Markov Chain converges to its stationary distribution $\pi(\boldsymbol{r})$ if the chain is ergodic (meaning that any state can transition to any other state in a finite number of steps) and a stationary distribution exists. A sufficient condition[36] for the existence of a stationary distribution $\pi(\boldsymbol{r})$ is known as *detailed balance*

$$\pi(\boldsymbol{r}_n)k(\boldsymbol{r}_m|\boldsymbol{r}_n) = \pi(\boldsymbol{r}_m)k(\boldsymbol{r}_n|\boldsymbol{r}_m), \tag{1.108}$$

where $k(\boldsymbol{r}_m|\boldsymbol{r}_n)$ denotes the probability of transitioning from configuration $\boldsymbol{r}_n$ to configuration $\boldsymbol{r}_m$. The Metropolis-Hastings algorithm is one way to construct a transition probability kernel k that satisfies detailed balance and has the target distribution p as stationary distribution π.

Detailed balance (DB) is sufficient for the existence of a stationary distribution, because the

[1]This paper from 1953 has a very illustrious author list: Co-authors of Nicholas Metropolis were Augusta Teller (who developed the initial implementation), Arianna Rosenbluth (who developed the final implementation), and their husbands Marshall Rosenbluth (who was later known as the "pope of plasma physics" and chief scientist at ITER) and Edward Teller (the "father of the hydrogen bomb"). The paper also bears the great title "Equation of State Calculations by Fast Computing Machines", tempting the author of this thesis to name it "Neural Wavefunctions using Very Fast Computing Machines".

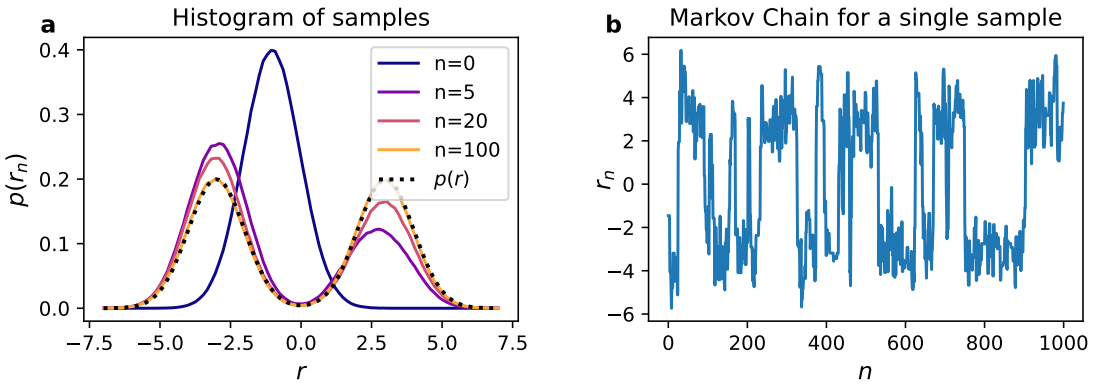

Figure 1.2: 1D example of MCMC on a 1D density $p(\boldsymbol{r})$ consisting of two Gaussians. The initial configurations $\boldsymbol{r}_0$ are drawn from a single Gaussian distribution. **a**: Histogram of samples after different number of MCMC steps n. After $n \approx 100$ steps the distribution of $\boldsymbol{r}_n$ aligns with the target distribution p. **b**: Path of a single sample. Subsequent samples are strongly correlated, depicting two distinct time-scales: A short time-scale corresponding to moves within a density peak and a long time scale corresponding to moves between the two peaks.

distribution π_{n+1} after a step is identical to the distribution π_n before the step:

$$\pi_{n+1}(\boldsymbol{r}) = \int \pi_n(\boldsymbol{r}')k(\boldsymbol{r}|\boldsymbol{r}')d\boldsymbol{r}' \tag{1.109}$$

$$\overset{\text{DB}}{=} \int \pi_n(\boldsymbol{r})k(\boldsymbol{r}'|\boldsymbol{r})d\boldsymbol{r}' \tag{1.110}$$

$$= \pi_n(\boldsymbol{r}) \int k(\boldsymbol{r}'|\boldsymbol{r})d\boldsymbol{r}' \tag{1.111}$$

$$= \pi_n(\boldsymbol{r}). \tag{1.112}$$

The Metropolis Hastings algorithm satisfies detailed balance, with stationary distribution p. For $\boldsymbol{r}_n = \boldsymbol{r}_m$ it is trivially satisfied. For $\boldsymbol{r}_n \neq \boldsymbol{r}_m$, the transition probability $k(\boldsymbol{r}_m|\boldsymbol{r}_n)$ is given by the product of the corresponding proposal times the probability of it being accepted (since $\boldsymbol{r}_n \neq \boldsymbol{r}_m$ it cannot have been rejected)

$$k(\boldsymbol{r}_m|\boldsymbol{r}_n) = q(\boldsymbol{r}_m|\boldsymbol{r}_n)p_{\text{accept}}(\boldsymbol{r}_n \to \boldsymbol{r}_m) \tag{1.113}$$

$$= q(\boldsymbol{r}_m|\boldsymbol{r}_n) \min\left(1, \frac{p(\boldsymbol{r}_m)q(\boldsymbol{r}_n|\boldsymbol{r}_m)}{p(\boldsymbol{r}_n)q(\boldsymbol{r}_m|\boldsymbol{r}_n)}\right). \tag{1.114}$$

Thus detailed balance for the MH-update is satisfied, since

$$p(\boldsymbol{r}_n)k(\boldsymbol{r}_m|\boldsymbol{r}_n) = p(\boldsymbol{r}_n)q(\boldsymbol{r}_m|\boldsymbol{r}_n) \min\left(1, \frac{p(\boldsymbol{r}_m)q(\boldsymbol{r}_n|\boldsymbol{r}_m)}{p(\boldsymbol{r}_n)q(\boldsymbol{r}_m|\boldsymbol{r}_n)}\right) \tag{1.115}$$

$$= \min\Big(p(\boldsymbol{r}_n)q(\boldsymbol{r}_m|\boldsymbol{r}_n),\ p(\boldsymbol{r}_m)q(\boldsymbol{r}_n|\boldsymbol{r}_m)\Big), \tag{1.116}$$

and the same result can be obtained when flipping the initial indices n, m. Therefore

$$p(\boldsymbol{r}_n)k(\boldsymbol{r}_m|\boldsymbol{r}_n) = p(\boldsymbol{r}_m)k(\boldsymbol{r}_n|\boldsymbol{r}_m), \tag{1.117}$$

the MH-algorithm satisfies detailed balance and its stationary distribution is given by p.

Practical considerations A common choice for the proposal function $q(\boldsymbol{r}_p|\boldsymbol{r}_n)$ is a multivariate Gaussian distribution centered around $\boldsymbol{r}_n$ and variance s^2

$$q(\boldsymbol{r}_p|\boldsymbol{r}_n) \propto \exp\left(-\frac{1}{2s^2}||\boldsymbol{r}_p - \boldsymbol{r}_n||^2\right) \tag{1.118}$$

where s is a tuneable parameter known as the step size. It is a valid choice, because in principle any configuration can be reached from any other configuration in a single step (since the Gaussian distribution has support on the whole domain) and thus the proposal satisfies ergodicity. Furthermore the fact that the Gaussian is symmetric in $\boldsymbol{r}_p$ and $\boldsymbol{r}_n$ allows to omit the q-ratio on the calculation of the acceptance rate since it is always 1. An alternative to a Gaussian Proposal distribution is to bias proposals towards increasing probability density to increase the probability of acceptance. This is known as Metropolis Adjusted Langevin Algorithm (MALA) and increases sampling efficiency[18] at the expense of higher computational cost to evaluate ∇p.

$$\boldsymbol{r}_p = \boldsymbol{r}_n + \tau\nabla \log p(\boldsymbol{r}_n) + s\,\mathcal{N}(\mathbf{0}, 1) \tag{1.119}$$

The choice of the step size s is important to achieve fast convergence and mixing of the Markov Chain: Choosing a very small step size only allows very small changes in $\boldsymbol{r}$, leading to slow convergence. Choosing a very large step size leads to proposed configurations $\boldsymbol{r}_p$ that are far from the original configuration $\boldsymbol{r}_n$ and are very often rejected, thus not moving at all. To address this issue, one can set a target acceptance rate of $\approx 50\%$ and automatically adjust s to approximately reach this acceptance rate.

One important aspect of the Metropolis-Hastings algorithm is that its acceptance criterion only depends on the ratio of probability densities, but not p itself. Therefore p does not have to be normalized (since any normalization would cancel out), allowing to sample directly from any unnormalized density $|\psi|^2$.

In principle any initial distribution can be used for the samples $\boldsymbol{r}_0$, but choosing an initial distribution that resembles the target distribution is obviously advantageous. Therefore the typical approach is to first take a large number of steps $N_{\text{burn-in}}$ for the samples to converge to the target distribution p. Then, to obtain more samples one does not start again from the initial distribution, but uses the latest sample as starting point for the next $N_{\text{intermed.}}$ steps to obtain a new sample. In practice $N_{\text{burn-in}} \gg N_{\text{intermed.}}$, for sampling from a wavefunction of a small molecule $N_{\text{burn-in}} \approx \mathcal{O}(10^3)$, while $N_{\text{intermed.}} \approx \mathcal{O}(10^1)$. A disadvantage of this approach is that subsequent samples are not fully independent of each other, but can still be correlated if $N_{\text{intermed.}}$ is too small. Figure 1.2b shows the trajectory of a single sample as a function of MH-steps, clearly showing correlations between subsequent samples. This issue is particularly pronounced when p has multiple maxima that are separated by regions of low probability, because it takes many steps to transition between these maxima.

1.7.3 Sampling directly from a generative model

Besides using an arbitrary model for ψ (and thus implicitly p) and using Metroplis-Hastings to sample, one can in principle also design models which allow direct sampling from the probability distribution.

One option are Normalizing Flows, a type of model that maps an easy to sample probability distribution (e.g. a Gaussian) to the target probability distribution p. It has been applied to the Schrödinger Equation, but only in the substantially simplified 1D case [37].

Another option are autoregressive models, which generate a full configuration of electrons one electron at a time, by conditioning the probability distribution on all previously added electrons:

$$p(\boldsymbol{r}_1, \dots \boldsymbol{r}_{n_\mathrm{el}}) = p(\boldsymbol{r}_1)\, p(\boldsymbol{r}_2|\boldsymbol{r}_1)\, p(\boldsymbol{r}_3|\boldsymbol{r}_1, \boldsymbol{r}_2) \cdots p(\boldsymbol{r}_{n_\mathrm{el}}|\boldsymbol{r}_1, \dots \boldsymbol{r}_{n_\mathrm{el}-1}) \tag{1.120}$$

Instead of sampling from the $3 \times n_\mathrm{el}$-dimensional probability distribution (the LHS of eq. (1.120)) all at once, instead one samples n_el times from a 3-dimensional probability distribution (each term on the RHS of eq. (1.120)). This structure is the currently dominant paradigm in large language models, which autoregressively sample one token / word at a time, from a probability distribution which is conditioned on the previously generated tokens [38]. This approach has also been applied to wavefunctions, but so far only for model Hamiltonians [39] and molecules in second quantization [23]. In both cases the state space is discretized, simplifying the sampling from the low-dimensional conditional probability distributions.

1.8 Optimization

1.8.1 Gradient of the energy

The expectation value of the energy $\langle E \rangle$ for an unnormalized wavefunction ψ is given by

$$\langle E \rangle_\psi = \frac{\int \psi^*(\boldsymbol{r}) H \psi(\boldsymbol{r}) d\boldsymbol{r}}{\int \psi^*(\boldsymbol{r}) \psi(\boldsymbol{r}) d\boldsymbol{r}} = \Big\langle E(\boldsymbol{r}) \Big\rangle_{\boldsymbol{r} \sim |\psi(\boldsymbol{r})|^2}, \tag{1.121}$$

where $E(\boldsymbol{r}) := \frac{H\psi(\boldsymbol{r})}{\psi(\boldsymbol{r})}$ is the local energy (see eq. (1.105)) and $\langle \cdot \rangle$ denotes the expectation value. For brevity, in the following the notation will be shortened by omitting the subscript of the expectation value (which will always be taken using the probability distribution $|\psi^2|$) and omitting the argument $\boldsymbol{r}$ in the integrals (which will always be taken over $d\boldsymbol{r}$):

$$\langle E \rangle = \frac{\int \psi^* H \psi}{\int \psi^* \psi} = \frac{1}{\Omega} \int \psi^* H \psi \tag{1.122}$$

$$\Omega := \int \psi^* \psi. \tag{1.123}$$

For a real-valued wavefunction with real-valued parameters, the gradient is given by

$$\nabla \langle E \rangle = \Big\langle \left(\nabla \log |\psi|^2\right) \Delta E \Big\rangle \tag{1.124}$$

$$\Delta E(\boldsymbol{r}) := E(\boldsymbol{r}) - \langle E \rangle . \tag{1.125}$$

For the derivation and the more general case of a complex-valued wavefunction, see below. Throughout this section the subscript θ will be dropped for the gradient ($\nabla := \nabla_\theta$).

Properties of the gradient One interesting thing to note is that the energy gradient is given as the correlation between the gradient of the wavefunction and the local energies. In particular it does *not* depend on derivatives of the local energy w.r.t. θ, but only on derivatives of ψ w.r.t. θ. This is of importance, because computing the local energy E already involves second derivatives of ψ w.r.t. the electron coordinates $\boldsymbol{r}$ (required to compute the kinetic energy). Computing the energy gradient does however not require third derivatives, but only first derivatives of ψ w.r.t. θ and second derivatives w.r.t. $\boldsymbol{r}$. Another consequence of this dependence on ΔE is that the gradients vanish as ψ approaches an eigenstate (since $E(\boldsymbol{r}) \equiv \langle E \rangle$ for an eigenfunction). This is expected for the targeted minimum of the ground-state, but also holds true for any other higher-lying excited state.

Energy clipping In practice the expectation value in eq. (1.124) is approximated using a finite number of samples, which have been drawn from the distribution $|\psi|^2$ using Monte Carlo sampling (see section 1.7). One common issue is that large outliers of the sampled local energies $E(\boldsymbol{r})$ can cause large gradients, which potentially render the optimization unstable. To remedy this issue it has been proposed to clip the local energies for gradient computation [10], following the following scheme:

$$E_{\text{center}} = \text{AVERAGE}(E) \tag{1.126}$$

$$\Delta E = \text{CLIP}(E - E_{\text{center}}) \tag{1.127}$$

For the function AVERAGE either the mean, the median, or the mean of the clipped samples from the previous step have been proposed. For the function CLIP either thresholding[10] or soft-thresholding[7] have been proposed:

$$\text{CLIP}_{\text{hard}}(x) = \max(-\delta, \min(\delta, x)) \tag{1.128}$$

$$\text{CLIP}_{\text{soft}}(x) = \delta \tanh\left(\frac{x}{\delta}\right) \tag{1.129}$$

The clipping width δ can be chosen as some measure of spread of the local energies, e.g. the standard deviation or the mean absolute error

$$\delta_{\text{std}} = \sqrt{\frac{1}{N_{\text{s}}} \sum_{n=1}^{N_{\text{s}}} (\Delta E_n)^2} \tag{1.130}$$

$$\delta_{\text{MAE}} = \frac{1}{N_{\text{s}}} \sum_{n=1}^{N_{\text{s}}} |\Delta E_n|. \tag{1.131}$$

Proof of energy gradient The energy in eq. (1.121) can be written as

$$\langle E \rangle = \frac{\int \psi^* H \psi}{\int \psi^* \psi} = \frac{1}{\Omega} \int \psi^* H \psi \tag{1.132}$$

$$\Omega := \int \psi^* \psi, \tag{1.133}$$

where Ω denotes the normalization. The gradient of eq. (1.122) is given by

$$\nabla \langle E \rangle = \frac{1}{\Omega} \int (\nabla \psi^*) H \psi - \frac{1}{\Omega^2} (\nabla \Omega) \int \psi^* H \psi + \frac{1}{\Omega} \int \psi^* H (\nabla \psi). \tag{1.134}$$

The first two of these three terms can be simplified as follows:

$$\frac{1}{\Omega}\int (\nabla\psi^*)\, H\psi = \int \frac{\psi^*\psi}{\Omega}\frac{\nabla\psi^*}{\psi^*}\frac{H\psi}{\psi} \tag{1.135}$$

$$= \int \frac{|\psi|^2}{\Omega}\,(\nabla \log\psi^*)\, E \tag{1.136}$$

$$= \langle (\nabla \log\psi^*)\, E\rangle \tag{1.137}$$

$$\frac{1}{\Omega^2}\,(\nabla\Omega)\int \psi^* H\psi = \langle E\rangle \frac{1}{\Omega}\int \nabla(\psi^*\psi) \tag{1.138}$$

$$= \langle E\rangle \frac{1}{\Omega}\int \psi^*(\nabla\psi) + \psi(\nabla\psi^*) \tag{1.139}$$

$$= \langle E\rangle \int \frac{|\psi|^2}{\Omega}\left(\frac{\nabla\psi}{\psi} + \frac{\nabla\psi^*}{\psi^*}\right) \tag{1.140}$$

$$= 2\,\langle E\rangle\,\langle \mathrm{Re}[\nabla \log\psi]\rangle \tag{1.141}$$

To simplify the third term, we use the fact that H is a hermitian operator to apply H to ψ^* instead of $\nabla\psi$:

$$\frac{1}{\Omega}\int \psi^* H\,(\nabla\psi) = \frac{1}{\Omega}\int (\nabla\psi)\,(H\psi^*) \tag{1.142}$$

$$= \int \frac{|\psi|^2}{\Omega}\frac{\nabla\psi}{\psi}\frac{H\psi^*}{\psi^*} \tag{1.143}$$

$$= \langle (\nabla \log\psi)\, E^*\rangle \tag{1.144}$$

Putting everything together, we get

$$\nabla\,\langle E\rangle = \langle (\nabla \log\psi)\, E^*\rangle + \langle (\nabla \log\psi)\, E^*\rangle^* - 2\,\langle E\rangle\,\langle \mathrm{Re}[\nabla \log\psi]\rangle \tag{1.145}$$

$$= 2\,\mathrm{Re}\left[\langle (\nabla \log\psi)\, E^*\rangle\right] - 2\,\langle E\rangle\,\langle \mathrm{Re}[\nabla \log\psi]\rangle \tag{1.146}$$

If the parameters are constrained to be real-valued, only the real part of the gradient is relevant for the parameter update. It is given by

$$\mathrm{Re}\left[\nabla\,\langle E\rangle\right] = 2\,\mathrm{Re}\left[\langle (\nabla \log\psi)\, E\rangle\right] - 2\,\langle \mathrm{Re}\left[\nabla \log\psi\right]\rangle\,\langle \mathrm{Re}\left[E\right]\rangle \tag{1.147}$$

$$= 2\,\langle \mathrm{Re}\left[\nabla \log\psi\right]\mathrm{Re}\left[E\right]\rangle + 2\,\langle \mathrm{Im}\left[\nabla \log\psi\right]\mathrm{Im}\left[E\right]\rangle \tag{1.148}$$

$$- 2\,\langle \mathrm{Re}\left[\nabla \log\psi\right]\rangle\,\langle \mathrm{Re}\left[E\right]\rangle \tag{1.149}$$

$$= 2\,\langle \mathrm{Re}\left[\nabla \log\psi\right]\mathrm{Re}\left[E - \langle E\rangle\right]\rangle + 2\,\langle \mathrm{Im}\left[\nabla \log\psi\right]\mathrm{Im}\left[E\right]\rangle\,. \tag{1.150}$$

The wavefunction is often expressed in log-space, by parameterizing ψ as

$$\psi = e^{\frac{1}{2}\rho + i\phi}, \tag{1.151}$$

with real-valued functions $\rho(\boldsymbol{r})$ and $\phi(\boldsymbol{r})$, yielding

$$\rho = \log|\psi|^2 \qquad\qquad \phi = \arg\psi \tag{1.152}$$

$$\mathrm{Re}\left[\log\psi\right] = \frac{1}{2}\rho \qquad\qquad \mathrm{Im}\left[\log\psi\right] = \phi \tag{1.153}$$

Then the real part of the gradient corresponds to

$$\mathrm{Re}\left[\nabla\,\langle E\rangle\right] = \Big\langle \nabla\rho\ \mathrm{Re}\left[E - \langle E\rangle\right]\Big\rangle + 2\Big\langle \nabla\phi\ \mathrm{Im}\left[E\right]\Big\rangle \tag{1.154}$$

Because H is hermitian, the expectation value of the energy is real valued. We can therefore subtract a term proportional to $\operatorname{Im}\langle E\rangle$ without changing the expectation value, leading to the final expression for the energy gradient of a complex wavefunction:

$$\Delta E := E - \langle E\rangle \tag{1.155}$$

$$\operatorname{Re}\left[\nabla\langle E\rangle\right] = \Big\langle \nabla\rho \ \operatorname{Re}\left[\Delta E\right]\Big\rangle + 2\Big\langle \nabla\phi \ \operatorname{Im}\left[\Delta E\right]\Big\rangle \tag{1.156}$$

1.8.2 Stochastic reconfiguration as a preconditioner

The energy can be minimized using (stochastic) gradient descent (SGD) using the gradient from eq. (1.124) (for a real-valued wavefunction) or eq. (1.156) (for a complex-valued wavefunction). The update rule for the parameters can be as simple as

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \lambda\left(\nabla E\right), \tag{1.157}$$

with a given learning rate λ or involve some form of momentum, e.g. using the Adam optimizer [40]. However, convergence of optimization can be substantially accelerated by not using the energy gradient directly (as in eq. (1.157)), but rather preconditioning it with the following matrix $\boldsymbol{S} \in \mathbb{R}^{N_{\text{param}}\times N_{\text{param}}}$

$$S_{\mu\nu} := \Big\langle \frac{\partial \log\psi}{\partial\theta_\mu}\frac{\partial\log\psi}{\partial\theta_\nu}\Big\rangle - \Big\langle\frac{\partial\log\psi}{\partial\theta_\mu}\Big\rangle\Big\langle\frac{\partial\log\psi}{\partial\theta_\nu}\Big\rangle, \tag{1.158}$$

and using this preconditioned gradient for stochastic gradient descent

$$\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t - \lambda\boldsymbol{S}^{-1}\left(\nabla E\right). \tag{1.159}$$

The update rule of eq. (1.159) is known as *Stochastic Reconfiguration* in the physics community [41] (where $\boldsymbol{S}$ is then referred to as the *Quantum Geometric Tensor*) and is very closely related to *Natural Gradient Descent* in the machine learning community [43] (where an object closely related to $\boldsymbol{S}$ is referred to as the Fisher information matrix). The following derivations, which have been adapted from [41], should give some perspective on why using $\boldsymbol{S}$ as a preconditioner is a sensible choice.

1.8.3 Stochastic reconfiguration as a local metric

When performing (stochastic) gradient descent, a crucial choice is the step size λ. One way of formulating this is to consider as loss $\mathcal{L}$ the energy plus an additional regularization term, which penalizes large changes $\boldsymbol{\delta}$ in parameter space.

$$\mathcal{L} = \langle E\rangle + \frac{\lambda}{2}\boldsymbol{\delta}^T\boldsymbol{\delta} \tag{1.160}$$

$$\boldsymbol{\delta} := \boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}_t \tag{1.161}$$

When minimizing eq. (1.160) with respect to all parameters θ_μ the classical SGD update rule is recovered:

$$\frac{\partial\mathcal{L}}{\partial\theta_\mu} = \frac{\partial E}{\partial\theta_\mu} + \lambda\delta_\mu \overset{!}{=} 0 \tag{1.162}$$

$$\boldsymbol{\delta} = -\lambda\nabla E. \tag{1.163}$$

SGD with a given learning-rate λ therefore minimizes the energy, while at the same time minimizing the Euclidean norm of the parameter update. While this is not an unreasonable choice per se, it

would be better to minimize the energy, while making *minimal changes to the wavefunction*. After all, the wavefunction might be very sensitive to some parameters and insensitive to others. We would therefore like to make small steps for sensitive parameters and larger steps for insensitive parameters.

The following metric can be used to assess the distance between two unnormalized wavefunctions ψ and ϕ (for this section the wavefunctions are assumed to be real-valued):

$$s(\phi, \psi)^2 = 1 - \frac{\langle \psi, \phi \rangle^2}{\langle \psi, \psi \rangle \langle \phi, \phi \rangle} \tag{1.164}$$

Equation (1.164) corresponds to 1 minus the squared overlap of the normalized wavefunctions and is thus 0 for $\phi \equiv \psi$ and 1 for $\phi \perp \psi$. Using eq. (1.164) as a metric to regularize the loss yields

$$\mathcal{L} = \langle E \rangle + \lambda \left(1 - \frac{\langle \psi_\theta, \psi_{\theta+\delta} \rangle^2}{\langle \psi_\theta, \psi_\theta \rangle \langle \psi_{\theta+\delta}, \psi_{\theta+\delta} \rangle} \right). \tag{1.165}$$

The updated wavefunction $\psi_{\theta+\delta}$ can be expressed as Taylor expansion up to first order,

$$\psi_{\theta+\delta} \approx \psi_\theta + \boldsymbol{\delta}^T \nabla \psi_\theta, \tag{1.166}$$

yielding (subscripts θ omitted for clarity):

$$\mathcal{L} = \langle E \rangle + \lambda \left(1 - \frac{\left\langle \psi, \psi + \boldsymbol{\delta}^T \nabla \psi \right\rangle^2}{\langle \psi, \psi \rangle \left\langle \psi + \boldsymbol{\delta}^T \nabla \psi, \psi + \boldsymbol{\delta}^T \nabla \psi \right\rangle} \right). \tag{1.167}$$

Expanding the regularization term, dividing the denominator and enumerator by $\langle \psi, \psi \rangle^2$, and introducing $\boldsymbol{O}$ yields

$$\boldsymbol{O} := \frac{\nabla \psi}{\psi} = \nabla \log |\psi| \tag{1.168}$$

$$\mathcal{L} = \langle E \rangle + \lambda \left(1 - \frac{\left(1 + \boldsymbol{\delta}^T \frac{\langle \psi, \nabla \psi \rangle}{\langle \psi, \psi \rangle} \right)^2}{1 + 2\boldsymbol{\delta}^T \frac{\langle \psi, \nabla \psi \rangle}{\langle \psi, \psi \rangle} + \boldsymbol{\delta}^T \frac{\langle \nabla \psi, \nabla \psi \rangle}{\langle \psi, \psi \rangle} \boldsymbol{\delta}} \right) \tag{1.169}$$

$$= \langle E \rangle + \lambda \left(1 - \frac{\left(1 + \left\langle \boldsymbol{\delta}^T \boldsymbol{O} \right\rangle \right)^2}{1 + 2\boldsymbol{\delta}^T \langle \boldsymbol{O} \rangle + \boldsymbol{\delta}^T \left\langle \boldsymbol{O}\boldsymbol{O}^T \right\rangle \boldsymbol{\delta}} \right) + \mathcal{O}(|\delta|^3). \tag{1.170}$$

Expanding the denominator up to second order in δ (using $(1+x)^{-1} \approx 1 - x + x^2$) and multiplying all terms finally yields

$$\mathcal{L} = \langle E \rangle + \lambda \boldsymbol{\delta}^T \boldsymbol{S} \boldsymbol{\delta} + \mathcal{O}(|\delta|^3) \tag{1.171}$$

$$\boldsymbol{S} = \left\langle \boldsymbol{O}\boldsymbol{O}^T \right\rangle - \langle \boldsymbol{O} \rangle \langle \boldsymbol{O} \rangle^T \tag{1.172}$$

The regularized loss in eq. (1.171) has the same structure as the simple eq. (1.160): The original loss + a quadratic regularization term – the only difference being that this time the metric is

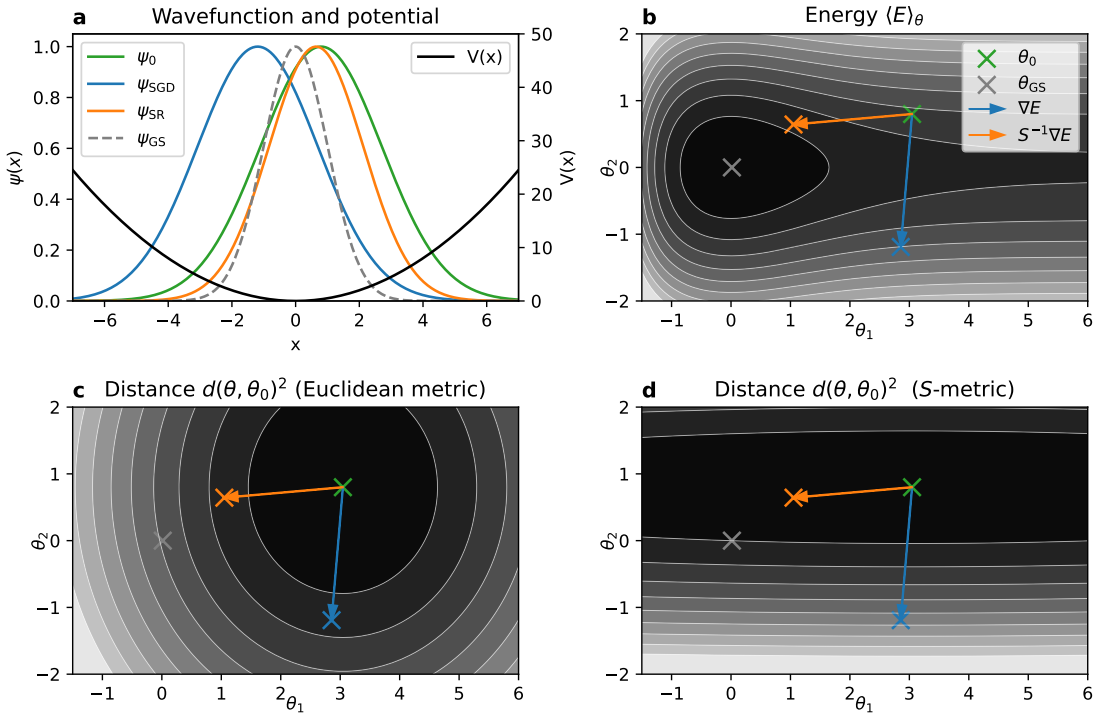

Figure 1.3: **1D toy example for 2-parameter wavefunction**: a) Plot of ground-state wavefunction ψ_{GS}, initial wavefunction ψ_0, and the resulting wavefunctions after update steps according to stochastic gradient descent (SGD) and stochastic reconfiguration (SR). b-d) Contour-plots as a function of wavefunction parameters θ_1 and θ_2 (darker colors correspond to lower values). b) Energy expectation value of corresponding wavefunction. c,d) Distance from initial parameter vector $\boldsymbol{\theta}_0$ measured in Euclidean metric and the metric induced by the preconditioner $\boldsymbol{S}$.

given by $\boldsymbol{S}$ instead of the Euclidean-norm. When minimizing this regularized loss one obtains the stochastic reconfiguration update rule (up to a factor of 2 in the learning rate)

$$\boldsymbol{\delta} = -2\lambda \boldsymbol{S}^{-1} (\nabla E) \,. \tag{1.173}$$

Note that $\boldsymbol{S}$ has been derived here using eq. (1.164) as distance metric. The same result (up to a constant factor) can be obtained by expanding the Kullback-Leibler divergence – a well known divergence to measure the distance between probability distribution – between the probability distributions $|\psi_\theta^2|$ and $|\psi_{\theta+\delta}|^2$. If the probability distributions are normalized then $\langle \boldsymbol{O} \rangle \equiv \boldsymbol{0}$, simplifying eq. (1.172) to the first term and the corresponding update rule of *natural gradient descent*.

1D toy system with 2 parameters fig. 1.3 demonstrates the effect of this preconditioning on a 1D-example (a single particle in a parabolical potential) with a wavefunction that has only two parameters:

$$\psi(x) = e^{-\frac{1}{2}\left(\frac{x-\theta_2}{2\sigma(\theta_1)}\right)^2}, \tag{1.174}$$

with the sigmoid function $\sigma(\theta) = \frac{1}{1+e^{-\theta}}$. This system has its ground-state at $\theta_1 = \theta_2 = 0$, depicted as ψ_{GS} in fig. 1.3a and the corresponding point in parameter-space $\boldsymbol{\theta}_{\text{GS}}$ in fig. 1.3b.

Starting from an arbitrary initial wavefunction ψ_0 (and corresponding parameters $\boldsymbol{\theta}_0$), two distinct new wavefunctions (and corresponding parameters) are depicted. The parameters $\boldsymbol{\theta}_{\text{SGD}}$ (and its wavefunction ψ_{SGD}) are obtained from the gradient descent update rule $\boldsymbol{\theta}_{\text{SGD}} = \boldsymbol{\theta}_0 - \lambda_{\text{SGD}} \nabla E$. The parameters $\boldsymbol{\theta}_{\text{SR}}$ (and its wavefunction ψ_{SR}) are obtained from the stochastic-reconfiguration update rule $\boldsymbol{\theta}_{\text{SR}} = \boldsymbol{\theta}_0 - \lambda_{\text{SR}} \boldsymbol{S}^{-1} \nabla E$. The learning rates λ_{SGD} and λ_{SR} are chosen such that the Euclidean distance in parameter space is identical in both case (as depicted in fig. 1.3c). However, the change of the wavefunction is markedly different: The SR-update rule leads to a much smaller change in the wavefunction (compared to the SGD update rule). This can be seen from the smaller S-distance in fig. 1.3d as well as visually when comparing ψ_{SR} and ψ_{SGD} in fig. 1.3a. Overall this leads to a lower energy after the update step (as can be seen in fig. 1.3b) and will lead to overall faster convergence towards the ground-state when iterating. The effect of the preconditioning is that the SR-update rotates the update step towards larger updates along the θ_1 parameter, which is less sensitive in this point of the parameter space.

1.8.4 Stochastic reconfiguration as a projection method

For real-valued wavefunctions there is an alternative view of eq. (1.159), which leads to the same result. It can be shown that in the limit of infinitesimal step sizes λ, stochastic reconfiguration is equivalent to applying the operator $e^{-H\lambda}$ (with the Hamiltonian H) and then projecting the resulting wavefunction onto the space spanned by ψ_θ and $\nabla\psi_\theta$. Because the operator $e^{-H\lambda}$ is identical to the time-evolution operator e^{-iHt} for an imaginary time $t = i\lambda$, this method is also known as *imaginary time evolution* in the physics community [44].

Exact imaginary time evolution leads to an exponential decay of the energy as can be seen by decomposing any initial wavefunction ψ into the eigenstates of ψ_n of H.

$$\psi = \sum_{n=0}^{\infty} c_n \psi_n \tag{1.175}$$

Applying the exponential operator t times, thus yields

$$\begin{aligned}
\psi_t &= e^{-H\lambda t}\psi && (1.176)\\
&= \sum_{n=0}^{\infty} c_n e^{-H\lambda t}\psi_n && (1.177)\\
&= \sum_{n=0}^{\infty} c_n e^{-E_n\lambda t}\psi_n && (1.178)\\
&= e^{-E_0\lambda t}\left(c_0\psi_0 + \sum_{n=1}^{\infty} c_n e^{-(E_n-E_0)\lambda t}\psi_n\right). && (1.179)
\end{aligned}$$

As $t \to \infty$ all contributions from states $i > 0$ decay exponentially, with the relevant time-scale $\frac{1}{E_n - E_0}$. Therefore as long as the initial state has non-zero overlap with the groundstate ($c_0 \neq 0$) and the groundstate is non-degenerate ($E_n > E_0 \ \forall \ n > 0$) this method will converge to the groundstate. The following proof that stochastic reconfiguration is equivalent to imaginary time evolution + projection is adapted from [41].

For a small step size λ the exponential can be expanded up to first order in λ.

$$e^{-H\lambda} \approx 1 - H\lambda \tag{1.180}$$

At every optimization step t, one implicitly performs 2 steps: Imaginary time evolution of ψ, yielding $\hat{\psi}$, and projection of $\hat{\psi}$ onto the basis spanned by ψ and its derivatives.

$$\hat{\psi} = (1 - H\lambda)\psi \qquad \text{Imag. time evolution} \tag{1.181}$$

$$\hat{\psi} \stackrel{!}{=} C\psi + \sum_{\mu=1}^{N_{\text{param}}} \delta_\mu \frac{\partial \psi}{\partial \theta_\mu} \qquad \text{Projection} \tag{1.182}$$

To find the expansion coefficients C and δ_μ one takes the scalar products of $\hat{\psi}$ with all basis functions, yielding eq. (1.183) for the scalar product with ψ and N_{param} equations eq. (1.184) for each derivative w.r.t. θ_ν.

$$\int \psi(1 - H\lambda) d\boldsymbol{r} = C \int \psi^2 d\boldsymbol{r} + \sum_{\mu=1}^{N_{\text{param}}} \delta_\mu \int \psi \frac{\partial \psi}{\partial \theta_\mu} d\boldsymbol{r} \tag{1.183}$$

$$\int \frac{\partial \psi}{\partial \theta_\nu}(1 - H\lambda)\psi d\boldsymbol{r} = C \int \frac{\partial \psi}{\partial \theta_\nu}\psi d\boldsymbol{r} + \sum_{\mu=1}^{N_{\text{param}}} \delta_\mu \int \frac{\partial \psi}{\partial \theta_\mu}\frac{\partial \psi}{\partial \theta_\mu} d\boldsymbol{r} \tag{1.184}$$

Dividing each equation by the normalization $\int \psi^2 d\boldsymbol{r}$ allows to express the equations in terms of expectation values

$$1 - \langle E \rangle = C + \sum_{\mu=1}^{N_{\text{param}}} \delta_\mu \langle O_\mu \rangle \tag{1.185}$$

$$\langle O_\nu \rangle - \langle O_\nu E \rangle = C \langle O_\nu \rangle + \sum_{\mu=1}^{N_{\text{param}}} \delta_\mu \langle O_\nu O_\mu \rangle \tag{1.186}$$

with the local energy $E(\boldsymbol{r}) = \frac{H\psi}{\psi}(\boldsymbol{r})$ and $O_\nu(\boldsymbol{r}) = \frac{\partial \log \psi}{\partial \theta_\nu}(\boldsymbol{r})$. Multiplying eq. (1.185) with $\langle O_\nu \rangle$ and subtracting from each of eq. (1.186) yields

$$-\Big(\langle O_\nu E \rangle - \langle E \rangle \langle O_\nu \rangle\Big) = \sum_{\mu=1}^{N_{\text{param}}} \Big(\langle O_\nu O_\mu \rangle - \langle O_\nu \rangle \langle O_\mu \rangle\Big)\delta_\mu \tag{1.187}$$

The LHS corresponds exactly to the gradient of the energy (see eq. (1.124)) and the expectation values on the RHS correspond to $\boldsymbol{S}$. Casting it in matrix form, again reveals the stochastic reconfiguration update rule

$$-\nabla E = \boldsymbol{S}\boldsymbol{\delta} \quad \Longrightarrow \quad \boldsymbol{\delta} = -\boldsymbol{S}^{-1}\nabla E \tag{1.188}$$

1.8.5 Stochastic reconfiguration in practice

So far the matrix $\boldsymbol{S}^{-1}$ was always assumed as given, but computing it in practice is hard. The first complication is that when $\boldsymbol{S}$ is being estimated from N_{s} samples it is at most of rank N_{s}:

$$S_{\mu\nu} = \langle O_\mu O_\nu \rangle - \langle O_\mu \rangle \langle O_\nu \rangle \tag{1.189}$$

$$= \Big\langle \Big(O_\mu - \langle O_\mu \rangle\Big)\Big(O_\nu - \langle O_\nu \rangle\Big) \Big\rangle \tag{1.190}$$

$$\approx \sum_{n=1}^{N_{\text{s}}} \Big(O_\mu(\boldsymbol{r}_n) - \langle O_\mu(\boldsymbol{r}_n) \rangle\Big)\Big(O_\nu(\boldsymbol{r}_n) - \langle O_\nu(\boldsymbol{r}_n) \rangle\Big) \tag{1.191}$$

Therefore for typical values of $N_\mathrm{s} \approx \mathcal{O}(10^3)$ and $N_\mathrm{param} \approx \mathcal{O}(10^6)$, $\boldsymbol{S}$ is rank deficient and cannot be inverted. This is typically addressed via Tikhonov regularization with a small damping constant ϵ:

$$\boldsymbol{S}_\mathrm{reg} = \boldsymbol{S} + \epsilon \mathbb{1}_{N_\mathrm{s}} \tag{1.192}$$

Another approach is to estimate $\boldsymbol{S}$ not only from the current batch, but as a moving average of the estimates from past batches, thus increasing the rank of the estimator. This helps to reduce Monte Carlo noise in the estimation but typically still requires regularization to avoid a singular matrix $\boldsymbol{S}$.

The second complication arises due to the size of $\boldsymbol{S}$, which is of dimension $N_\mathrm{param} \times N_\mathrm{param}$. Therefore, for a neural network wavefunction with $\mathcal{O}(10^6)$ parameters, even storing this matrix with $\mathcal{O}(10^{12})$ elements becomes impossible. Even worse, this large matrix must be inverted, an operation that has computational cost $\mathcal{O}({N_\mathrm{param}}^3)$ using Gaussian elimination. But don't panic [42], there are two viable routes in practice: Find a (sparse) approximation of $\boldsymbol{S}$ and invert it exactly, or find a way to approximately invert $\boldsymbol{S}$ without fully materializing $\boldsymbol{S}$.

KFAC KFAC (Kronecker-Factored Approximation of Curvature) [43] is of the first type, making two approximations to $\boldsymbol{S}$. First it assumes that there are no dependencies between parameters belonging to different layers of the neural network, effectively assuming $\boldsymbol{S}$ to be block-diagonal. Second it assumes that each remaining block can be expressed as an eponymous Kronecker product of two smaller matrices. This allows inversion of the approximated $\boldsymbol{S}$ via inversion of many small matrices, which is computationally feasible even for networks involving millions of parameters. KFAC has first been applied to neural network wavefunctions by Pfau et al. [10] and since been used widely throughout the neural wavefunction community, including all the papers that form this thesis. While it is computationally efficient and can yield good results, it has two downsides in practice. First, it involves approximations that cannot be systematically improved upon. Second, the optimizer does not only require access to the wavefunction, energies and their gradients, but also requires access to intermediate activations and gradients of the model. This can introduce substantial complexity for practical implementation and leads to some unwanted coupling between the wavefunction model and the optimizer.

Conjugate Gradient An alternative approach is to make no approximations to $\boldsymbol{S}$, but to only invert it approximately. One common approach is to use the Conjugate Gradient (CG) method to compute $\boldsymbol{S}^{-1}\nabla E$ without materializing $\boldsymbol{S}$. CG only requires the repeated evaluation of the matrix-vector product $\boldsymbol{S}\boldsymbol{x}$ for arbitrary vectors $\boldsymbol{x}$. This can be obtained by a vector-jacobian-product (VJP) followed by a jacobian-vector-product (JVP), which are implemented using back-propagation and forward-mode differentiation respectively and don't require materializing the full jacobian.

Exploiting low-rank nature of $\boldsymbol{S}$ A very promising approach is to use the fact the regularized $\boldsymbol{S}_\mathrm{reg}$ is a sum of a full-rank, but easy to invert diagonal matrix ($\epsilon \mathbb{1}$), and a low-rank matrix $\boldsymbol{S}$. Inversion of $\boldsymbol{S}_\mathrm{reg}$ can therefore be done using the Sherman-Morrison-Woodbury formula [45], which only requires the inversion of a $N_\mathrm{s} \times N_\mathrm{s}$ matrix. This forms the basis for the MinSR optimizer introduced by Rende et al. [46] and a variation named SPRING [47].

Explicit imaginary time evolution The observation that stochastic reconfiguration is equivalent to imaginary time evolution with subsequent projection (section 1.8.4) can also be used to directly design an optimizer which avoids $\boldsymbol{S}$. This is achieved by alternating imaginary time

evolution steps, with explicit projection:

$$\hat{\psi} = (1 - H\lambda)\psi_\theta \qquad \text{Imag. time evolution} \tag{1.193}$$

$$\theta \leftarrow \min_{\theta'} \int |\hat{\psi} - \psi_{\theta'}|^2 d\boldsymbol{r} \qquad \text{Projection using gradient descent} \tag{1.194}$$

The minimization in eq. (1.194) can be performed using any gradient-based optimizer, e.g. stochastic gradient descent or Adam. This method has been dubbed *Supervised Wavefunction Optimization* [48]. While it avoids the preconditioning step, it comes at the extra cost of solving a separate minimization problem (eq. (1.194)) at every step of the main optimizer.

1.8.6 Supervised pretraining of orbitals

So far only variational minimization of the energy, which requires no external reference data, has been discussed. While variational optimization from randomly initialized parameters is in principle sufficient to obtain groundstate wavefunctions, convergence can initially be slow and unstable. This is because the initial wavefunction may be far from the groundstate wavefunction and the samples drawn from it may initially insufficiently cover the relevant regions in configurational space. For example, there may initially be very few samples close to the nuclei, although this is typically the region with the highest electron density. To obtain a better initialization of the parameters it has been proposed to pretrain the wavefunction against a given reference wavefunction, to obtain a better starting point for subsequent variational optimization [10]. This is achieved by minimizing the residual between the orbitals ϕ of the neural network wavefunction and some reference orbitals ϕ_{ref}, yielding the pretraining loss

$$\mathcal{L}^{\text{supervised}} = \sum_{i,k=1}^{n_{\text{el}}} \left\langle \left| \phi_{ik}^{\text{ref}}(\boldsymbol{r}) - \phi_{ik}(\boldsymbol{r}) \right|^2 \right\rangle_{\boldsymbol{r} \sim |\psi_{\text{ref}}|^2}. \tag{1.195}$$

While pretraining can substantially accelerate subsequent variational optimization it risks biasing the neural network towards a solution that is similar to the reference wavefunction. It has been shown empirically that excessive pretraining can therefore deteriorate final accuracy [7]. It would also seem intuitive that pretraining against a more accurate reference wavefunction ψ_{ref} would lead to better results, but it has also been shown empirically that this is not necessarily true: Cheaper, less accurate reference wavefunctions, such as Hartree-Fock, can lead to better final variational energies compared to more accurate reference wavefunctions like CASSCF [7].

Chapter 2

Summary of Publications

The following briefly summarizes the papers, which form the core of this dissertation. Given that the field of neural network wavefunctions has rapidly evolved over the past years, some of the presented methods have been been superseded by more recent, better approaches. Therefore for every paper a brief summary as well as some hindsight comments are given, highlighting for each paper the findings which are still relevant and the things one would do differently today.

2.1 Gold-standard solutions to the Schrödinger equation using deep learning: How much physics do we need?

2.1.1 Paper summary

This paper (see appendix A, [7]) is the only paper among the included works which does not concern itself with transferable wavefunctions, but rather focuses on solving the Schrödinger equation for a single molecule at a time. The work is heavily based on the two major architectures which had been published at that time: FermiNet [10, 49], using a large MLP-based embedding combined with simple exponential envelopes, and PauliNet [12], using a graph convolutional neural network and elaborate envelopes based on Hartree-Fock orbitals. The initial hypothesis was that combining the FermiNet embedding (having substantially more parameters than the PauliNet embedding) with the PauliNet orbitals (including more physical intuition) would yield a superior wavefunction. Interestingly the opposite turned out to be the case: Using a graph convolutional neural network as the embedding, combined with the simple exponential FermiNet envelopes yielded a better architecture than either FermiNet or PauliNet. We demonstrated this by computing for several small atoms and molecules the lowest variational energies known at the time, hence the perhaps overly bold title of "Gold-standard solutions". The paper systematically analyzes which changes contribute how much to the improvements in energy as depicted in fig. 2.1.

The subtitle "How much physics do we need?" allures to the counterintuitive observation that using more intuition from physics can actually deteriorate accuracy. This is demonstrated for two separate aspects: First, using physics inspired orbitals from a mean-field method performs worse than using simple exponential envelopes. Second, more accurate supervised pretraining of the wavefunction (either through more steps or a more accurate reference method) can actually

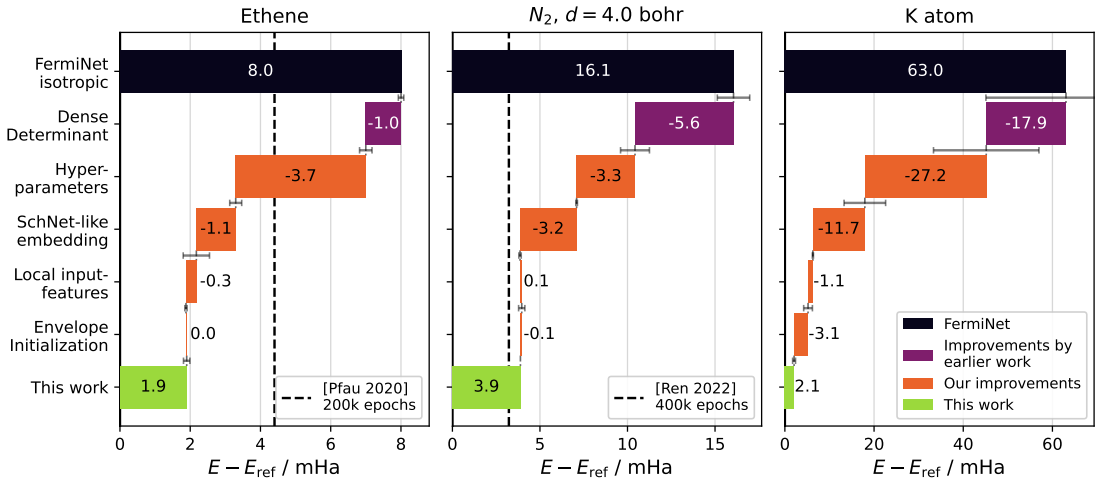

Figure 2.1: **Main result of [7], improvements in accuracy for three different systems**. This plot depicts the obtained groundstate energy for three distinct systems and breaks down the contribution of various changes to the increased accuracy. Improvements presented in the paper include an optimized set of hyperparameters, an embedding based on graph convolutions (SchNet-like embedding), a local coordinate system, and improved parameter initialization for heavier atoms.

deteriorate the final accuracy after variational training.

2.1.2 Hindsight comments

Most observations made in this paper still hold true and have informed subsequent work. Combining exponential envelopes with an expressive embedding (which improves upon FermiNet by using some form of position dependent message passing) is still state of the art today. Also subsequent work by the authors of PauliNet [18, 50] has used this type of architecture, instead of using CASSCF orbitals. The observation that excessive supervised pretraining of the wavefunction can induce biases that are hard to overcome in variational observation has also been seen in [34].

In retrospect, there are three aspects which would be presented somewhat differently today: First, hyperparameters do not necessarily have a 1:1 correspondence across codebases. Therefore the presented "improved" hyperparameters for FermiNet – while yielding substantially better results in the DeepErwin codebase – may not lead to similar improvements in the original FermiNet codebase, a point which was not clear at the time and should have been more clearly highlighted. Second, the local coordinate systems, which were proposed to aid generalization in subsequent multi-geometry work, have turned out to be too restrictive to express arbitrary wavefunctions and have therefore not been used in later work. Lastly, the paper strongly emphasizes having obtained the best published variational energies across several systems. While this was factually true at the time of publication, subsequent architectures such as PsiFormer [15] have obtained even lower energies. In such a fast-paced field, calling results "gold-standard" appear ill-advised in hindsight, even if true at the time.

2.2 Solving the electronic Schrödinger equation for multiple nuclear geometries with weight-sharing deep neural networks

2.2.1 Paper summary

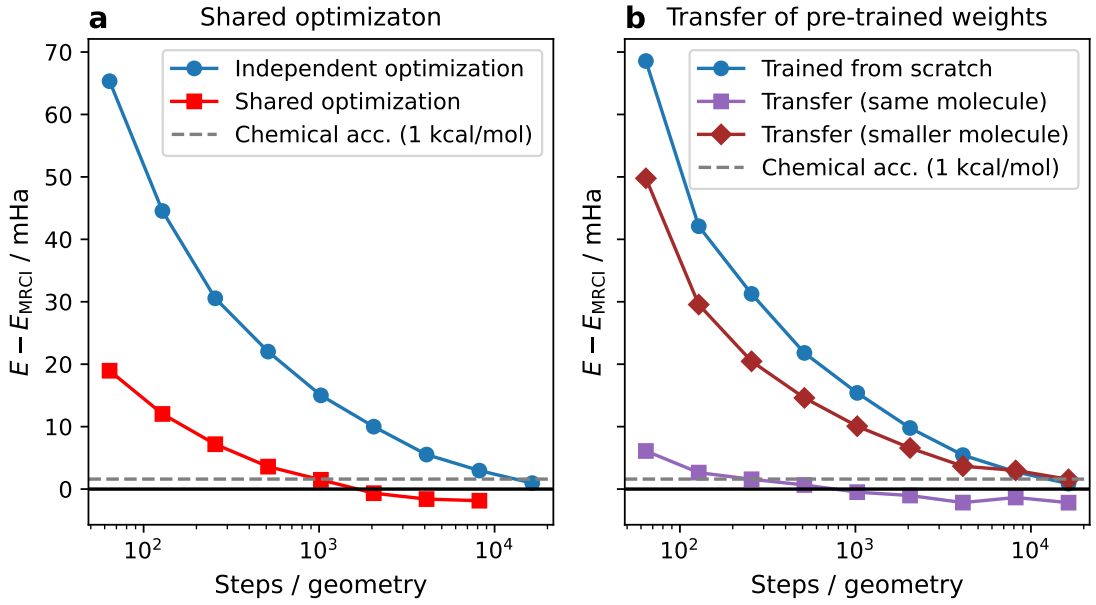

Figure 2.2: **Main result of [13], shared optimization and transfer of pre-trained weights to compute energies of ethene**. Average energy relative to an accurate reference calculation (MRCI: Multi reference Configurational Interaction) for the potential energy surface of Ethene as a function of optimization steps. **a)** Training a single model to represent the wavefunction for many geometries simultaneously substantially accelerates energy convergence compared to performing independent calculations. **b)** Transferring these pretrained weights to new geometries again leads to substantial speed-ups. This works well when transferring from different geometries of the same molecule, but poorly when transferring weights pretrained on a smaller molecule (methane).

This paper (see appendix B, [13]) marks the first in a series of papers developing neural network wavefunctions, which are transferable across systems. Similarly to later works it proposes to use a single neural network embedding for multiple geometries. Unlike later works, it does not yet address the challenge of generating orbitals for different systems, but instead uses different trainable orbital parameters for every geometry. The vast majority of trainable parameters ($\approx 95\%$) is shared across geometries, while a few weights ($\approx 5\%$) are chosen differently for each geometry.

The paper establishes two important ways to test transferable neural wavefunctions: The first is shared training, where a single model with shared parameters is optimized to simultaneously represent the groundstate wavefunction for several different geometries of a molecule (fig. 2.2a). The second is the transfer of such a pretrained wavefunction to new, unseen systems (fig. 2.2b).

The approach is demonstrated on toy-systems comprised of Hydrogen-atoms and several distorted geometries of Ethene. In both cases shared optimization accelerates optimization by $\approx 10\times$ compared to independent optimization of a separate wavefunction for each system. When transferring weights from a pretrained wavefunction to new geometries of the same molecule, even larger speed-ups are observed.

2.2.2 Hindsight comments

The main idea of the paper – having a single wavefunction represent multiple geometries at once – has proven to be a fruitful research direction. Both applications of such a transferable wavefunction – shared optimization of a wavefunction for multiple geometries and transfer of the parameters to new systems – have been further explored and improved upon in future work. Furthermore the test systems (in particular ethene, twisted around the C=C bond) have been used as a baseline for future work.

On the other hand, the specific architecture proposed turned out to be a poor choice. On the one hand, it is based on the PauliNet architecture, which is based on incorporating Hartree-Fock orbitals, an approach that later turned out to be inferior to the simpler FermiNet orbitals. On the other hand, the necessity of having some of the weights be geometry specific limits the transferability and usefulness of such a wavefunction. Concurrent work by Gao et al. [11], built on the FermiNet architecture, managed to improve on several results presented in this paper and managed to train a single wavefunction across geometries without the need for separate parameters per geometry.

2.3 Towards a transferable fermionic neural wavefunction for molecules

2.3.1 Paper summary

This paper (see appendix C, [20]) is one of the core contributions of this dissertation. It builds on the previous work (section 2.2), but substantially extends it by introduction of the transferable atomic orbitals (see section 1.6.4), a method to generate neural network orbitals based on features extracted from a mean-field calculation. This change is significant because it no longer requires separate trainable parameters for each system and enables training of a single model across different molecules. The paper demonstrates the transfer capabilities of this approach on simple test systems, comparing favorably against own prior work, as well as a concurrent proposal by Gao et al. [14].

Optimizing this transferable wavefunction across diverse dataset of small molecules yields a pretrained model, which can be used as an effective initialization for the wavefunction of new, unseen molecules. Figure 2.3 demonstrates that fine-tuning such a pretrained wavefunction model converges much faster in energy compared to optimizing the wavefunction from scratch. Compared to fig. 2.2b, where the transfer only worked well when transferring to similar geometries of the same molecule, the new approach works much better. When transferring the model to new, unseen molecules which are of similar size as the ones in the training set (fig. 2.3a) it can reach accuracies of ≈ 10 mHa with minimal training. Even transferring to new molecules which are larger than the largest ones in the pretraining set (fig. 2.3b), yields substantial speed-ups in convergence for these larger systems.

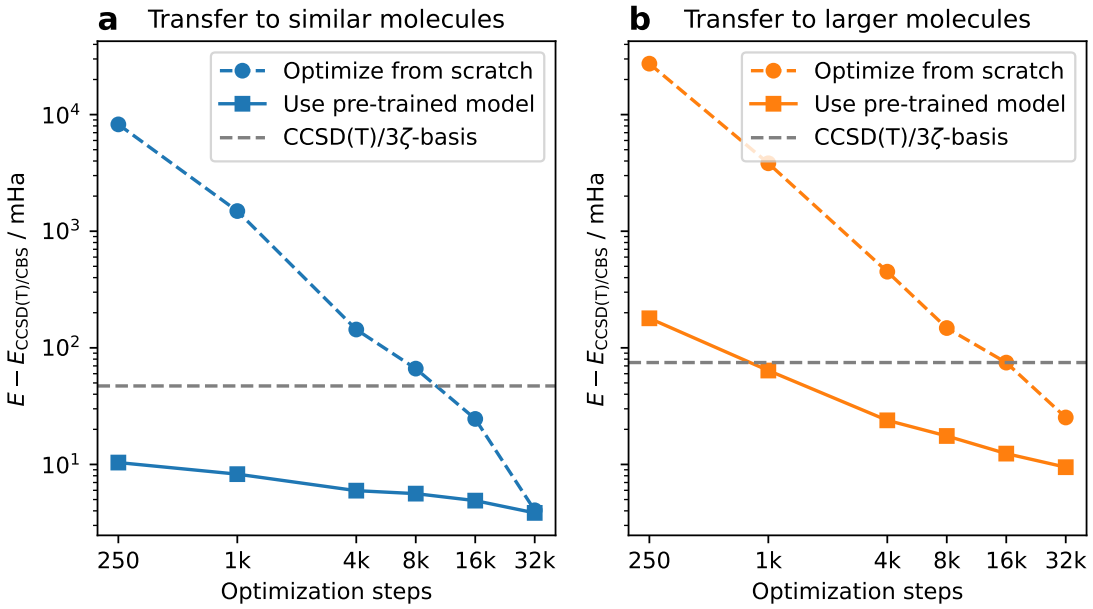

Figure 2.3: **Main result of [20], transfer of pretrained wavefunction model to new, unseen molecules**. Energy error relative to an accurate reference calculation (CCSD(T)-CBS) as a function of optimization steps. Dashed line represents training of a new wavefunction model, solid line represents fine-tuning of a wavefunction model which has been pretrained on a large dataset of molecules. **a)** Performance when transferring the model to new molecules which are similar (same number of non-hydrogen atoms) to the ones in the pretraining dataset. **b)** Performance when transferring the model to new molecules which are larger than any molecule in the pretraining dataset (up to 4 non-hydrogen atoms vs. maximum of 3 in the pretraining dataset).

2.3.2 Hindsight comments

The approach proposed in the paper – using mean-field orbitals as inputs to obtain correlated neural network orbitals – has served the authors well for subsequent work. At the same time there are minor shortcomings which have been addressed in follow-up work, substantially increasing the accuracy.

First, the embedding network used in this paper (essentially FermiNet with added convolutional filters) is insufficiently expressive to model embeddings for many different molecules. Second, the elements of the Slater matrix are essentially obtained as the inner product between an electron embedding $\boldsymbol{h}_i$ and an orbital embedding $\boldsymbol{c}_{kJ}$

$$\Phi_{ik} = \sum_{J=1}^{N_{\mathrm{nuc}}} \langle \boldsymbol{h}_i, \boldsymbol{c}_{kJ} \rangle \varphi_{kiJ}, \tag{2.1}$$

with i, k enumerating electrons and orbitals respectively, J enumerating nuclei and φ representing the envelope. When applying this ansatz to solids (see section 2.5) it has become apparent that this approach cannot represent arbitrary mean-field wavefunctions and is therefore insufficiently expressive. This can be overcome by using embeddings $\boldsymbol{h}_{iJ}$, which represent the interaction

between an electron i and a nucleus J. This minimal change,

$$\Phi_{ik} = \sum_{J=1}^{N_{\text{nuc}}} \langle \boldsymbol{h}_{iJ}, \boldsymbol{c}_{kJ} \rangle \varphi_{kiJ}, \tag{2.2}$$

exchanging $\boldsymbol{h}_i$ for $\boldsymbol{h}_{iJ}$ allows expressing any mean-field wavefunction directly from localized orbital features $\boldsymbol{c}_{kJ}$ thus simplifying the architecture and increasing accuracy. While this has only been tested for periodic systems so far, it seems likely that this change would also increase the accuracy for molecules.

2.4 Variational Monte Carlo on a Budget — Fine-tuning pre-trained Neural Wavefunctions

2.4.1 Paper summary

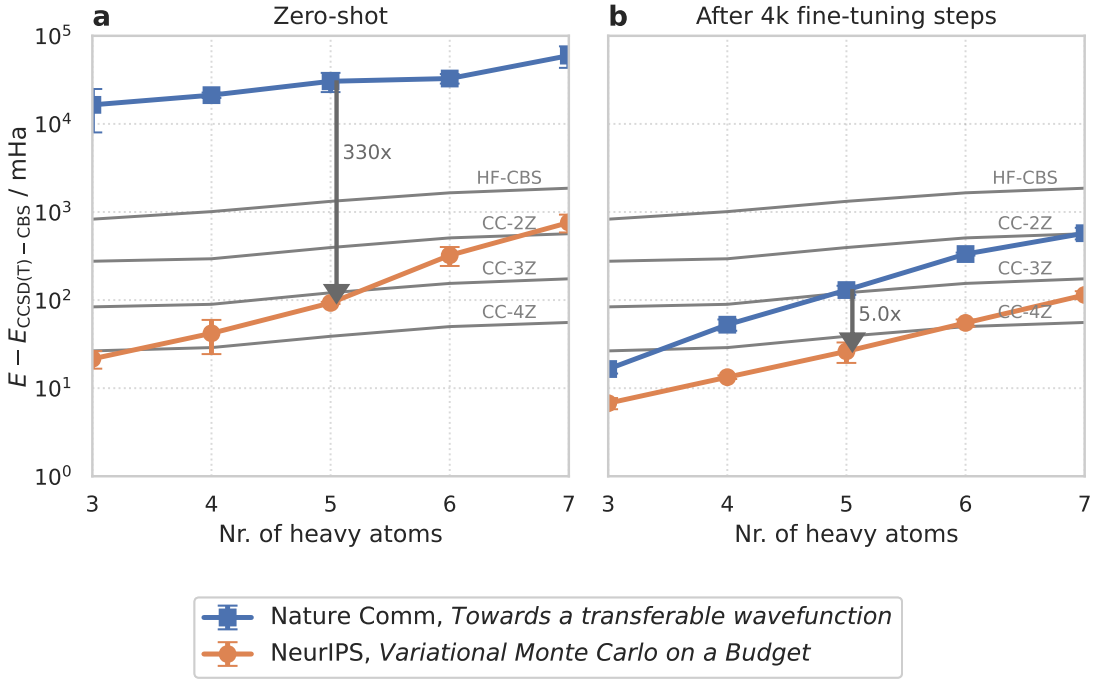

Figure 2.4: **Main result of [33], transfer of pretrained wavefunction to new, unseen molecules**. Accuracy of transferred wavefunctions (blue squares: section 2.3, orange circles: section 2.4) when applied to new molecules as a function of the size of the new molecules (number of non-hydrogen atoms). **a)**Energy error when the transferred wavefunction is not further optimized for the new molecules. **b)** Energy error after additional 4000 optimization steps. Horizontal lines correspond to Coupled Cluster results (CCSD(T)) in increasing basis sets (2ζ-4ζ).

This paper (see appendix D, [33]) improves on several of the shortcomings of the previous paper (section 2.3). The key difference is that it replaces Hartree-Fock, which was used to generate orbital features $\boldsymbol{c}_k$, by a separate neural network based on PhisNet [51]. This additional network

is a graph neural network, which operates purely on the nuclear coordinates and is pretrained to predict the overlap and Fock-matrix, key intermediate quantities in a mean field calculation. The orbital features $\boldsymbol{c}_k$ are obtained via diagonalization of the predicted Fock-matrix, avoiding the self-consistent iterations required for a typical mean-field calculation. Avoiding the Hartree-Fock calculation yields several benefits: For a given computational budget, more unique molecules and geometries can be considered, since computing the orbital features is substantially accelerated. This allows variational pretraining of the wavefunction on a much larger and more diverse dataset of molecules, containing 100 distinct molecules with 700 distinct geometries. Additionally, the pretraining dataset can be augmented by random rotations and distortions of the molecules. The auxiliary graph neural network also generates features for each atom as a byproduct, which can be used as inputs to the electron embedding network, further improving expressiveness and generalization.

Using this improved embedding and pretraining on a larger, more diverse dataset yields a transferable wavefunction model, which – when applied to new, unseen molecules – yields surprisingly accurate energies. Figure 2.4 plots the accuracy of the pretrained wavefunction as a function of the size of the molecules in the test set. Independent of the amount of additional optimization steps, the new model obtained in [33] yields substantially more accurate energies compared to previous work [13]. In particular in the zero-shot regime, i.e. without any subsequent optimization of the wavefunction on the new molecule, the model yields energies that are more accurate than CCSD(T) in a 3-ζ basis set. This even holds for molecules that are slightly larger than the largest molecules present in the pretraining dataset.

2.4.2 Hindsight comments

This work still presents a strong baseline for a transferable neural network wavefunction for small molecules, in particular in the zero-shot regime. Even if the model is eventually surpassed by more accurate architectures, the dataset and data augmentation methodology used for variational pretraining of the wavefunction appear to be useful contributions in their own right.

The main drawback of this work is the relatively high complexity of the model. In particular the PhisNet based model, which predicts orbital coefficients $\boldsymbol{c}_k$, introduces substantial additional complexity for two reasons: First, it requires separate pretraining on a mean-field dataset, and this model must itself be able to generalize across molecules. Second the PhisNet model explicitly preserves SE(3)-equivariance of the orbital coefficients under rotation or translation of the molecule. While this is elegant and leads to very accurate results, it requires a more complex architecture, adding further complexity. Ultimately it would be desirable to also update the parameters of the orbital model during variational training, to enable end-to-end unsupervised optimization of the wavefunction. While this was not done in this work to reduce complexity, it is in principle straightforward and could lead to more accurate energies.

2.5 Transferable Neural Wavefunctions for Solids

2.5.1 Paper summary

This paper (see appendix E, [34]), which at the time of writing this thesis is currently under peer review, applies the approach of transferable wavefunctions developed in the previous papers to crystalline solids. neural network wavefunctions can in principle be extended straightforwardly to model the wavefunction for crystalline systems, but solids pose some additional challenges. Crystalline solids are composed of atoms arranged on a periodic lattice, which for practical

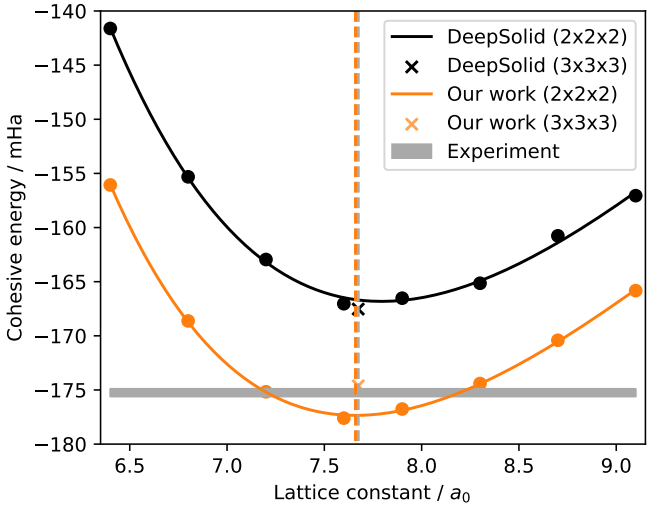

Figure 2.5: **Main result of [34], Potential Energy Surfaces of a lithium hydride crystal**. Cohesive energy of LiH as function of the lattice constant. Lines correspond to energies calculated in a $2 \times 2 \times 2$ supercell, crosses correspond to energies calculated in a $3 \times 3 \times 3$ supercell. Energies obtained from our work in the $3 \times 3 \times 3$ supercell closely match the experimental cohesive energy of the (gray horizontal band).

purposes extends almost infinitely in all directions. Simulating an ideal crystal therefore requires in principle simulation of an infinitely extended system. This is in practice approximated by computing the energy of a supercell, a simulation cell containing a finite number of repetitions of the lattice unit cell. This truncation typically leads to large energy errors - known as *finite size errors* - and considerable effort has been dedicated to minimizing these finite size errors.

First, instead of open boundary conditions, periodic or *twisted boundary conditions* [52] are used. Here the translation of an electron by a supercell lattice vector $\boldsymbol{L}^{\text{sc}}$ must yield the same wavefunction up to phase factor, specified by a parameter $\boldsymbol{k}$

$$\psi(\boldsymbol{r}_1, \ldots, \boldsymbol{r}_i + \boldsymbol{L}^{\text{sc}}, \ldots \boldsymbol{r}_{n_{\text{el}}}) = e^{i\langle \boldsymbol{k}, \boldsymbol{L}^{\text{sc}} \rangle} \psi(\boldsymbol{r}_1, \ldots, \boldsymbol{r}_i, \ldots \boldsymbol{r}_{n_{\text{el}}}). \tag{2.3}$$

Averaging results for many different twists $\boldsymbol{k}$ yields an energy estimator which has smaller finite size errors. Second, computations are typically done on increasingly large supercells (and correspondingly decreasing finite size errors) to estimate and subtract these errors. Both approaches require many similar calculations for distinct systems, varying the boundary conditions specified by $\boldsymbol{k}$, the system size specified by the supercell, and furthermore potentially varying the geometry of the crystal.

This practical challenge has so far limited the applications of neural network wavefunctions mostly to model systems [53, 54]. A notable exception is the work by Li et al. [55], which applied a FermiNet-like architecture (dubbed DeepSolid) to real solids, but required up to 80,000 GPU-hours to compute energies for a single system. In our paper [34] we show that using the transferable atomic orbitals developed for molecules, can be applied to solids. Because the transferable wavefunction allows optimization of a single wavefunction for many different systems at once (in particular across geometries, supercell sizes and boundary conditions $\boldsymbol{k}$), our approach can model solids at substantially lower cost. Figure 2.5 shows as an example the potential energy

surface of lithium hydride. We find that our approach yields energies which improve on DeepSolid by $\approx 10\,\text{mHa}$ and obtains excellent agreement with experiments. Most importantly though, we obtain these more accurate results at approximately $50\times$ lower computational cost.

2.5.2 Hindsight comments

Since this paper is very recent, there does not yet exist a proper hindsight view. One takeaway from the project was that, beyond the domain knowledge required for studying molecules, modelling of solids requires an even larger amount of specialized tools and tricks. In particular there are several layers of corrections required to be able to compare against experiments: Twist averaging to reduce finite size errors in the kinetic energy [52], structure factor based corrections to reduce finite size errors in the potential energy [56], and zero point vibrational energies. While real solids form an important and highly interesting class of systems, the inherently large supercells required to study them substantially increase the difficulty of method development for these systems.

Bibliography

[1] Paul Adrien Maurice Dirac. "Quantum Mechanics of Many-Electron Systems". In: *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character* 123.792 (Apr. 1929), pp. 714–733. DOI: 10.1098/rspa.1929.0094.

[2] Norbert Schuch and Frank Verstraete. "Computational Complexity of Interacting Electrons and Fundamental Limitations of Density Functional Theory". In: *Nature Physics* 5.10 (Oct. 2009), pp. 732–735. DOI: 10.1038/nphys1370.

[3] W. Kohn and L. J. Sham. "Self-Consistent Equations Including Exchange and Correlation Effects". In: *Physical Review* 140.4A (Nov. 1965), A1133–A1138. DOI: 10.1103/PhysRev.140.A1133.

[4] Stephen J. Gustafson and Israel Michael Sigal. *Mathematical Concepts of Quantum Mechanics*. Universitext. Cham: Springer International Publishing, 2020. DOI: 10.1007/978-3-030-59562-3.

[5] Attila Szabo and Neil S. Ostlund. *Modern Quantum Chemistry: Introduction to Advanced Electronic Structure Theory*. Courier Corporation, July 1996. ISBN: 978-0-486-69186-2.

[6] J. C. Slater. "The Theory of Complex Spectra". In: *Physical Review* 34.10 (Nov. 1929), pp. 1293–1322. DOI: 10.1103/PhysRev.34.1293.

[7] Leon Gerard et al. "Gold-Standard Solutions to the Schrödinger Equation Using Deep Learning: How Much Physics Do We Need?" In: *Advances in Neural Information Processing Systems*. Oct. 2022.

[8] Benjamin P. Pritchard et al. "New Basis Set Exchange: An Open, Up-to-Date Resource for the Molecular Sciences Community". In: *Journal of Chemical Information and Modeling* 59.11 (Nov. 2019), pp. 4814–4820. DOI: 10.1021/acs.jcim.9b00725.

[9] G. Cybenko. "Approximation by Superpositions of a Sigmoidal Function". In: *Mathematics of Control, Signals and Systems* 2.4 (Dec. 1989), pp. 303–314. DOI: 10.1007/BF02551274.

[10] David Pfau et al. "Ab Initio Solution of the Many-Electron Schrödinger Equation with Deep Neural Networks". In: *Phys. Rev. Res.* 2.3 (Sept. 2020), p. 033429. DOI: 10.1103/PhysRevResearch.2.033429.

[11] Nicholas Gao and Stephan Günnemann. "Ab-Initio Potential Energy Surfaces by Pairing GNNs with Neural Wave Functions". In: *arXiv:2110.05064 [physics]* (Nov. 2021). eprint: 2110.05064.

[12] Jan Hermann, Zeno Schätzle, and Frank Noé. "Deep-Neural-Network Solution of the Electronic Schrödinger Equation". In: *Nature Chemistry* 12.10 (Oct. 2020), pp. 891–897. DOI: 10.1038/s41557-020-0544-y.

[13] Michael Scherbela et al. "Solving the Electronic Schrödinger Equation for Multiple Nuclear Geometries with Weight-Sharing Deep Neural Networks". In: *Nature Computational Science* 2.5 (May 2022), pp. 331–341. DOI: 10.1038/s43588-022-00228-x.

[14] Nicholas Gao and Stephan Günnemann. *Generalizing Neural Wave Functions*. Feb. 2023. DOI: 10.48550/arXiv.2302.04168.

[15] Ingrid von Glehn, James S. Spencer, and David Pfau. "A Self-Attention Ansatz for Ab-initio Quantum Chemistry". In: *The Eleventh International Conference on Learning Representations*. Sept. 2022.

[16] Nicholas Gao and Stephan Günnemann. "Sampling-Free Inference for Ab-Initio Potential Energy Surface Networks". In: (May 2022). DOI: 10.48550/arXiv.2205.14962.

[17] Jie Zhou et al. "Graph Neural Networks: A Review of Methods and Applications". In: *AI Open* 1 (Jan. 2020), pp. 57–81. DOI: 10.1016/j.aiopen.2021.01.001.

[18] Z. Schätzle et al. "DeepQMC: An Open-Source Software Suite for Variational Optimization of Deep-Learning Molecular Wave Functions". In: *The Journal of Chemical Physics* 159.9 (Sept. 2023), p. 094108. DOI: 10.1063/5.0157512.

[19] Tosio Kato. "On the Eigenfunctions of Many-Particle Systems in Quantum Mechanics". In: *Commun. Pure Appl. Math.* 10.2 (1957), pp. 151–177. DOI: 10.1002/cpa.3160100201.

[20] Michael Scherbela, Leon Gerard, and Philipp Grohs. "Towards a Transferable Fermionic Neural Wavefunction for Molecules". In: *Nature Communications* 15.1 (Jan. 2024), p. 120. DOI: 10.1038/s41467-023-44216-9.

[21] Ashish Vaswani et al. *Attention Is All You Need*. June 2017. DOI: 10.48550/arXiv.1706.03762. eprint: 1706.03762.

[22] Tom B. Brown et al. "Language Models Are Few-Shot Learners". In: (May 2020). DOI: 10.48550/arXiv.2005.14165.

[23] Thomas D. Barrett, Aleksei Malyshev, and A. I. Lvovsky. "Autoregressive Neural-Network Wavefunctions for Ab Initio Quantum Chemistry". In: *Nature Machine Intelligence* 4.4 (Apr. 2022), pp. 351–358. DOI: 10.1038/s42256-022-00461-z.

[24] Jiequn Han, Linfeng Zhang, and Weinan E. "Solving Many-Electron Schrödinger Equation Using Deep Neural Networks". In: *Journal of Computational Physics* 399 (Dec. 2019), p. 108929. DOI: 10.1016/j.jcp.2019.108929.

[25] Jack Richter-Powell et al. *Sorting Out Quantum Monte Carlo*. Nov. 2023. DOI: 10.48550/arXiv.2311.05598. eprint: 2311.05598.

[26] Nicholas Gao and Stephan Günnemann. *Neural Pfaffians: Solving Many Many-Electron Schrödinger Equations*. May 2024. DOI: 10.48550/arXiv.2405.14762. eprint: 2405.14762.

[27] J. M. Foster and S. F. Boys. "Canonical Configurational Interaction Procedure". In: *Reviews of Modern Physics* 32.2 (Apr. 1960), pp. 300–302. DOI: 10.1103/RevModPhys.32.300.

[28] János Pipek and Paul G. Mezey. "A Fast Intrinsic Localization Procedure Applicable for Ab Initio and Semiempirical Linear Combination of Atomic Orbital Wave Functions". In: *The Journal of Chemical Physics* 90.9 (May 1989), pp. 4916–4926. DOI: 10.1063/1.456588.

[29] Susi Lehtola and Hannes Jónsson. "Pipek–Mezey Orbital Localization Using Various Partial Charge Estimates". In: *Journal of Chemical Theory and Computation* 10.2 (Feb. 2014), pp. 642–649. DOI: 10.1021/ct401016x.

[30] Qiming Sun et al. "PySCF: The Python-based Simulations of Chemistry Framework". In: *Wiley Interdiscip. Rev. Comput. Mol. Sci.* 8.1 (2018), e1340. DOI: 10.1002/wcms.1340.

[31] Christian Brouder et al. "Exponential Localization of Wannier Functions in Insulators". In: *Physical Review Letters* 98.4 (Jan. 2007), p. 046402. DOI: 10.1103/PhysRevLett.98.046402.

[32] Jeremy Oliver Richardson. "Communication: Machine Learning of Double-Valued Nonadiabatic Coupling Vectors around Conical Intersections". In: *The Journal of Chemical Physics* (Dec. 2022). DOI: 10.1063/5.0133191.

[33] Michael Scherbela, Leon Gerard, and Philipp Grohs. "Variational Monte Carlo on a Budget — Fine-tuning Pre-Trained Neural Wavefunctions". In: *Thirty-Seventh Conference on Neural Information Processing Systems*. Nov. 2023.

[34] Leon Gerard et al. *Transferable Neural Wavefunctions for Solids*. May 2024. DOI: 10.48550/arXiv.2405.07599. eprint: 2405.07599.

[35] Nicholas Metropolis et al. "Equation of State Calculations by Fast Computing Machines". In: *The Journal of Chemical Physics* 21.6 (June 1953), pp. 1087–1092. DOI: 10.1063/1.1699114.

[36] W. K. Hastings. "Monte Carlo Sampling Methods Using Markov Chains and Their Applications". In: *Biometrika* 57.1 (1970), pp. 97–109. DOI: 10.2307/2334940. JSTOR: 2334940.

[37] Luca Thiede, Chong Sun, and Alán Aspuru-Guzik. *Waveflow: Enforcing Boundary Conditions in Smooth Normalizing Flows with Application to Fermionic Wave Functions*. Apr. 2023. DOI: 10.48550/arXiv.2211.14839. eprint: 2211.14839.

[38] Alec Radford et al. *Improving Language Understanding by Generative Pre-Training*. Jan. 2021.

[39] Mohamed Hibat-Allah et al. "Recurrent Neural Network Wave Functions". In: *Physical Review Research* 2.2 (June 2020), p. 023358. DOI: 10.1103/PhysRevResearch.2.023358.

[40] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. Jan. 2017. DOI: 10.48550/arXiv.1412.6980. eprint: 1412.6980.

[41] Federico Becca and Sandro Sorella. *Quantum Monte Carlo Approaches for Correlated Systems*. Cambridge: Cambridge University Press, 2017. ISBN: 978-1-107-12993-1. DOI: 10.1017/9781316417041.

[42] Douglas Adams. *The Hitchhiker's Guide to the Galaxy*. London: Pan Books, 1979.

[43] James Martens. "New Insights and Perspectives on the Natural Gradient Method". In: *Journal of Machine Learning Research* 21.146 (2020), pp. 1–76.

[44] W. M. C. Foulkes et al. "Quantum Monte Carlo Simulations of Solids". In: *Reviews of Modern Physics* 73.1 (Jan. 2001), pp. 33–83. DOI: 10.1103/RevModPhys.73.33.

[45] Max A. Woodbury. *Inverting Modified Matrices*. Department of Statistics, Princeton University, 1950.

[46] Riccardo Rende et al. *A Simple Linear Algebra Identity to Optimize Large-Scale Neural Network Quantum States*. Oct. 2023. DOI: 10.48550/arXiv.2310.05715. eprint: 2310.05715.

[47] Gil Goldshlager, Nilin Abrahamsen, and Lin Lin. *A Kaczmarz-inspired Approach to Accelerate the Optimization of Neural Network Wavefunctions*. Jan. 2024. DOI: 10.48550/arXiv.2401.10190. eprint: 2401.10190.

[48] Dmitrii Kochkov and Bryan K. Clark. *Variational Optimization in the AI Era: Computational Graph States and Supervised Wave-function Optimization*. Nov. 2018. DOI: 10.48550/arXiv.1811.12423.

[49] James Spencer et al. "Better, Faster Fermionic Neural Networks". In: *Third Workshop on Machine Learning and the Physical Sciences (NeurIPS 2020)*. Dec. 2020.

[50] M. T. Entwistle et al. "Electronic Excited States in Deep Variational Monte Carlo". In: *Nature Communications* 14.1 (Jan. 2023), p. 274. DOI: 10.1038/s41467-022-35534-5.

[51] Oliver Unke et al. "SE(3)-Equivariant Prediction of Molecular Wavefunctions and Electronic Densities". In: *Advances in Neural Information Processing Systems*. Vol. 34. Curran Associates, Inc., 2021, pp. 14434–14447.

[52] C. Lin, F.-H. Zong, and D. M. Ceperley. "Twist-Averaged Boundary Conditions in Continuum Quantum Monte Carlo". In: *Physical Review E* 64.1 (June 2001), p. 016702. DOI: 10.1103/PhysRevE.64.016702. eprint: cond-mat/0101339.

[53] Wan Tong Lou et al. *Neural Wave Functions for Superfluids*. May 2023. DOI: 10.48550/arXiv.2305.06989. eprint: 2305.06989.

[54] Gino Cassella et al. "Discovering Quantum Phase Transitions with Fermionic Neural Networks". In: *Physical Review Letters* 130.3 (Jan. 2023), p. 036401. DOI: 10.1103/PhysRevLett.130.036401.

[55] Xiang Li, Zhe Li, and Ji Chen. "Ab Initio Calculation of Real Solids via Neural Network Ansatz". In: *Nature Communications* 13.1 (Dec. 2022), pp. 1–9. DOI: 10.1038/s41467-022-35627-1.

[56] Simone Chiesa et al. "Finite-Size Error in Many-Body Simulations with Long-Range Interactions". In: *Physical Review Letters* 97.7 (Aug. 2006), p. 076404. DOI: 10.1103/PhysRevLett.97.076404.

Appendix: Publications

Gold-standard solutions to the Schrödinger equation using deep learning: How much physics do we need?

Leon Gerard[†,a,*], Michael Scherbela[†,*], Philipp Marquetand[†,§], and Philipp Grohs[†,‡,¶]

[†]Research Network Data Science @ Uni Vienna, Kolingasse 14-16, A-1090 Vienna, Austria
[‡]Faculty of Mathematics, University of Vienna, Oskar-Morgenstern-Platz 1, A-1090 Vienna, Austria
[§]Institute of Theoretical Chemistry, Faculty of Chemistry, University of Vienna, Währinger Straße 17, 1090 Vienna, Austria
[¶]Johann Radon Institute for Computational and Applied Mathematics, Austrian Academy of Sciences, Altenbergerstrasse 69, 4040 Linz, Austria
[a]leon.gerard@univie.ac.at
[*]These authors contributed equally

Abstract

Finding accurate solutions to the Schrödinger equation is the key unsolved challenge of computational chemistry. Given its importance for the development of new chemical compounds, decades of research have been dedicated to this problem, but due to the large dimensionality even the best available methods do not yet reach the desired accuracy. Recently the combination of deep learning with Monte Carlo methods has emerged as a promising way to obtain highly accurate energies and moderate scaling of computational cost. In this paper we significantly contribute towards this goal by introducing a novel deep-learning architecture that achieves 40-70% lower energy error at 6x lower computational cost compared to previous approaches. Using our method we establish a new benchmark by calculating the most accurate variational ground state energies ever published for a number of different atoms and molecules. We systematically break down and measure our improvements, focusing in particular on the effect of increasing physical prior knowledge. We surprisingly find that increasing the prior knowledge given to the architecture can actually decrease accuracy.

1 Introduction

The challenge of the Schrödinger Equation Accurately predicting properties of molecules and materials is of utmost importance for many applications, including the development of new materials or pharmaceuticals. In principle, any property of any molecule can be calculated from its wavefunction, which is obtained by solving the Schrödinger equation. In practice, computing accurate wavefunctions and corresponding energies is computationally extremely difficult for two reasons: First, the wavefunction is a high-dimensional function, depending on all coordinates of all electrons, subjecting most methods to the curse of dimensionality. Second, the required level of accuracy is extremely high. While total energies of small molecules are typically hundreds of Hartrees, the chemically relevant energy differences are on the order of 1 milli-Hartree as depicted in Fig. 1. Decades of research have produced a plethora of methods, which all require a trade-off between accuracy and computational cost: On one end of the spectrum are approximate methods such as Hartree-Fock (HF) or Density Functional Theory (DFT), which was awarded the Nobel prize in 1998. These methods can treat thousands of particles but can often only crudely approximate chemical properties. On the other end of the spectrum are "gold-standard" methods such as FCI (Full Configuration Interaction) or CCSD(T) (Coupled Clusters Singles Doubles (Perturbative Triples))

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

which yield energies that often closely agree with experiments, but can only treat up to 100 particles. Despite all these efforts, even for small molecules there do currently not exist highly accurate energy calculations. A 2020 benchmark of state-of-the-art methods for the benzene molecule found a spread of 4 mHa across different methods [1] – as a matter of fact, our results show that the absolute energies calculated in [1] are off by at least 600 mHa, due to the small basis set used in their calculations. An important characteristic of a method is the type of approximation being made: Hartree-Fock or

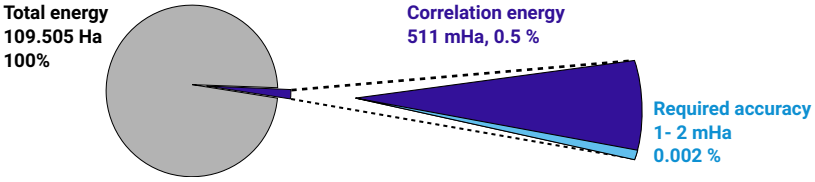

Figure 1: Conceptual visualization of the required accuracies on the example of an N_2 molecule: For chemical applications, total energies must achieve accuracies of $\sim$99.998%.

FCI are "variational", meaning that their predicted energies at least upper-bound the ground-truth energy. Since a lower energy is always guaranteed to be a closer approximation of the true energy, this makes assessment of these methods straight-forward. In contrast, CCSD(T) or DFT do not have any guarantees on the accuracy of their results. The approximations resulting from such methods, while working well for many common systems, often fail for chemically challenging situations such as breaking of chemical bonds [2, 3].

Deep-learning-based variational Monte Carlo Combining deep learning and Monte Carlo methods has recently emerged as a promising new approach for solving the Schrödinger equation [4, 5]. These methods offer high accuracy, moderate scaling of computational cost with system size and obey the variational principle. Within a few years deep-learning-based methods have managed to outperform conventional high-accuracy methods for many different molecules, potentially defining a new gold-standard for high-accuracy solutions. In the Born-Oppenheimer approximation a molecule, consisting of n_{nuc} nuclei and n_{el} electrons, is fully described by its Hamiltonian in atomic units

$$H = -\frac{1}{2}\sum_i \nabla^2_{\boldsymbol{r}_i} + \sum_{i>j}\frac{1}{|\boldsymbol{r}_i - \boldsymbol{r}_j|} + \sum_{I>J}\frac{Z_I Z_J}{|\boldsymbol{R}_I - \boldsymbol{R}_J|} - \sum_{i,I}\frac{Z_I}{|\boldsymbol{r}_i - \boldsymbol{R}_I|}.$$

Here $\boldsymbol{R}_I$, Z_I, $I \in \{1,\dots,n_{\text{nuc}}\}$ denote the coordinates and charges of the nuclei, $\boldsymbol{r} = (\boldsymbol{r}_1,\dots,\boldsymbol{r}_{n_\uparrow},\dots,\boldsymbol{r}_{n_{\text{el}}}) \in \mathbb{R}^{3\times n_{\text{el}}}$ denotes the set of n_{el} Cartesian electron coordinates differentiated between $n_\uparrow$ spin-up and $n_\downarrow$ spin-down electrons. We define the inter-particle vectors $\boldsymbol{r}_{ij} = \boldsymbol{r}_i - \boldsymbol{r}_j$ and $\boldsymbol{\rho}_{iJ} = \boldsymbol{r}_i - \boldsymbol{R}_J$. All properties of the molecule depend on the wavefunction $\psi(\boldsymbol{r})$, which must fulfill the antisymmetry constraint: $\psi(\mathcal{P}\boldsymbol{r}) = -\psi(\boldsymbol{r})$ for any permutation $\mathcal{P}$ of two electrons with the same spin [6]. The wavefunction ψ can be found as the solution to the Schrödinger equation $H\psi = E_0\psi$ with the ground-state energy and smallest eigenvalue E_0. By the Rayleigh-Ritz principle [7], the ground-state energy and the corresponding wavefunction can be found through minimization of the loss

$$\mathcal{L}(\psi_\theta) = \mathbb{E}_{\boldsymbol{r}\sim\psi^2_\theta(\boldsymbol{r})}\left[\frac{H\psi_\theta(\boldsymbol{r})}{\psi_\theta(\boldsymbol{r})}\right] \geq E_0 \tag{1}$$

for a trial wavefunction ψ_θ, parameterized by parameters θ. The trial function ψ_θ is represented by a neural network and typically has the form

$$\psi_\theta(\boldsymbol{r}) = \sum_{d=1}^{n_{\text{det}}} \det\left[\Lambda^d_{ki}(\boldsymbol{r})\Omega^{d\alpha_i}_k(\boldsymbol{r}_i)\right]_{k,i=1,\dots,n_{\text{el}}} \tag{2}$$

with $\Lambda^d_{ki} : \mathbb{R}^{3\times n_{\text{el}}} \to \mathbb{R}$, $\Omega^d_k : \mathbb{R}^3 \to \mathbb{R}$, $\alpha_i \in \{\uparrow,\downarrow\}$, $i \in \{1,\dots,n_{\text{el}}\}$, $k \in \{1,\dots,n_{\text{el}}\}$. Each determinant is taken over a $n_{\text{el}} \times n_{\text{el}}$ matrix, with row-indices k running over orbitals and column-indices i running over electrons. The determinant enforces antisymmetry, Ω^d_k are envelope functions enforcing the boundary condition $\lim_{|\boldsymbol{r}|\to\infty}\psi_\theta(\boldsymbol{r}) = 0$, and Λ^d_{ki} are neural networks. The local energy $\frac{H\psi}{\psi}$ can be evaluated using automatic differentiation and the loss can be minimized by gradient based methods. The computation of the expectation value in eq. 1 over the high-dimensional space $\mathbb{R}^{3\times n_{\text{el}}}$ is done using Monte Carlo integration by sampling electron coordinates $\boldsymbol{r}$ distributed according to ψ^2_θ using the Metropolis-Hastings [8] algorithm. A thorough discussion of deep-learning-based variational Monte Carlo (DL-VMC) can be found in [9].

Related work Two major neural network architectures and their extensions have emerged throughout literature: PauliNet [9] and FermiNet [10]. PauliNet puts emphasis on maximizing physical prior knowledge, by focusing on the the envelope function. They use the output of CASSCF (Complete Active Space Self Consistent Field, a sophisticated conventional quantum-chemistry method) as Ω and use a relatively small ($\sim$ 100k weights) neural network for Λ. FermiNet on the other hand uses a simple exponential function as envelope Ω and uses a large ($\sim$ 700k weights) neural network for Λ. Both approaches have been applied with great success to many different systems and properties, such as energies of individual molecules [4, 10, 9], ionization energies [10], potential energy surfaces [11, 12], forces [11], excited states [13], model-systems for solids [14, 15] and actual solids [16]. Several approaches have been proposed to increase accuracy or decrease computational cost, most notably architecture simplifications [17], alternative antisymmetric layers [18], effective core potentials [19] and Diffusion Monte Carlo (DMC) [20, 21]. FermiNet commonly reaches lower (i.e. more accurate) energies than PauliNet[10], but PauliNet has been observed to converge faster [12]. It has been proposed [9] that combining the embedding of FermiNet and the physical prior knowledge of PauliNet could lead to a superior architecture.

Our contribution In this work we present the counter-intuitive observation that the opposite approach might be more fruitful. By combining a PauliNet-like neural network embedding with the envelopes of FermiNet and adding several improvements to the embedding, input features, and initialization of parameter (Sec. 2), we obtain the currently best neural network architecture for the numerical solution of the electronic Schrödinger equation. Combining our new architecture with VMC we establish a new benchmark by calculating the most accurate variational ground state energies ever published for a number of different atoms and molecules - both when comparing to deep-learning-based methods, as well as when comparing to classical methods (Sec. 3). Across systems we reduce energy errors by 40-100% and achieve these results with 3-4x fewer optimization epochs compared to FermiNet. In Sec. 4 we systematically break down which changes cause these improvements. We hypothesize that including too much physical prior knowledge can actually hinder optimization and thus deteriorate accuracy – we provide ample experimental evidence in Sec. 5.

2 Improved approach

Similar to FermiNet, our architecture expresses Λ^d_{ki} as a linear combination of high-dimensional electron embeddings $\boldsymbol{h}^L_i$, and the envelopes $\Omega^{d\alpha_i}_k$ as a sum of exponential functions

$$\Lambda^d_{ki}(\boldsymbol{r}) = W^{d\alpha_i}_k \boldsymbol{h}^L_i \qquad \Omega^{d\alpha_i}_k(\boldsymbol{r}_i) = \sum_{I=1}^{n_{\text{nuc}}} \pi^{d\alpha_i}_{kI} \exp(-\omega^{d\alpha_i}_{kI} |\boldsymbol{\rho}_{iI}|), \tag{3}$$

where $W^{d\alpha_i}_k, \pi^{d\alpha_i}_{kI}, \omega^{d\alpha_i}_{kI}$ are trainable parameters and we enforce $\omega^{d\alpha_i}_{kI} \geq 0$. We compute these embeddings $\boldsymbol{h}^L_i$ by first transforming the inputs $\boldsymbol{R}_I, \boldsymbol{r}_i$ into feature vectors

$$\boldsymbol{h}^0_i = \left[|\boldsymbol{\rho}_{iI}|, \tilde{\boldsymbol{\rho}}_{iI} \right]_{I \in \{1,\dots,n_{\text{nuc}}\}} \qquad \boldsymbol{v}^0_{iI} = \left[|\boldsymbol{\rho}_{iI}|, \tilde{\boldsymbol{\rho}}_{iI} \right] \qquad \boldsymbol{g}^0_{ij} = |\boldsymbol{r}_{ij}|$$

where $[\cdot]$ denotes the concatenation operation and then applying L iterations of an embedding network (Fig. 2a). The local difference vectors $\tilde{\boldsymbol{\rho}}_{iI}$ are obtained by applying rotation matrices onto $\boldsymbol{\rho}_{iI}$ as described in Sec. 2.2.

2.1 Convolutional layers in embedding

Our embedding network uses four residual neural network streams (Fig. 2b): A primary one-electron stream that embeds a single electron, and three auxiliary streams modelling the two-particle-interactions (electrons with same spins, electrons with different spin, and electron-ion).

$$\boldsymbol{h}^{l+1}_i = \boldsymbol{A}^l_{\text{one}}(\boldsymbol{f}^l_i) + \boldsymbol{h}^l_i \qquad \boldsymbol{g}^{l+1}_{ij} = \boldsymbol{A}^l_{\sigma_{ij}}(\boldsymbol{g}^l_{ij}) + \boldsymbol{g}^l_{ij} \qquad \boldsymbol{v}^{l+1}_{iI} = \boldsymbol{A}^l_{\text{nuc}}(\boldsymbol{v}^l_{iI}) + \boldsymbol{v}^l_{iI} \tag{4}$$

Here l denotes the embedding iteration, $\boldsymbol{A}^l$ denote fully connected neural networks, and $\boldsymbol{g}^l_{ij}$, $\boldsymbol{v}^l_{iI}$ denote electron-electron- and electron-nucleus-embeddings. We use σ_{ij} = 'same' for same-spin

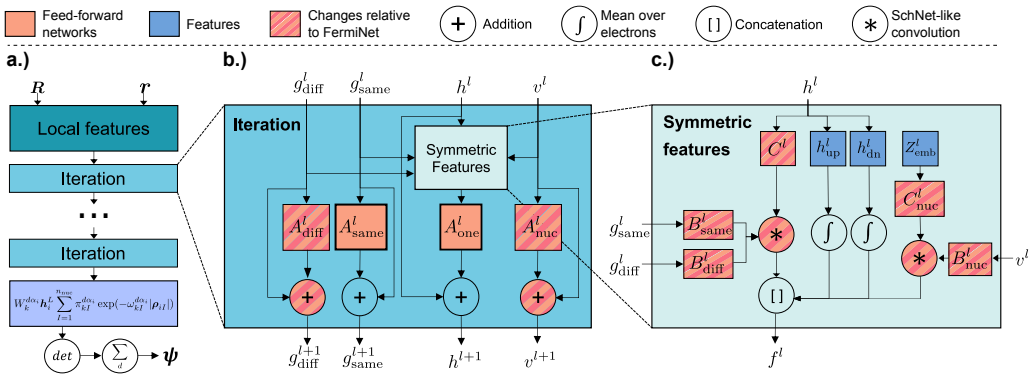

Figure 2: Our architecture: a) High-level overview b) One single embedding iteration c) Sub-block of assembling symmetric features

pairs of electrons and $\sigma_{ij} =$ 'diff' for pairs of electrons with different spin. Similar to FermiNet, in each iteration we assemble the input $\boldsymbol{f}_i^l$ to the primary stream from the auxiliary streams (Fig. 2c):

$$\boldsymbol{f}_i^l = \left[\boldsymbol{h}_i^l, \quad \frac{1}{n_\uparrow}\sum_{j=1}^{n_\uparrow}\boldsymbol{h}_j^l, \quad \frac{1}{n_\downarrow}\sum_{j=1+n_\uparrow}^{n_{el}}\boldsymbol{h}_j^l, \quad \boldsymbol{s}_i^{l,\text{el}}, \quad \boldsymbol{s}_i^{l,\text{nuc}}\right]. \tag{5}$$

Inspired by the success of SchNet [22] and the efficiency of the PauliNet embedding, we use the sum of element-wise multiplication ($\odot$), effectively forming a convolution, to aggregate the auxiliary two-particle streams:

$$\boldsymbol{s}_i^{l,\text{el}} = \sum_{j=1}^{n_{el}} \boldsymbol{B}_{\sigma_{ij}}^l\left(\boldsymbol{g}_{ij}^l\right) \odot \boldsymbol{C}_{\sigma_{ij}}^l\left(\boldsymbol{h}_j^l\right) \qquad \boldsymbol{s}_i^{l,\text{nuc}} = \sum_{I=1}^{n_{ion}} \boldsymbol{B}_{\text{nuc}}^l\left(\boldsymbol{v}_{iI}^l\right) \odot \boldsymbol{C}_{\text{nuc}}^l\left(Z_I^{\text{emb}}\right) \tag{6}$$

Eq. 5 and 6 form the core of the architecture and are the key difference between FermiNet, PauliNet and our architecture. The PauliNet architecture emphasizes two-particle interactions and essentially only uses convolutions as input features: $\boldsymbol{f}_i^l = [\boldsymbol{s}_i^{l,\text{el}}, \boldsymbol{s}_i^{l,\text{nuc}}]$. In addition to not using the $\boldsymbol{h}_i$ as input features, PauliNet also limits its effective depth by making the convolutional kernels $\boldsymbol{B}$ functions of the electron-electron distances $|\boldsymbol{r}_{ij}|$ instead of the high-dimensional embedded representations $\boldsymbol{g}_{ij}$. The FermiNet architecture on the other hand emphasizes the one-electron stream and only uses sums over $\boldsymbol{g}_{ij}$ as input features, essentially corresponding to $\boldsymbol{B}^l = \text{Id}$, $\boldsymbol{C}(\cdot) = \boldsymbol{1}$. Furthermore FermiNet does not contain an explicit stream for electron-nucleus interactions. Since our architecture adequately models both the one-electron-embedding as well as the two-particle-interactions, we expect our architecture to be more expressive than either predecessor, as demonstrated in Sec. 4.

2.2 Local, invariant input features

The first stage of any VMC wavefunction model is typically the computation of suitable input features from the raw electron coordinates $\boldsymbol{r}$ and nuclear coordinates $\{\boldsymbol{R}_I\}$. While the subsequent embedding stage could in principle take the raw coordinates, appropriate features allow to explicitly enforce symmetries and improve the model's transferability and accuracy. Input features should have three properties: First, they should be sufficiently expressive to encode any physical wavefunction. Second, the features should be invariant under geometric transformations. Third, the features should primarily depend on the local environment of a particle, i.e. similar local geometries should generate similar local features, mostly independent of changes to the geometry far from the particle in question. Published architectures have so far not been able to address all three points: PauliNet [10] uses only distances as input features, making them invariant and local, but not sufficiently expressive, as demonstrated by [12]. FermiNet [10] uses raw distances and differences, making the inputs expressive and local, but not invariant under rotations. PESNet [12] proposes a global coordinate system along the principle axes of a molecule, making the inputs invariant and sufficiently expressive, but not local.

We propose using local coordinate systems centered on every nucleus and evaluating the electron-nuclei differences in these local coordinate systems. Effectively this amounts to applying a rotation

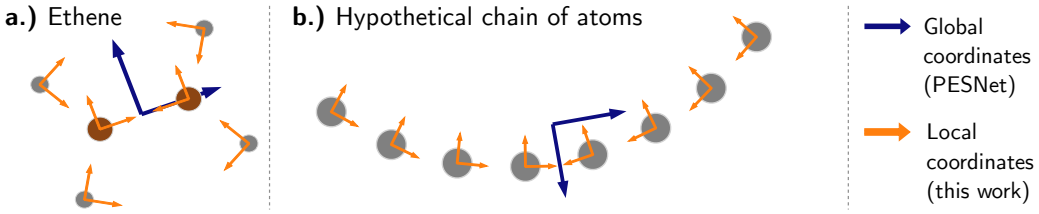

Figure 3: Visualization of resulting coordinate systems for 2 example molecules: a) Ethene b) A hypothetical bent chain of atoms.

matrix U_J to the raw electron-nucleus differences: $\tilde{\boldsymbol{\rho}}_{iJ} = U_J \boldsymbol{\rho}_{iJ}$. As opposed to [12] where U is constant for all nuclei, in our approach U_J can be different for every nucleus. These electron-nucleus differences $\tilde{\boldsymbol{\rho}}_{iJ}$ are invariant under rotations, contain all the information contained in raw Cartesian coordinates and depend primarily on the local environment of an atom. To compute the 3x3 rotation matrices U_J, we first run a single Hartree-Fock calculation using a minimal basis set to obtain a density matrix D. For each atom J we choose the 3x3 block of the density matrix which corresponds to the 3 p-orbitals of the atom J and compute its eigenvectors $U_J = \text{eig}(D_J)$. To make our coordinate system unique we sort the eigenvectors of D_J by their corresponding eigenvalues. If the eigenvalues are degenerate, we pick the rotation of the eigenvectors that maximizes the overlap with the coordinate-axes of the previous nucleus. Fig. 3 shows the resulting coordinate systems spanned by U_J. Note that the local coordinate system generally depicts more physically meaningful directions such as "along the chain". We find that these local coordinates slightly increase the accuracy for single geometries, but more importantly we expect the wavefunction to generalize better across different molecules or geometries. This should improve the accuracy of approaches that attempt to learn wavefunctions for multiple geometries at once [11, 12].

2.3 Initialization of orbital envelope weights

When considering a single atom, the entries of the wavefunction determinant have essentially the form

$$\Lambda_{ki}(\boldsymbol{r}) \exp\left(-\omega_k |\boldsymbol{\rho}_{iI}|\right).$$

In [10], the exponential envelope was purely motivated by the boundary condition that the orbitals must decay to zero, and initialization with $\omega_k = 1$ was proposed. However, when comparing this ansatz to analytical solutions, an interesting parallel can be found: Analytical solutions to the Schrödinger equation for atoms with a single electron – the only systems that have analytical solutions – are of the form

$$\widetilde{\Lambda}_k(\boldsymbol{\rho}_{iI}) \exp\left(-\frac{Z}{n_k}|\boldsymbol{\rho}_{iI}|\right),$$

where $\widetilde{\Lambda}_k(\boldsymbol{\rho}_{iI})$ is a product of a Laguerre polynomial and a spherical harmonic, and $n_k \in \mathbb{N}^+$ is known as the principal quantum number. This suggests $\omega_k \approx {}^Z\!/_{n_k}$, which we also find when analyzing the weights of a fully trained wavefunction. When initializing with $\omega_k = {}^Z\!/_{n_k}$ instead of $\omega_k = 1$, we observe faster convergence, lower final energies, and lower variance of the energies (Sec. 4). The effect is most notable for molecules containing nuclei with large Z, where ${}^Z\!/_{n_k} \gg 1$.

2.4 Improved hyperparameters

Beyond the improved neural network architecture we fine-tuned the hyperparameters to reduce the number of optimization steps required for convergence. Starting from the hyperparameters proposed by [10], we increased the norm constrain by 3x for the second-order optimizer KFAC [23, 24], decreased learning rate by 0.5x, and decreased learning rate decay time by 0.4x. We observe that these changes stabilize the optimization and enable usage of 50% fewer Monte Carlo walkers, which results in $\sim$2x faster optimization and reduced memory allocation. A complete set of hyperparameters can be found in appendix B.

3 Results of improved approach

We evaluated the accuracy of our approach by comparing our computed energies against the most accurate references available in the literature. Fig. 4 compares our energies against variational methods – for which lower energies are guaranteed to be more accurate – as well as non-variational high-accuracy methods. We find that across many different systems (small and large atoms, molecules at equilibrium geometry, molecules in transition states), our approach yields substantially lower – and thus more accurate – energies than previous variational results. Across all tested systems, we outperform almost all existing variational methods, both deep-learning-based methods as well as classical ones. When comparing to high-accuracy FermiNet VMC calculations, we not only reach substantially lower energies, but also do so using 3-4x fewer training steps, with each step being 40% faster (cf. appendix C). Comparing to a concurrently published Diffusion Monte Carlo approach, which used $\sim$10x more computational resources, we achieve similar or better accuracy for molecules like N_2 and cyclobutadiene and slightly lower accuracy for benzene. Non-variational methods (e.g. CCSD(T)) yield slightly lower energies than our calculations for some molecules, but since those methods do not provide upper bounds or uncertainty guarantees they do not provide a ground-truth. For many applications not only absolute energies are important, but energy differences between

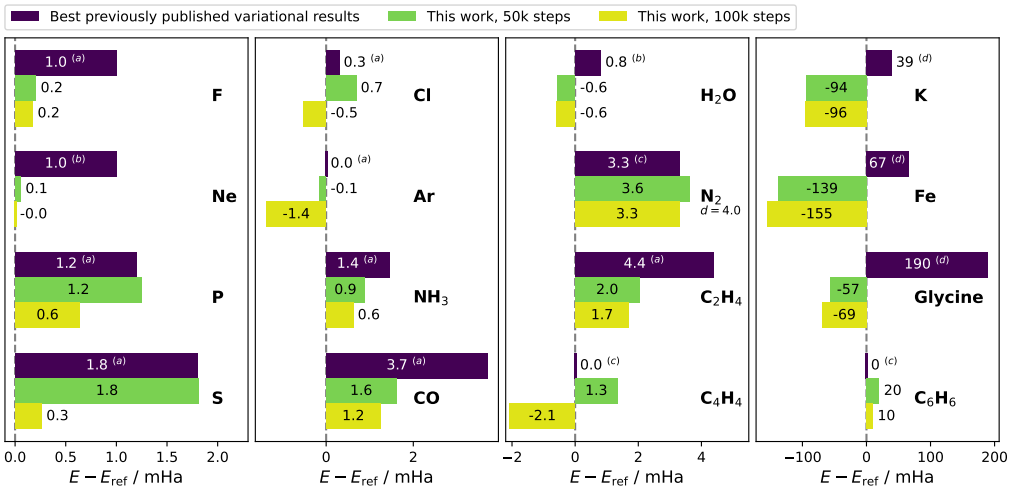

Figure 4: Energies relative to the previously known best estimate, (lower is better). Blue bars depict best published variational energies, footnotes mark the method: a: FermiNet VMC [10, 17], b: Conventional DMC [25, 26, 27], c: FermiNet DMC [21], d: MRCI-F12. A table of absolute energies and methods for E_{ref} can be found in appendix A. Note that E_{ref} is not necessary variational and thus may underestimate the true energy.

different molecules or geometries are of interest, for example to determine the energy required to break a chemical bond. A particularly difficult challenge is the dissociation of the N_2 molecule, i.e. the energy of an N_2 molecule at different bond lengths (inset Fig. 5). Even methods that are generally regarded as highly accurate, such as CCSD(T), predict energies that deviate far from experimental data at bond-lengths from 2.5 - 4 bohr. Fig. 5 depicts this deviation between experimental and computed energy for our method and the best available reference calculations. We find that our results are closer to the experimental absolute energies than all previous work, and are similar to concurrently published FermiNet-DMC results which require 5-10x more computational resources. When comparing relative energies, our approach outperforms all other deep-learning-based methods and CCSD(T), and is only beaten by the highly specialized r12-MR-ACPF method [28]. Similar to absolute energies, we also find that our relative energies converge substantially faster than for other deep-learning-based methods, with relative energies being almost fully converged after 50k epochs.

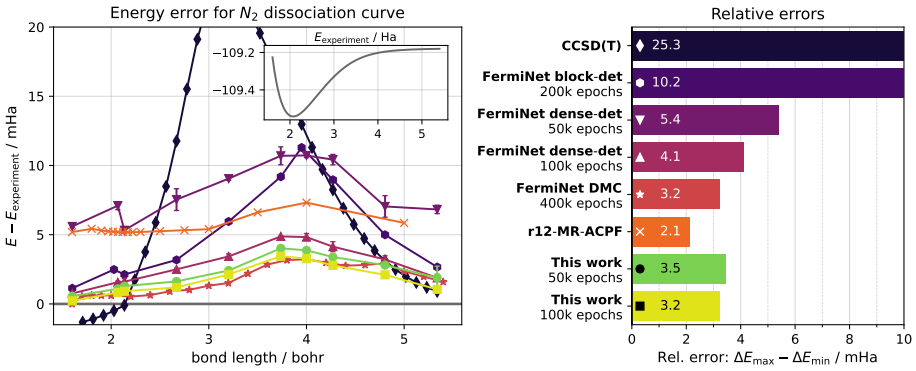

Figure 5: Comparison of energy error E - $E_{\text{experiment}}$ for the dissociation of the N_2 molecule across various methods. Errorbars corresponds to the standard deviation wrt. two different seeds. Errorbars for our work are too small to be visible ($\sim$ 0.1 mHa). Results for $E_{\text{experiment}}$ can be found in [29], FermiNet block-det & CCSD(T) in [10], FermiNet DMC in [21] and r12-MR-ACPF in [28].

4 Ablation study

To investigate which specific changes lead to the observed improvements in accuracy, we start from the improved FermiNet architecture proposed in [17] and incrementally add improvements in the following order: First, we use dense $n_{\text{el}} \times n_{\text{el}}$ determinants introduced by the FermiNet authors [10, 17] in their GitHub repository and described in [18] instead of block-diagonal determinants. This generalization increases computational cost and parameter count (cf. appendix C) but has been found to better describe the wavefunction's nodal surface and thus increase expressiveness of the ansatz. Second, we change hyperparameters as described in Sec. 2.4, which increases throughput by $\sim$ 2x. Third, we augment the electron embedding using our new SchNet-like neural network architecture described in Sec. 2.1. This leads to a moderate increase in parameter count and computational cost. Fourth, we switch to local, invariant input features as described in Sec. 2.2 and remove the electron-electron difference vectors $\boldsymbol{r}_{ij}$ as inputs. Lastly we switch to initializing $\omega_{kI}^{d} = {}^{Z}/_{n_k}$ as described in Sec. 2.3, resulting in our proposed final method. We note that the accuracy gains of these changes are not fully independent of each other and the relative attribution depends on the order in which they are applied: Earlier changes will generally generate larger energy improvements compared to later changes. At each step we compute total energies for three different molecules: ethene, N_2 at the challenging bond-length of 4.0 bohr, and the K-atom. Fig. 6 depicts the energy of our implementation of FermiNet, and the energy change caused by each subsequent improvement. Each experiment was repeated two times with different RNG seeds (appendix D), the errorbars depict the spread in energy. Overall we find that all our changes combined yield a $\sim$3-20x improvement in the energy error. For ethene, the dominant contribution (3.7 mHa) comes from improved hyperparameters, which lead to the results being mostly converged after 50k epochs vs. the original settings which require 200k epochs for convergence. Using a lower learning rate in combination with a larger gradient-norm-constraint ensures that more optimization steps are taken according to curvature estimated by KFAC and fewer steps are clipped by the gradient-norm-constraint. Architectural improvements (embedding and input features) lower the energy error by additional 1.4 mHa. Because our embedding is a strict generalization of both FermiNet and PauliNet, our ansatz is more expressive and can therefore reach lower energies than previous ansätze. For N_2 it has already been observed that a single dense determinant can outperform models with multiple block-diagonal determinants [18]. We find the corresponding result that 32 dense determinants substantially lower the energy relative to an ansatz with 32 block-diagonal determinants. Comparing N_2 to ethene, we observe larger contributions from our architectural improvements and smaller contributions from improved hyperparameters. For the K atom, the overall gains are largest, totalling 60mHa, with substantial accuracy gains from all improvements. Since K has a much larger nuclear charge (Z=19) than the constituents of ethene (Z=1,6) and N_2 (Z=7), also the physics-inspired initialization of the envelope parameters yields a substantial contribution. This improved initialization leads to a better initial guess for the wavefunction, which not only reduces the number of required optimization steps, but also leads to more accurate initial sampling.

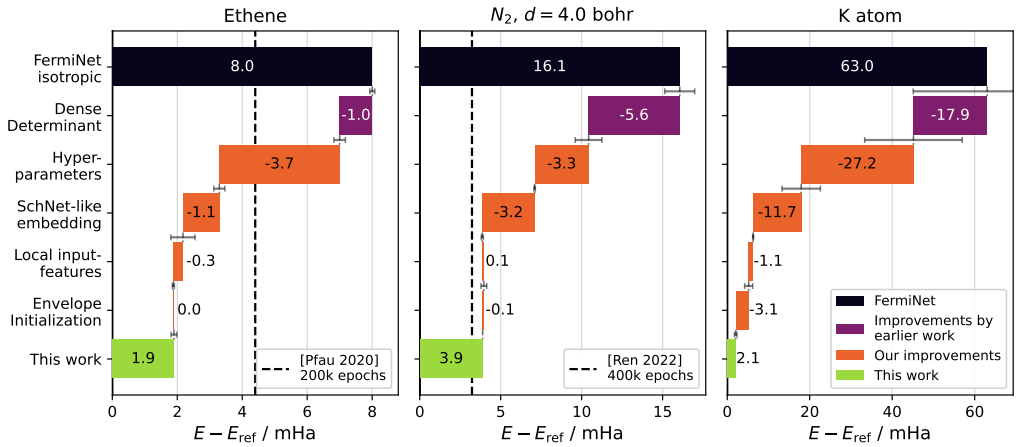

Figure 6: Breakdown of accuracy improvements for three different molecules, each trained for 50k epochs. The black dashed line depicts the best published variational result and as reference we use the best available estimates: CCSD(T) [10] for ethene, experimental data for N_2 [29], and our own VMC calculations after 100k epochs for the K atom.

5 Incorporating prior knowledge

To further understand the effect of incorporating prior knowledge into neural network architectures for physical problems as the electronic Schrödinger equation, we examined two distinct ways of increasing prior information in our model: First, by including a built-in approximate physical model, analogous to PauliNet. Second, by increasing the number of pre-training steps to more closely match a reference wavefunction before starting the optimization.

Explicitly include CASSCF PauliNet maximizes the physical prior information by computing the envelopes Ω with CASSCF, a sophisticated conventional method, and explicitly enforcing the Kato cusp conditions [30]. Starting from our proposed architecture, we modified our approach step-by-step until we arrived at a PauliNet-like architecture. Fig. 7a shows the energies of an NH_3 molecule trained for 50k epochs at each step.

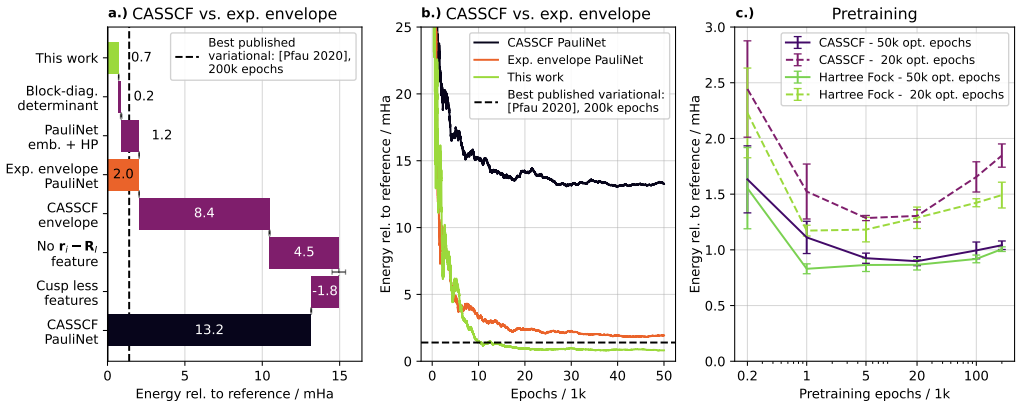

Figure 7: Effect of prior knowledge for NH_3: a) Change in accuracy after 50k optimization epochs when transitioning from our approach to a PauliNet-like architecture, by increasing prior knowledge. b) Energy error for 3 architectures over number of training epochs. c) Mean and standard deviation of the energy error as a function of pre-training steps. Pre-training used CASSCF or HF as a reference, and the energy was evaluated after 20k or 50k subsequent variational optimization epochs.

First, we switch from dense determinants to block-diagonal determinants as used by the original PauliNet, leading to small loss in accuracy. Second, we exchange our embedding for the PauliNet-like embedding using the hyperparameters proposed in [11], leading to a substantial loss in accuracy, presumably caused by a loss in expressiveness. Next, we replace the simple exponential envelopes by the more physically inspired CASSCF-envelopes, causing a large loss in accuracy. We then remove the vector $\boldsymbol{r}_i - \boldsymbol{R}_I$ as input feature (keeping only its absolute value) as done in the original PauliNet architecture [9]. This again deteriorates accuracy, presumably due to enforcing rotational invariance which is too restrictive of a symmetry class as pointed out by [12]. Lastly we replace the electron-electron distances $|\boldsymbol{r}_{ij}|$ (which are not smooth at $\boldsymbol{r}_{ij} = \boldsymbol{0}$ and thus lead to cusps) by smooth, cuspless radial basis functions as input feature and add an explicit term to enforce the electron-electron cusp condition. Since the CASSCF-envelopes are designed to fulfill the electron-ion cusp condition, this change leads to an enforced cusp condition, slightly improving the energy accuracy. The largest loss in accuracy throughout these changes is caused by introducing the CASSCF-envelopes, suggesting that they introduce a strong bias of the wavefunction that cannot be easily overcome during training. Fig. 7b shows that architectures using exponential envelopes converge to lower absolute energies compared to the CASSCF-based PauliNet and outperform PauliNet already after $\sim$5000 epochs.

Increase pre-training accuracy Before starting the unsupervised variational optimization of the wavefunction, we run a short supervised pre-training of the wavefunction to roughly match a given reference wavefunction. This is computationally inexpensive because it only requires evaluation of ψ_θ and a back-propagation step to update the neural network weights θ but not the second derivative of the Hamiltonian. If the reference method yields a decent approximation to the true wavefunction, this pre-training significantly speeds-up optimization and avoids unstable parameter regimes [10]. To incorporate more prior knowledge, one could either use a more sophisticated reference method (e.g. CASSCF instead of HF) or increase the number of pre-training steps. In Fig. 7c we pre-trained the wavefunction with a block diagonal determinant for the NH_3 molecule using a CASSCF and Hartree-Fock reference. We increased pre-training iteration steps and evaluated the energy after subsequent 20k and 50k variational optimization epochs, each run was repeated with five different seeds. Increasing the number of pre-training steps initially increases accuracy – since it provides a better starting point for the subsequent variational optimization – but when increasing pre-training beyond 20k steps, accuracy deteriorates for both methods. Surprisingly, we observe a drop in accuracy when using CASSCF as a reference method compared to the simpler Hartree-Fock method. This effect is even more pronounced when increasing the number of pre-training steps. It suggests that excessive pre-training introduces a bias that is hard to overcome during variational optimization, similarly to a built-in reference method.

6 Discussion and Limitations

Discussion We find that our approach yields substantially more accurate absolute energies than all previous work – both classical as well as deep-learning-based – and that we reach these accuracies 4-6x faster than the next best method (FermiNet). Especially for larger systems, such as 4th row atoms or the amino acid glycine, we outperform conventional "gold-standard" methods like MRCI-F12(Q) by $\sim$100 mHa. This corroborates the fact that deep-learning-based methods are emerging as a new gold-standard in computational chemistry and showcases the immense potential of machine-learning-based methods in the natural sciences. A concurrent work [21] was able to achieve similar accuracies by applying Diffusion Monte Carlo (DMC) on top of a FermiNet VMC calculation, highlighting the potential of deep-learning Monte Carlo methods. However, [21] required $\sim$10x more computational resources and their VMC results – already by themselves 8x more expensive then our calculations – are consistently inferior to our results. This showcases a promising route towards further improvements by using our substantially cheaper and more accurate VMC results as a starting point for a DMC calculation.

Regarding the question of how much physics to include in the model, we find varying results. For exact physical constraints, such as symmetries or the cusp conditions, inclusion in the model generally appears to be helpful. However for prior knowledge from existing approximate solutions (such as CASSCF) the situation is more subtle. On the one hand, soft physical guidance such as short supervised pre-training or physics-inspired weight initialization accelerates optimization. On the other hand, we show empirically that increasing physical prior knowledge, e.g. by incorporating

CASSCF or extensive supervised pre-training, does not necessarily increase accuracy, but can in fact introduce detrimental biases that are hard to overcome during wavefunction optimization.

Limitations and outlook Despite the proposed improvements and favorable scaling of the method, computation of energies for large molecules still takes days of GPU-time on current hardware. While the same holds true for conventional high-accuracy approaches, substantial speed-ups are still required to make DL-VMC more accessible for practitioners. Additionally, when increasing the nuclear charges, the wavefunction becomes increasingly localised, which leads to a reduction in average Monte Carlo stepsize and potentially correlated samples. We circumvent this effect for 4th row atoms by increasing the number of intermediate Monte Carlo steps, but further research into Monte Carlo sampling methods [31, 32] is required to fully address this issue. Despite our improvements for the accuracy of energy differences between different molecules or geometries, DL-VMC is still outperformed by other, computationally cheaper methods in some cases. Initial research into the regularity of the wavefunction across different molecules [11, 12] provides a promising route to improvements. We note in passing that thanks to the local coordinate input features, our architecture fulfills the required rotational invariance required for these approaches.

7 Code availability

The code alongside a detailed documentation is available as part of the DeepErwin code package on the Python Package Index (PyPI) and github (`https://github.com/mdsunivie/deeperwin`) under the MIT license.

8 Acknowledgements

We gratefully acknowledge financial support from the following grants: Austrian Science Fund FWF Project I 3403 (P.G.), WWTF-ICT19-041 (L.G.). The computational results have been achieved using the Vienna Scientific Cluster (VSC). The funders had no role in study design, data collection and analysis, decision to publish or preparation of the manuscript. Additionally, we thank Nicholas Gao for providing his results and data and Rafael Reisenhofer for providing valuable feedback to the manuscript.

References

[1] Janus J. Eriksen et al. "The Ground State Electronic Energy of Benzene". In: *The Journal of Physical Chemistry Letters* 11.20 (Oct. 2020). Publisher: American Chemical Society, pp. 8922–8929. DOI: 10.1021/acs.jpclett.0c02621. URL: https://doi.org/10.1021/acs.jpclett.0c02621 (visited on 04/21/2022).

[2] Narbe Mardirossian and Martin Head-Gordon. "Thirty years of density functional theory in computational chemistry: an overview and extensive assessment of 200 density functionals". In: *Molecular Physics* 115.19 (2017), pp. 2315–2372. DOI: 10.1080/00268976.2017.1333644. eprint: https://doi.org/10.1080/00268976.2017.1333644. URL: https://doi.org/10.1080/00268976.2017.1333644.

[3] Aron J. Cohen, Paula Mori-Sánchez, and Weitao Yang. "Challenges for Density Functional Theory". In: *Chemical Reviews* 112.1 (Jan. 2012). Publisher: American Chemical Society, pp. 289–320. ISSN: 0009-2665. DOI: 10.1021/cr200107z. URL: https://doi.org/10.1021/cr200107z.

[4] Jiequn Han, Linfeng Zhang, and Weinan E. "Solving many-electron Schrödinger equation using deep neural networks". In: *J. Comput. Phys.* 399 (Dec. 2019), p. 108929. ISSN: 0021-9991. DOI: 10.1016/j.jcp.2019.108929. URL: http://dx.doi.org/10.1016/j.jcp.2019.108929.

[5] Giuseppe Carleo and Matthias Troyer. "Solving the quantum many-body problem with artificial neural networks". In: *Science* 355.6325 (Feb. 2017). Publisher: American Association for the Advancement of Science, pp. 602–606. DOI: 10.1126/science.aag2302. URL: https://www.science.org/doi/10.1126/science.aag2302 (visited on 05/09/2022).

[6] Wolfgang Pauli. *Nobel lecture*. Dec. 1946.

[7] Walter Ritz. "Über eine neue Methode zur Lösung gewisser Variationsprobleme der mathematischen Physik." In: 1909.135 (1909), pp. 1–61. DOI: doi:10.1515/crll.1909.135.1. URL: https://doi.org/10.1515/crll.1909.135.1.

[8] W. K. Hastings. "Monte Carlo sampling methods using Markov chains and their applications". In: *Biometrika* 57.1 (Apr. 1970), 97–109. ISSN: 0006-3444. DOI: 10.1093/biomet/57.1.97. eprint: https://academic.oup.com/biomet/article-pdf/57/1/97/23940249/57-1-97.pdf. URL: https://doi.org/10.1093/biomet/57.1.97.

[9] Jan Hermann, Zeno Schätzle, and Frank Noé. "Deep-neural-network solution of the electronic Schrödinger equation". In: *Nat. Chem.* 12.10 (Oct. 2020), 891–897. ISSN: 1755-4349. DOI: 10.1038/s41557-020-0544-y. URL: https://doi.org/10.1038/s41557-020-0544-y.

[10] David Pfau et al. "Ab initio solution of the many-electron Schrödinger equation with deep neural networks". In: *Phys. Rev. Res.* 2 (3 Sept. 2020), p. 033429. DOI: 10.1103/PhysRevResearch.2.033429. URL: https://link.aps.org/doi/10.1103/PhysRevResearch.2.033429.

[11] Michael Scherbela et al. *Solving the electronic Schrödinger equation for multiple nuclear geometries with weight-sharing deep neural networks*. 2021. DOI: 10.48550/ARXIV.2105.08351. URL: https://arxiv.org/abs/2105.08351.

[12] Nicholas Gao and Stephan Günnemann. *Ab-Initio Potential Energy Surfaces by Pairing GNNs with Neural Wave Functions*. 2021. DOI: 10.48550/ARXIV.2110.05064. URL: https://arxiv.org/abs/2110.05064.

[13] Mike Entwistle et al. *Electronic excited states in deep variational Monte Carlo*. 2022. DOI: 10.48550/ARXIV.2203.09472. URL: https://arxiv.org/abs/2203.09472.

[14] Max Wilson et al. *Wave function Ansatz (but Periodic) Networks and the Homogeneous Electron Gas*. 2022. DOI: 10.48550/ARXIV.2202.04622. URL: https://arxiv.org/abs/2202.04622.

[15] G. Cassella et al. *Discovering Quantum Phase Transitions with Fermionic Neural Networks*. DOI: 10.48550/ARXIV.2202.05183. URL: https://arxiv.org/abs/2202.05183.

[16] Xiang Li, Zhe Li, and Ji Chen. "Ab initio calculation of real solids via neural network ansatz". In: *arXiv:2203.15472 [cond-mat, physics:physics]* (Mar. 2022). arXiv: 2203.15472. DOI: 10.48550/ARXIV.2203.15472. URL: http://arxiv.org/abs/2203.15472 (visited on 04/20/2022).

[17] James S. Spencer et al. *Better, Faster Fermionic Neural Networks*. 2020. DOI: 10.48550/ARXIV.2011.07125. URL: https://arxiv.org/abs/2011.07125.

[18] Jeffmin Lin, Gil Goldshlager, and Lin Lin. "Explicitly antisymmetrized neural network layers for variational Monte Carlo simulation". In: *arXiv:2112.03491 [physics]* (Dec. 2021). arXiv: 2112.03491. DOI: 10.48550/ARXIV.2112.03491. URL: http://arxiv.org/abs/2112.03491 (visited on 03/25/2022).

[19] Xiang Li et al. "Fermionic neural network with effective core potential". In: *Physical Review Research* 4.1 (Jan. 2022). Publisher: American Physical Society, p. 013021. DOI: 10.1103/PhysRevResearch.4.013021. URL: https://link.aps.org/doi/10.1103/PhysRevResearch.4.013021 (visited on 04/26/2022).

[20] Max Wilson et al. *Simulations of state-of-the-art fermionic neural network wave functions with diffusion Monte Carlo*. 2021. DOI: 10.48550/ARXIV.2103.12570. arXiv: 2103.12570 [physics.chem-ph].

[21] Weiluo Ren, Weizhong Fu, and Ji Chen. *Towards the ground state of molecules via diffusion Monte Carlo on neural networks*. 2022. DOI: 10.48550/ARXIV.2204.13903. URL: https://arxiv.org/abs/2204.13903.

[22] Kristof Schütt et al. "SchNet: A continuous-filter convolutional neural network for modeling quantum interactions". In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017. URL: https://proceedings.neurips.cc/paper/2017/file/303ed4c69846ab36c2904d3ba8573050-Paper.pdf.

[23] James Martens and Roger Grosse. "Optimizing neural networks with kronecker-factored approximate curvature". In: *International conference on machine learning*. PMLR. 2015, pp. 2408–2417.

[24] Aleksandar Botev and James Martens. *KFAC-JAX*. Version 0.0.1. 2022. URL: http://github.com/deepmind/kfac-jax.

[25] P. Seth, P. López Ríos, and R. J. Needs. "Quantum Monte Carlo study of the first-row atoms and ions". In: *The Journal of Chemical Physics* 134.8 (2011), p. 084105. DOI: 10.1063/1.3554625. eprint: https://doi.org/10.1063/1.3554625. URL: https://doi.org/10.1063/1.3554625.

[26] Norbert Nemec, Michael D. Towler, and R. J. Needs. "Benchmark all-electron ab initio quantum Monte Carlo calculations for small molecules". In: *The Journal of Chemical Physics* 132.3 (Jan. 2010). Publisher: American Institute of Physics, p. 034111. ISSN: 0021-9606. DOI: 10.1063/1.3288054. URL: http://aipscitation.org/doi/full/10.1063/1.3288054 (visited on 04/12/2022).

[27] Bryan K. Clark et al. "Computing the energy of a water molecule using multideterminants: A simple, efficient algorithm". In: *The Journal of Chemical Physics* 135.24 (Dec. 2011). Publisher: American Institute of Physics, p. 244105. ISSN: 0021-9606. DOI: 10.1063/1.3665391. URL: http://aipscitation.org/doi/full/10.1063/1.3665391 (visited on 05/31/2022).

[28] Robert J. Gdanitz. "Accurately solving the electronic Schrödinger equation of atoms and molecules using explicitly correlated (r12-)MR-CI: the ground state potential energy curve of N2". In: *Chemical Physics Letters* 283.5 (1998), pp. 253–261. ISSN: 0009-2614. DOI: https://doi.org/10.1016/S0009-2614(97)01392-4. URL: https://www.sciencedirect.com/science/article/pii/S0009261497013924.

[29] Robert J. Le Roy, Yiye Huang, and Calvin Jary. "An accurate analytic potential function for ground-state N2 from a direct-potential-fit analysis of spectroscopic data". In: *The Journal of Chemical Physics* 125.16 (2006), p. 164310. DOI: 10.1063/1.2354502. eprint: https://doi.org/10.1063/1.2354502. URL: https://doi.org/10.1063/1.2354502.

[30] Tosio Kato. "On the eigenfunctions of many-particle systems in quantum mechanics". In: *Commun. Pure Appl. Math.* 10.2 (1957), 151–177. DOI: 10.1002/cpa.3160100201. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/cpa.3160100201. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/cpa.3160100201.

[31] C. J. Umrigar, M. P. Nightingale, and K. J. Runge. "A diffusion Monte Carlo algorithm with very small time-step errors". In: *The Journal of Chemical Physics* 99.4 (1993), pp. 2865–2890. DOI: 10.1063/1.465195. eprint: https://doi.org/10.1063/1.465195. URL: https://doi.org/10.1063/1.465195.

[32] C. J. Umrigar. “Accelerated Metropolis method”. In: *Phys. Rev. Lett.* 71 (3 July 1993), pp. 408–411. DOI: 10.1103/PhysRevLett.71.408. URL: https://link.aps.org/doi/10.1103/PhysRevLett.71.408.

nature computational science

ARTICLES

https://doi.org/10.1038/s43588-022-00228-x

Check for updates

Solving the electronic Schrödinger equation for multiple nuclear geometries with weight-sharing deep neural networks

Michael Scherbela[1,5], Rafael Reisenhofer[1,2,5] ✉, Leon Gerard[1,5], Philipp Marquetand[1,3] and Philipp Grohs[1,2,4]

The Schrödinger equation describes the quantum-mechanical behaviour of particles, making it the most fundamental equation in chemistry. A solution for a given molecule allows computation of any of its properties. Finding accurate solutions for many different molecules and geometries is thus crucial to the discovery of new materials such as drugs or catalysts. Despite its importance, the Schrödinger equation is notoriously difficult to solve even for single molecules, as established methods scale exponentially with the number of particles. Combining Monte Carlo techniques with unsupervised optimization of neural networks was recently discovered as a promising approach to overcome this curse of dimensionality, but the corresponding methods do not exploit synergies that arise when considering multiple geometries. Here we show that sharing the vast majority of weights across neural network models for different geometries substantially accelerates optimization. Furthermore, weight-sharing yields pretrained models that require only a small number of additional optimization steps to obtain high-accuracy solutions for new geometries.

Using a deep neural network-based ansatz for variational Monte Carlo (VMC) has recently emerged as a novel approach for highly accurate ab initio solutions to the multi-electron Schrödinger equation[1–5]. It has been observed that such methods can exceed the accuracy of gold-standard quantum-chemistry methods such as coupled clusters with single-, double- and perturbative triplet-excitations (CCSD(T))[6], with the computational cost per step scaling only with complexity $\mathcal{O}(N^4)$ for the number of electrons N (ref. [4]). This suggests a drastic improvement from classical quantum-chemistry methods such as CCSD(T) or configuration interaction singles, doubles, triples, quadruples (CISDTQ), which scale with $\mathcal{O}(N^7)$ and $\mathcal{O}(N^{10})$, respectively. However, due to the large number of free parameters and the need for Monte Carlo integration, the constant runtime prefactor for neural network-based methods is typically much larger than for classical approaches such that even systems of modest size still require days or weeks for computation when using highly optimized implementations on state-of-the-art hardware[7]. This often renders deep neural network (DNN)-based ansatz methods unfeasible in practice, in particular when highly accurate results for a large number of molecular geometries are required.

Among such tasks are computational structure search, determination of chemical transition states and the generation of training datasets for supervised machine learning algorithms in quantum chemistry. The last two methods are applied with great success to interpolate results of established quantum-chemistry methods such as energies and forces[8–10], properties of excited states[11], underlying objects such as orbital energies[12] or the exchange energy[13]. Given sufficient training data, these interpolations already achieve chemical accuracy relative to the training method (for example, density functional theory)[14], highlighting the need for increasingly accurate ab initio methods that can be used to generate reference training data.

The goal of making DNN-based VMC applicable for the generation of such high-quality datasets for previously untractable molecules is a key motivation for this work. The apparent success of supervised learning in quantum chemistry suggests a high degree of regularity of the aforementioned properties and the wavefunction itself within the space of molecular geometries.

Here we already aim to exploit potential regularities of the wavefunction within the space of molecular geometries during VMC optimization by applying a simple technique called weight-sharing. Throughout optimizing instances of the same neural network-based wavefunction model for different molecular geometries, we enforce that each instance has the exact same neural network weights for large parts of the model. In particular, this means that on the parts of the model in which weight-sharing is applied, each instance computes precisely the same function. We note that this idea is reminiscent of (and inspired by) the machine learning technique deep transfer learning, where parts of a pretrained model are reused for different similar tasks to breakthrough results, for example, in natural language processing[15] or computer vision[16].

Weight-sharing can be viewed as a regularization technique that requires large parts of the optimized model to work equally well for a potentially wide variety of different nuclear (or even molecular) geometries. Under the assumption that the wavefunctions are sufficiently uniform across geometries, weight-sharing should therefore have a stabilizing effect on the optimization process and yield wavefunctions that also generalize well to new molecular geometries when used as an initial guess before optimization. For a shared weight, each gradient descent update during optimization for a specific geometry is applied to the complete set of considered

[1]Research Network Data Science at University of Vienna, Vienna, Austria. [2]Faculty of Mathematics, University of Vienna, Vienna, Austria. [3]Faculty of Chemistry, University of Vienna, Vienna, Austria. [4]Johann Radon Institute for Computational and Applied Mathematics, Austrian Academy of Sciences, Linz, Austria. [5]These authors contributed equally: Michael Scherbela, Rafael Reisenhofer, Leon Gerard. ✉e-mail: rafael.reisenhofer@uni-bremen.de

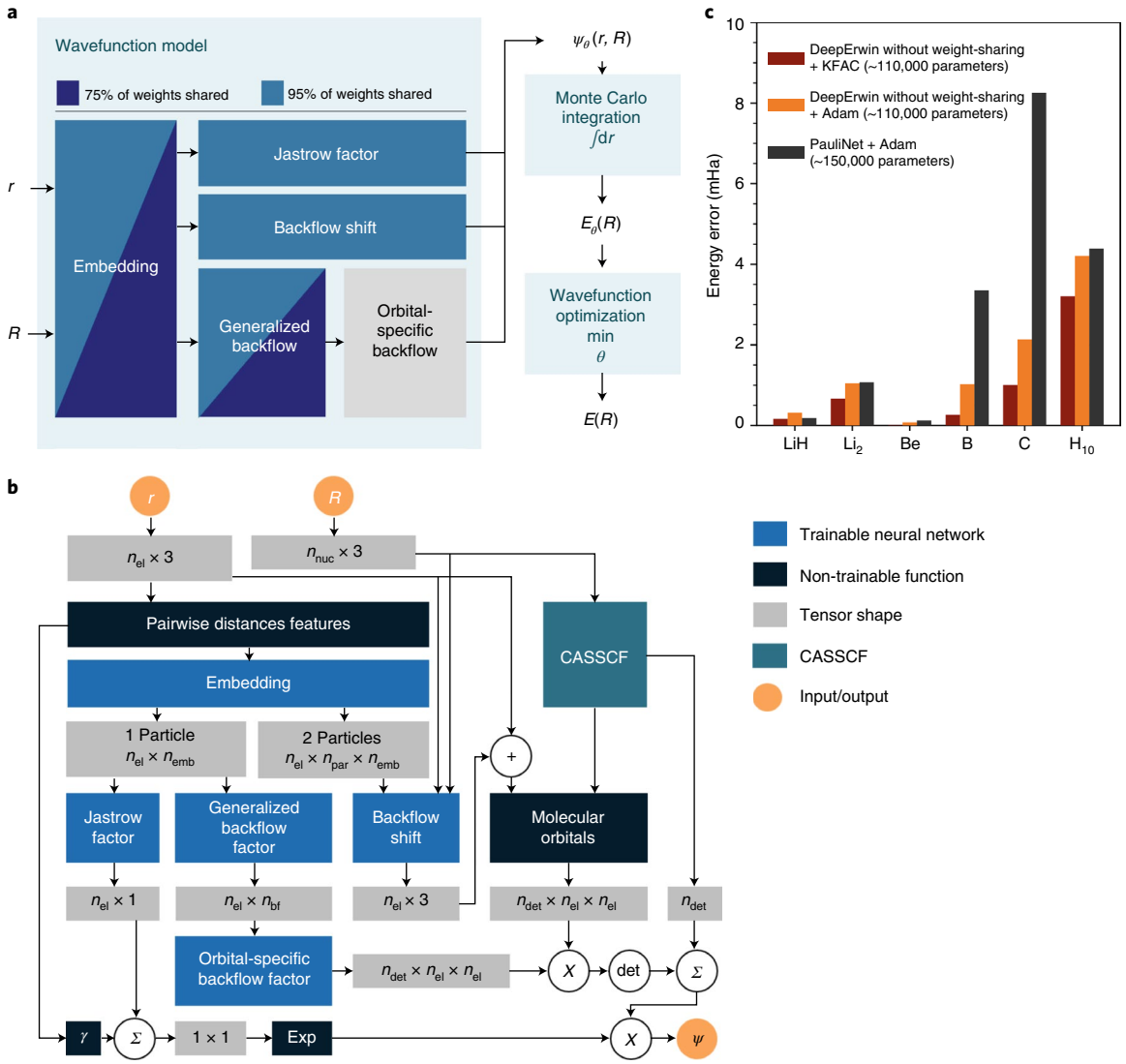

Fig. 1 | Overview of the DeepErwin framework. a, Overview of the neural network parts of the wavefunction model implemented in the DeepErwin framework, including a top-level visualization of the two weight-sharing set-ups considered in our experiments. **b**, Overview of the wavefunction model implemented in DeepErwin. Spin dependence has been omitted for clarity, but in practice there are two parallel streams for spin-up and -down, respectively. n_{el}, number of electrons; n_{nuc}, number of nuclei; n_{par}, number of particles (electrons + nuclei); n_{det}, number of determinants; n_{emb}, embedding dimension; n_{bf}, backflow feature dimension. **c**, Energy of optimized wavefunctions for several small atoms and molecules relative to reference calculations[4] for DeepErwin with first-order optimizer Adam[21], second-order optimizer K-FAC, and PauliNet[2]. DeepErwin baseline solutions without weight-sharing achieve a higher accuracy than PauliNet across the tested systems despite a smaller number of parameters.

geometries. Weight-sharing therefore has the potential to substantially accelerate the optimization process.

Our main numerical results highlight the benefits of weight-sharing compared with independent optimization and the applicability of pretrained shared weights for new calculations. In particular, we show that by applying these techniques in combination with second-order optimization, it is possible to consistently reach the energies of multi-reference configuration interaction (MRCI)-F12 reference calculations—up to chemical accuracy—for molecules up to the size of ethene after only $\mathcal{O}(10^2)$ optimization epochs per geometry. Note that wavefunctions are typically being optimized from $\mathcal{O}(10^4)$ to $\mathcal{O}(10^5)$ epochs for the most recently proposed DNN-based VMC methods. To further demonstrate the applicability of the proposed framework in practice, we calculate the transition path for H_4^+ between two symmetry-equivalent minima, wavefunctions for a set of differently twisted and stretched ethene configurations, as well as the potential energy surface (PES)—including forces—of a H_{10} chain on a two-dimensional grid of nuclear coordinates. Our approach yields these forces in addition to energies at a low incremental cost, as opposed to other high-accuracy methods such as domain-based local-pair natural-orbital-coupled cluster[17], for which forces typically have a substantial incremental computational cost.

Results

To investigate weight-sharing for neural network-based models in VMC, we consider a framework called DeepErwin, in which the trial wavefunction is modelled similarly to the recently proposed PauliNet[2], with modifications leading to an overall smaller network that yields higher accuracies. The basic idea behind this model is to enhance a Slater determinant ansatz with deep neural networks, where initial orbitals are obtained from a complete active space self-consistent field (CASSCF) calculation with a small basis set and a small active space. The resulting wavefunction is then modified by applying a backflow transformation to the orbitals as well

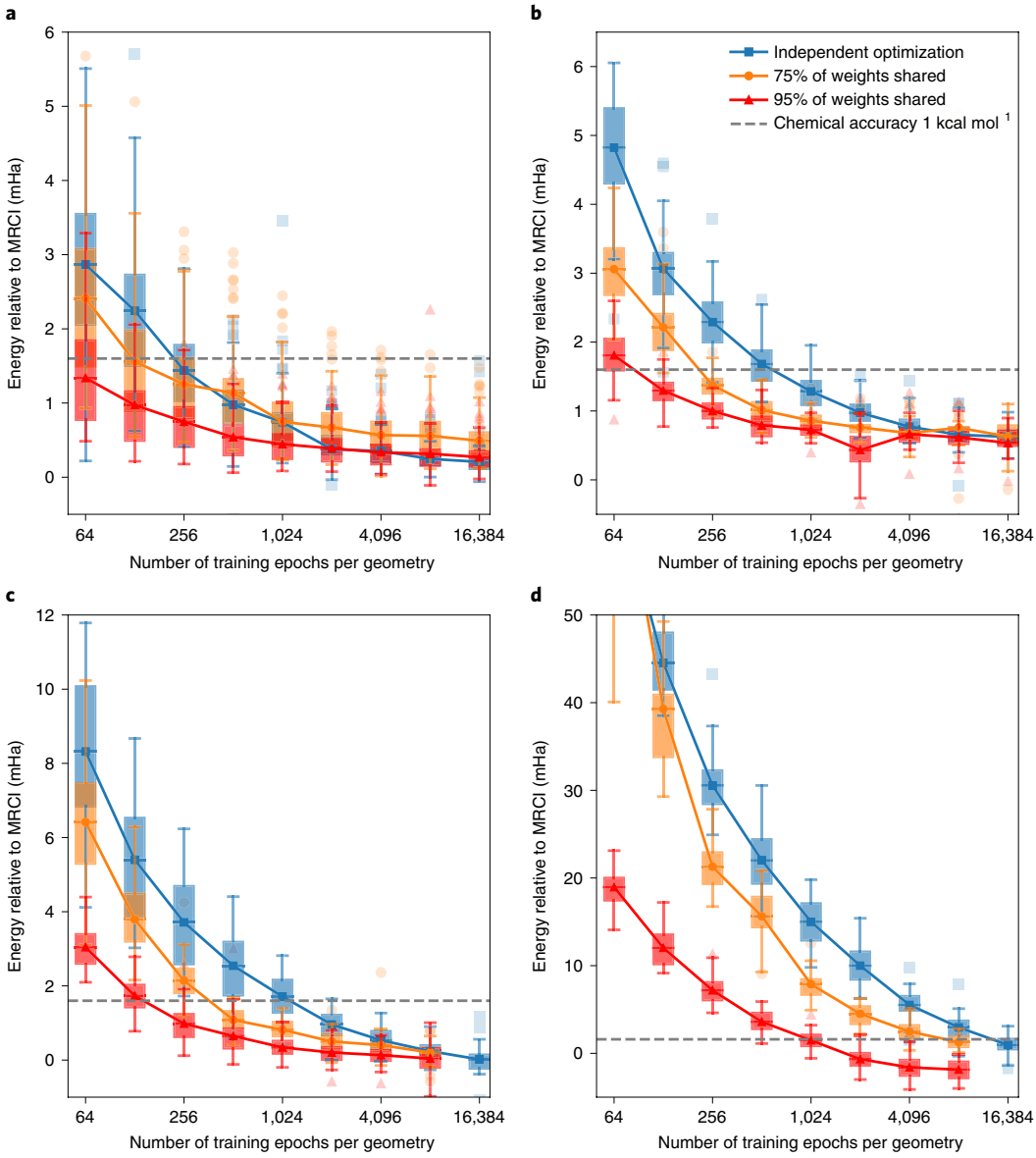

Fig. 2 | Results using weight-sharing during optimization for four different sets of molecules. Each subplot depicts the energy deviation between DeepErwin and a high-accuracy MRCI reference calculation as a function of training epochs. The three coloured lines correspond to three distinct experiments: no weight-sharing (blue), sharing 75% of weights (orange) and sharing 95% of weights (red). Boxplots show the 25–75th percentile of energy deviations, connecting lines show mean energy deviations, whiskers span the non-outlier range (1.5 interquartile ranges above and below the boxes), energy deviations beyond the whiskers are plotted individually. Dashed horizontal lines (gray) correspond to an error of 1 kcal mol^{-1} (1.6 mHa), commonly referred to as 'chemical accuracy'. **a**, Weight-sharing for 112 H_4^+ geometries. **b**, Weight-sharing for 49 H_6 geometries. **c**, Weight-sharing for 49 H_{10} geometries. Note that the reference energies here differ slightly from the reference energy in Fig. 1c, which was not available for the complete set of 49 geometries. **d**, Weight-sharing for 30 ethene geometries.

as to the electron coordinates and via an additional Jastrow factor. All of these enhancements depend on an embedding of the electron coordinates into a high-dimensional feature space that takes into account interactions with all other particles. This embedding is based on Electronic SchNet[18] and PauliNet. Cusp correction is performed explicitly[19]. By contrast to PauliNet, we use an additional equivariant backflow shift for the electron coordinates, smaller embedding networks and no residual in the embedding layers. For a realization of the wavefunction model implemented in the DeepErwin framework, the energy can be approximated through Monte Carlo integration. To eventually obtain the wavefunction of the ground state for a specific molecule, this energy is minimized by applying gradient descent steps to the free parameters of the wavefunction model. A detailed description of our architecture and the optimization procedure can be found in the Methods.

A top-level overview of the neural network parts of DeepErwin and a detailed visualization of the corresponding wavefunction model are shown in Fig. 1a,b. In comparison with PauliNet, we find that, without using weight-sharing constraints, we achieve a higher accuracy with fewer trainable weights for the small systems tested in ref. [2] when training for the same number of epochs (Fig. 1c). This indicates that the architecture implemented in DeepErwin provides a meaningful baseline to investigate the effects of weight-sharing on the optimization process. Application of weight-sharing is of course not limited to this specific model but could equally well be adapted for any neural network-based wavefunction model.

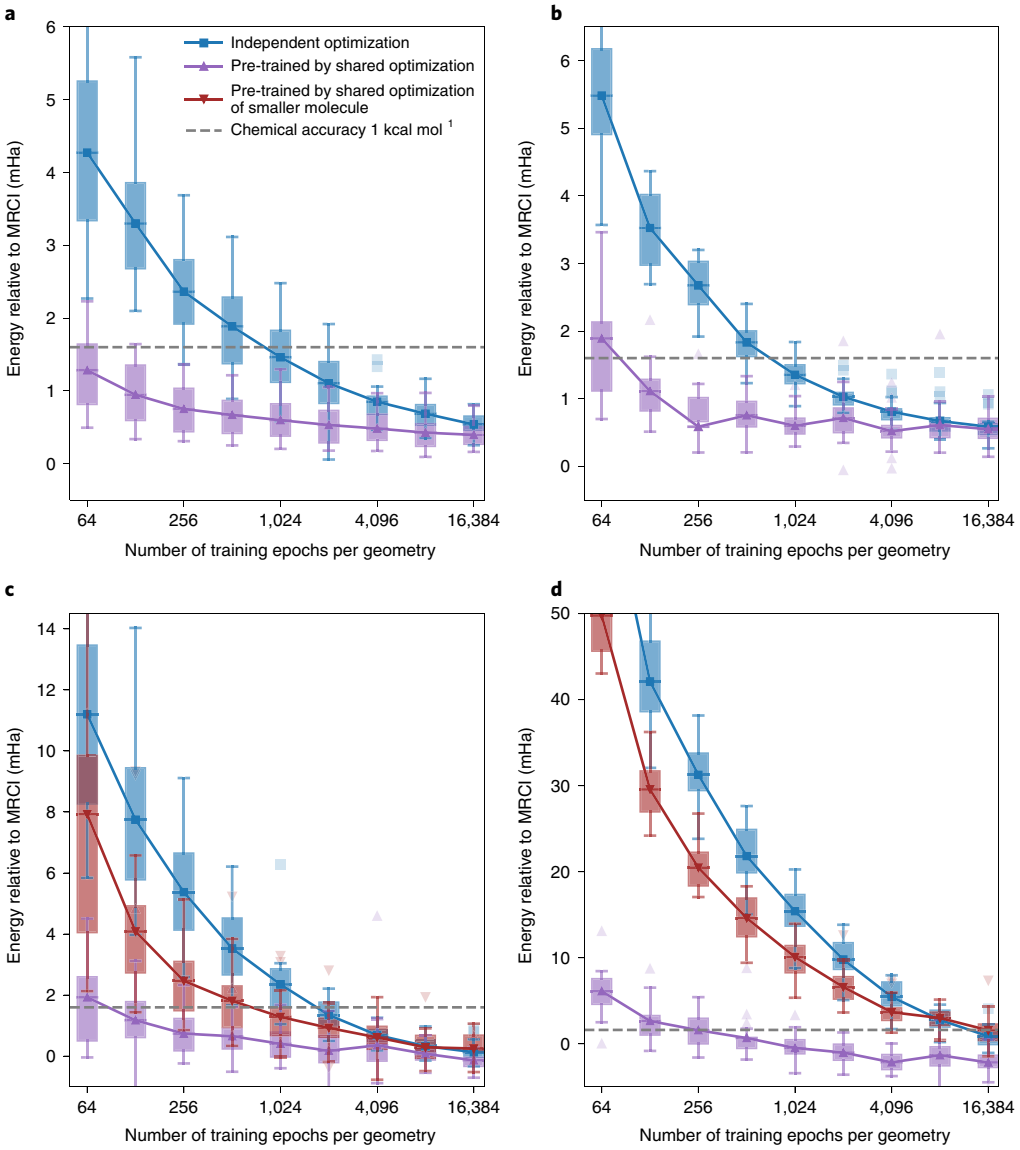

Fig. 3 | Results when reusing pretrained weights for four different sets of molecules. Each panel depicts the energy deviation between DeepErwin and a high-accuracy MRCI reference calculation as a function of training epochs. The three lines correspond to three distinct experiments: independent optimization without reusing weights (blue), reusing weights that were pretrained on different geometries of the same molecule (purple) and reusing weights which were pretrained on a different, smaller molecule (red). Boxplots as in Fig. 2. Dashed horizontal lines (gray) correspond to an error of 1 kcal mol^{-1} (1.6 mHa), commonly referred to as 'chemical accuracy'. **a**, Reusing weights for 16 H_4^+ geometries. **b**, Reusing weights for 23 H_6 geometries. **c**, Reusing weights for 23 H_{10} geometries. Pretraining on smaller molecules was performed on H_6. Note that the reference energies here differ slightly from the reference energy in Fig. 1c, which was not available for the complete set of considered H_{10} geometries. **d**, Reusing weights for 20 ethene geometries. Pretraining on smaller molecules was done on methane.

All subsequently reported results were obtained via second-order optimization using a Kronecker-factored approximate curvature (K-FAC)[20], which was already implemented for FermiNet[4]. To show that our findings are consistent across different types of optimization, we also report results regarding the accelerated optimization through weight-sharing and the applicability of weight-sharing for pretraining when using the well known Adam algorithm[21] (Supplementary Figs. 1 and 2).

Accelerated optimization through weight-sharing. The implemented architecture and the hyperparameters used in our experiments are designed to allow for a maximum number of free parameters to be potentially shared across geometries. Two parts of the model that should be particularly well suited for weight-sharing are the electron coordinate embedding and the generalized part of the backflow factor. Both basically serve as feature extractors that compute high-dimensional embeddings of electron coordinates and are therefore not necessarily required to perform geometry-specific computations. The orbital-specific part of the backflow factor, on the other hand, has a one-to-one correspondence to the orbitals yielded by CASSCF for a given geometry, suggesting that the neural network weights defining it cannot be shared across geometries in a meaningful way.

When weight-sharing is restricted to the embedding and the generalized part of the backflow factor, usually about 75% of the weights in the model are covered by weight-sharing constraints. In the most extreme case, when all weights are being shared—except the ones defining the orbital-specific backflow—this number

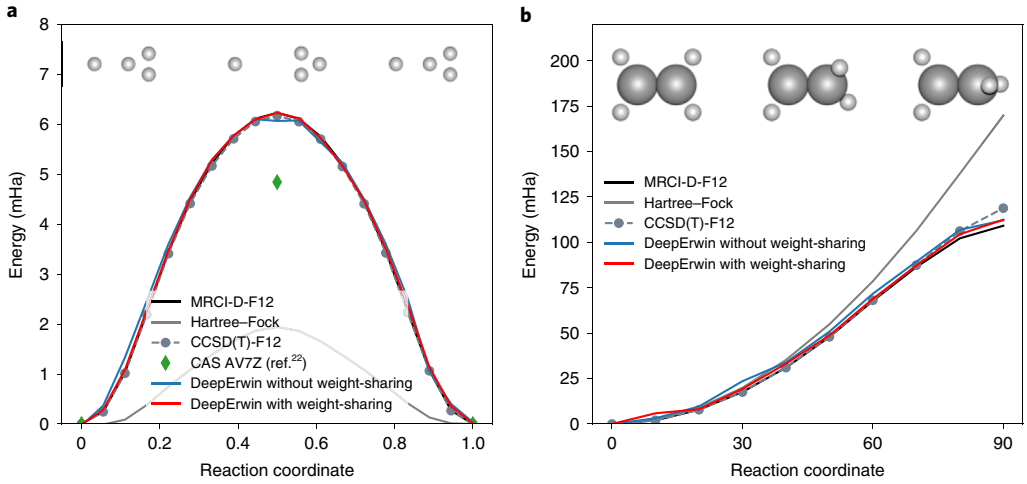

Fig. 4 | Transition barriers for H_4^+ and ethene. Insets depict molecular geometries along the transition path (large grey atoms are carbon and small white atoms are hydrogen), and graphs plot the corresponding energy differences to the ground-state energy along the path. **a**, Energy of H_4^+ along reaction path between symmetry-equivalent minima, via transition state 1 as defined in ref. [22]. DeepErwin with weight-sharing constraints is in perfect agreement with MRCI (MRCI-F12(Q)) and coupled clusters (CCSD(T)-F12) after 7,000 optimization epochs per geometry. Hartree–Fock and a complete active space calculation (CAS AV7Z) underestimate the barrier height. **b**, Energies for ten geometries that describe a twist around the carbon–carbon double bond for twisted ethene from 0° to 90°. Results for DeepErwin with and without weight-sharing are plotted after 8,192 optimization epochs per geometry.

grows to roughly 95% (Fig. 1a). Note that precise counts for the numbers of total and shared model parameters used for the experiments in this section slightly differ between different molecules (Supplementary Table 2).

We evaluate both of these set-ups by computing the PES of four different molecules; namely, H_4^+, the linear hydrogen chains H_6 and H_{10}, and ethene. For H_4^+, we consider a diverse set of 112 different configurations, covering both low-energy relaxed geometries and strongly distorted configurations. For both hydrogen chains, we calculate the wavefunction of the ground state for 49 different nuclear geometries that lie on a regular grid with respect to a parametrization based on the distance a of two adjacent hydrogen atoms and the distance x between these H_2 pairs. The corresponding PES for H_{10} is visualized in the subsequent section on calculating transition paths and forces. In the case of twisted ethene, the set of 30 geometries iterates over ten different twist angles and three different bond lengths for the C=C bond. The section on calculating transition paths and forces contains a plot of the obtained minimum-energy-path from the non-twisted equilibrium geometry to the 90° rotated molecule, considering for each twist angle the C=C bond length with the lowest energy.

For all four molecules, we compare the optimization of the wavefunction models when applying weight-sharing with the respective independent optimizations. The results of these experiments are compiled in Fig. 2. Across all physical systems, the optimization converges fastest when 95% of the weights are being shared. In particular, we find that in this case, the reference energy (MRCI-F12(Q)/cc-pVQZ-F12; see Methods for computational details) can be reached, up to chemical accuracy, between 6 and 13 times faster than when optimizing the respective geometries independently of each other.

To test our findings in this section against a more difficult benchmark, we also performed an additional experiment in which independent optimizations for different ethene configurations were fully initialized with weights from a wavefunction that had already been optimized for a similar but different molecular configuration using an independent optimization scheme. Although this approach seems to be advantageous during early optimization over a scheme that applies a weight-sharing constraint for 95% of the weights in the model, at the time that the wavefunctions reach chemical accuracy, shared optimization without pretraining yields an improvement in accuracy that is comparable with the results shown in Fig. 2. Detailed results for this experiment can be found in Supplementary Fig. 3.

Shared optimization as pretraining. Results from the previous section show that even in a setting where ground-state wavefunctions are closely approximated for a wide range of nuclear geometries by different instances of our model, the overwhelming number of free parameters can in fact be identical across those instances. This suggests that parts of our model that were successfully optimized using a weight-sharing constraint encode general computational building blocks that are well suited for the approximation of different wavefunctions. In particular, one could hope that those building blocks also generalize well to molecular geometries for which they were not previously optimized.

To test this hypothesis, we consider for each of the four molecules from the previous experiment a small new set of nuclear geometries that were not part of the original shared optimization. For these sets, we compare two types of independent optimization. In one case, a default random method is used to initialize the weights of the neural networks before optimization. In the second case, we reuse results from the previous optimization in the sense that all weights of the model that were shared in the first experiment are now initialized from the result of this optimization. For the remaining 5% of weights, default random initialization is applied.

Pushing this approach even further, we also use previously optimized shared weights to initialize wavefunction models for an entirely different molecule. In particular, we use shared weights that were optimized for the hydrogen chain H_6 to initialize models for H_{10}, and weights that were optmized for methane to initialize models for ethene. This is possible as the embedding network architecture is independent of the number of particles in the respective molecule. If successful, this method can be used to pretrain weights for large and expensive molecules by solving the Schrödinger equation for smaller, computationally cheaper systems.

The results of these experiments are shown in Fig. 3. For all four considered molecules, we used weights that were optimized with

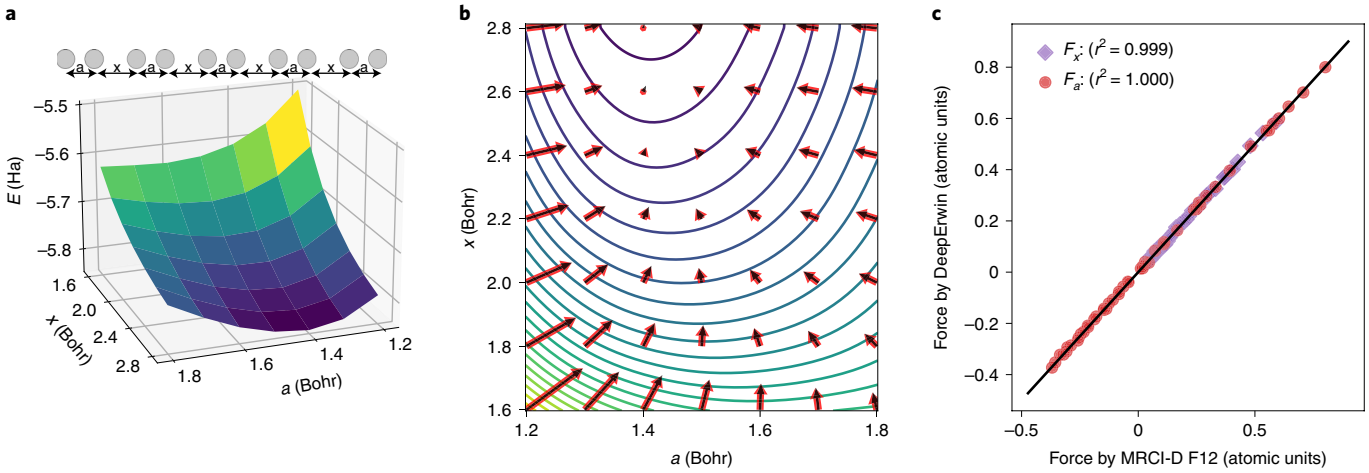

Fig. 5 | Validation of nuclear forces. a, Potential energy surface for the H_{10} chain. The variable a represents the distance between two hydrogen atoms and x is the distance between two H_2 molecules. The lowest ground state of the PES describes the dimerization. **b**, Cubic interpolation of the PES with force vectors for the H_{10} chain. Red arrows depict force vectors computed by DeepErwin via the Hellmann–Feynman theorem, whereas black arrows represent numerical gradients that are based on finite differences of MRCI-F12 reference calculations. **c**, Forces computed by DeepErwin plotted against the respective forces obtained from finite differences of the MRCI-F12 reference calculation.

a shared weight constraint on a set of different geometries (Fig. 2) for 8,192 optimization epochs per geometry. Across all systems, pretraining via shared optimization with different geometries of the same molecules dramatically accelerates the subsequent optimization such that the reference energy can be consistently reached up to chemical accuracy after little more than a hundred optimization epochs. In the case of H_{10}, the usage of weights that were pretrained on different configurations of a smaller molecule also yields substantial, albeit much smaller improvements. When using methane configurations to pretrain a wavefunction model for ethene, however, we could only find slight improvements during early optimization.

Calculating transition paths and forces. The substantial speed-ups obtained through weight-sharing enable efficient computational studies for systems that consist of many different geometries of the same molecule. We demonstrate the capabilities of our approach on two exemplary tasks: finding transition paths and calculating potential energy surfaces. As a first example, we calculate the transition path for H_4^+ between two symmetry-equivalent minima via a specific transition state previously proposed in the literature[22]. For all 19 points along the transition path, wavefunctions are optimized simultaneously for 7,000 epochs per geometry using a weight-sharing constraint that covers about 95% of total weights in the model. We furthermore compute the energies along a reaction path for twisted ethene which describes a rotation of the twist by 90°. The ten geometries considered in this task are a subsample of the 30 geometries previously used in the computations shown in Fig. 2. As a baseline, we consider independent optimization without a weight-sharing constraint as well as classical methods from computational chemistry.

The results of these calculations are shown in Fig. 4a,b. We find our method to be in agreement with high-accuracy reference calculations: DeepErwin with weight-sharing predicts barrier heights that agree with MRCI within 1 μHa (0.02%) for H_4^+, and 3.3 mHa (3%) for ethene. This leads us to believe that the barrier height for H_4^+ has been underestimated by approximately 1 mHa in previous high-accuracy calculations[22]. For the electronically challenging case of twisted ethene, both Hartree–Fock as well as CCSD(T)-F12 overestimate the energy of the 90° twisted molecule. DeepErwin, however, yields barrier energies that are much closer to the MRCI-D-F12 reference calculations.

For calculating the transition paths, we used predefined sets of geometries as a given input. In many cases, however, it is not a priori clear which nuclear geometries are of interest in a given task, and a careful exploration of the respective PES is required. Not only are energies are required to do this efficiently, but also forces on the nuclei (that is, gradients of the energy with respect to the nuclear coordinates). For realizations of the DeepErwin wavefunction model, these forces can be calculated in a straightforward and computationally efficient fashion via the Hellman–Feynman theorem[23] and by applying established variance correction schemes to accelerate convergence of the Monte Carlo integration (Methods). The PES and the corresponding forces for the linear hydrogen chain H_{10}, evaluated on a regular grid of 49 geometries, are depicted in Fig. 5a,b. The corresponding wavefunctions were optimized for 8,192 epochs per geometry using a weight-sharing constraint for approximately 95% of the model weights. We find that the energetic minimum is given by the dimerization into five H_2 molecules with a covalent bond of $a = 1.4$ Bohr each, rather than for an equally spaced arrangement of atoms. This is an instance of the well known Peierls distortion[24].

Figure 5c shows that the force vectors obtained by DeepErwin via the Hellman–Feynman theorem are in agreement with the forces computed from finite differences of MRCI-F12 reference calculations. Our computational experiments do not show any signs of spurious Pulay forces[25], which occur when the approximated wavefunction is not an eigenfunction of the Hamiltonian. This suggests that DeepErwin yields not only accurate energies, but also accurate wavefunctions.

Discussion

We found that optimized shared weights yield highly applicable initial weights when considering nuclear geometries for which the wavefunction model was not previously optimized. Even for molecules such as ethene, pretraining with shared optimization makes it possible to reach an MRCI-F12(Q)/cc-pVQZ-F12 reference calculation up to chemical accuracy after only a few hundred optimization epochs. A possibly attractive route towards making VMC optimization tractable for more complex molecules could be to pretrain large parts of a wavefunction model on small, computationally cheap systems. In our experiments we found that the

optimization of H_{10} wavefunctions can in fact be improved substantially when using shared optimization for a set of H_6 geometries to pretrain the respective models. In the case of ethene, however, we could only see small improvements during early optimization when parts of the respective model were pretrained on sets of smaller methane geometries. Due to the fact that any optimization of DNN-based wavefunction models is highly sensitive to changes in the architecture and optimization hyperparameters, we would consider our findings as preliminary evidence that merits further research towards the development of a kind of universal wavefunction, whose neural network parts were optimized for a great number of diverse molecular geometries and which is therefore capable of closely of approximating wavefunctions of the ground state for many physical systems after only a brief step of additional optimization.

Our findings suggest a strong regularity of wavefunctions within the space of nuclear geometries. Further evidence for this is also provided by a concurrent method named PESNet, which was released shortly after DeepErwin was first available as a preprint[26]. PESNet is based on the FermiNet model and employs a meta graph neural network (GNN) to simultaneously learn wavefunctions for a complete PES such that after optimization, the model parameters for a new geometry can be predicted via a simple forward pass through the meta GNN. The fact that this approach reliably yields high-accuracy results for different configurations that were sampled from a PES further underlines our observation that the regularity of wavefunctions within the space of geometries can be heavily exploited for DNN-based quantum Monte Carlo methods by also enforcing regularity on large domains of the parameter space.

Our results do not provide a conclusive answer to whether—for the investigated sets of molecular geometries—the considered weight-sharing set-ups actually limit the capability of our model to approximate the true wavefunctions of the ground state. In general, we would expect this to be the case, but judging from our experimental results, the loss of expressiveness introduced by weight-sharing regularization might often be negligible. This is evidenced by the fact that across all experiments, sharing 95% of model weights yields the same or even lower energies than the respective independent optimizations. Furthermore, it is not clear yet on what an optimal algorithm that exploits weight-sharing for a given task could look like. Based on our results so far, for an exhaustive study of the PES of a molecule, we would suggest a procedure where—possibly guided by estimates of the forces on the nuclei—geometries of interest are iteratively included in a shared optimization, which is eventually concluded by an additional step of independent optimization for some or all of the considered geometries.

The proposed method of weight-sharing is not limited to the specific architecture used in this work but could potentially be exploited for any neural network-based wavefunction model. Due to the interesting regularization properties, it could even be beneficial to apply weight-sharing in a context where only the wavefunction for a single molecular geometry is of interest.

A potential drawback of the proposed method in practice is that shared optimization cannot easily be parallelized across multiple devices (GPUs or CPUs), because each geometry is dependent on updates from all other geometries. One possibility to overcome this issue would be to consider an average loss across all geometries during gradient descent. Such a loss could easily be parallelized by using a separate device for each geometry. In our current implementation of the DeepErwin framework, however, for each epoch only a single geometry is considered during shared optimization, and it is therefore only possible to distribute the Monte Carlo samples within a batch across multiple devices, as it is common practice[7].

Methods

For a molecule with n_{nuc} nuclei, $n_\uparrow$ spin-up electrons and $n_\downarrow$ spin-down electrons, we write $\mathbf{r} = (\mathbf{r}_1, \ldots, \mathbf{r}_{n_\uparrow}, \ldots, \mathbf{r}_{n_\uparrow+n_\downarrow})$ to denote the set of cartesian electron coordinates, and $\mathbf{R} = (\mathbf{R}_1, \ldots, \mathbf{R}_{n_{\text{nuc}}})$ for the set of coordinates of nuclei. The electron coordinates $\mathbf{r}$ are always assumed to be ordered such that the first $n_\uparrow$ entries correspond to spin-up electrons, whereas the last $n_\downarrow$ entries are coordinates of spin-down electrons. We write $n_{\text{el}} = n_\uparrow + n_\downarrow$ for the total number of electrons and Z_i for the charge of the ith nucleus.

Wavefunction model. The model implemented in DeepErwin is closely related to the recently proposed PauliNet[2]. Let θ denote the set of all free (trainable) parameters in the model and n_{det} the number of enhanced Slater determinants. With a high-dimensional embedding of electron coordinates $\mathbf{x}(\mathbf{r};\mathbf{R}) = (\mathbf{x}_1, \ldots, \mathbf{x}_{n_\uparrow}, \ldots, \mathbf{x}_{n_\uparrow+n_\downarrow})$ and an explicit term $\gamma(\mathbf{r})$ for cusp correction in the Jastrow factor, a realization ψ_θ of the DeepErwin wavefunction model can be written as

$$\psi_\theta(\mathbf{r}) = e^{J(\mathbf{x}(\mathbf{r};\mathbf{R}))+\gamma(\mathbf{r})} \sum_{d=1}^{n_{\text{det}}} \alpha_d \det[\Phi_d^\uparrow(\mathbf{r}, \mathbf{x}(\mathbf{r};\mathbf{R}))] \det[\Phi_d^\downarrow(\mathbf{r}, \mathbf{x}(\mathbf{r};\mathbf{R}))], \quad (1)$$

where $\alpha_d \in \mathbb{R}$ is a trainable weight, the scalar function J defining the Jastrow factor is represented by two fully connected feedforward neural networks, and Slater determinants for spin-up electrons are defined via

$$\Phi_d^\uparrow(\mathbf{r}, \mathbf{x}(\mathbf{r};\mathbf{R})) = \begin{bmatrix} \varphi_1^{\uparrow,d}(\mathbf{r}_1 + \mathbf{s}_1(\mathbf{r}, \mathbf{x};\mathbf{R}))\eta_1^{\uparrow,d}(\mathbf{x}_1) & \cdots & \varphi_1^{\uparrow,d}(\mathbf{r}_{n_\uparrow} + \mathbf{s}_{n_\uparrow}(\mathbf{r}, \mathbf{x};\mathbf{R}))\eta_1^{\uparrow,d}(\mathbf{x}_{n_\uparrow}) \\ \vdots & \ddots & \vdots \\ \varphi_{n_\uparrow}^{\uparrow,d}(\mathbf{r}_1 + \mathbf{s}_1(\mathbf{r}, \mathbf{x};\mathbf{R}))\eta_{n_\uparrow}^{\uparrow,d}(\mathbf{x}_1) & \cdots & \varphi_{n_\uparrow}^{\uparrow,d}(\mathbf{r}_{n_\uparrow} + \mathbf{s}_{n_\uparrow}(\mathbf{r}, \mathbf{x};\mathbf{R}))\eta_{n_\uparrow}^{\uparrow,d}(\mathbf{x}_{n_\uparrow}) \end{bmatrix}. \quad (2)$$

Matrices $\Phi_d^\downarrow(\mathbf{r}, \mathbf{x}(\mathbf{r};\mathbf{R}))$ for spin-down electrons can be defined analogously. The single-electron orbitals $\varphi_i^{\uparrow,d}$ are obtained from the n_{det} most significant Slater determinants from a CASSCF method and remain fixed throughout optimization. The backflow shifts $\mathbf{s}_i$ as well as the backflow factors $\eta_i^{\uparrow,d}$ are represented by fully connected feedforward neural networks. The embedded coordinate $\mathbf{x}_i$ of the ith electron takes into account all particle positions in the system independent of particle type and spin. Further details on our implementation of the coordinate embedding, the Jastrow factor, backflow transformationand cusp correction are given below.

Electron coordinate embedding. The embedding $\mathbf{x}(\mathbf{r};\mathbf{R})$ is a slightly simplified version of the SchNet embedding used for PauliNet[2,18]. For brevity, we extend our notation to also include the nuclear coordinates in the coordinates $\mathbf{r}_j$ by defining $\mathbf{r}_j = \mathbf{R}_{j-n_{\text{el}}}$ for $n_{\text{el}} < j \leq n_{\text{nuc}}$. To embed the coordinates r_i of the ith electron, we consider input features based on pairwise differences and distances with respect to all other particles in the system. Let $i \in \{1, \ldots, n_{\text{el}}\}$, $j \in \{1, \ldots, n_{\text{el}} + n_{\text{nuc}}\}$ and n_{rbf} denote the number of radial basis features. We use $\mathbf{r}_{ij} = \mathbf{r}_i - \mathbf{r}_j$ and $r_{ij} = |\mathbf{r}_{ij}|$ to denote pairwise differences and distances, respectively, and define the pairwise feature vector

$$\mathbf{h}_{ij} = \left(e^{-(r_{ij}-\mu_1)^2/\sigma_1^2}, \ldots, e^{-(r_{ij}-\mu_{n_{\text{rbf}}})^2/\sigma_{n_{\text{rbf}}}^2}, \frac{1}{r_{ij}+0.01} \right) \in \mathbb{R}^{n_{\text{rbf}}+1+3}, \quad (3)$$

where the mean and variance parameters are defined as

$$\mu_k = cq_k^2 \text{ and } \sigma_k = \frac{1}{7}(1 + cq_k), \quad (4)$$

respectively, for an index $k \in \{1, \ldots, n_{\text{rbf}}\}$, a parameter q_k that is chosen from an equidistant grid of the interval $[0, 1]$, and a cutoff parameter $c \in \mathbb{R}$.

For a fixed integer L and an embedding dimension n_{emb}, let $\left(\mathbf{g}^l, \mathbf{w}^l_{\text{same}}, \mathbf{w}^l_{\text{op}}, \mathbf{w}^l_{\text{nuc}}\right)_{l=0}^{L}$ and $\left(\mathbf{f}^l_{\text{same}}, \mathbf{f}^l_{\text{op}}\right)_{l=1}^{L}$ denote sequences of vector-valued functions, where each function is represented by a fully connected feedforward neural network with output dimension n_{emb}. To embed the coordinates of the ith electron based on the pairwise feature vectors $\mathbf{h}_{ij}$, we use $\odot$ to denote element-wise multiplication and define for $i \in \{1, \ldots, n_{\text{el}}\}$

$$\mathbf{x}_i^0 = \mathbf{g}^0 \left(\sum_{\substack{j=1 \\ j \neq i}}^{n_{\text{el}}} \mathbf{w}^0_{\sigma_{ij}}(\mathbf{h}_{ij}) \odot \mathbf{f}^0_{\sigma_{ij}} + \sum_{j=n_{\text{el}}+1}^{n_{\text{el}}+n_{\text{nuc}}} \mathbf{w}^0_{\text{nuc}}(\mathbf{h}_{ij}) \odot \mathbf{f}^0_{Z_{j-n_{\text{el}}}} \right), \quad (5)$$

$$\mathbf{x}_i^l = \mathbf{g}^l \left(\sum_{\substack{j=1 \\ j \neq i}}^{n_{\mathrm{el}}} \mathbf{w}^l_{\sigma_{ij}}(\mathbf{h}_{ij}) \odot \mathbf{f}^l_{\sigma_{ij}}(\mathbf{x}_j^{l-1}) + \sum_{j=n_{\mathrm{el}}+1}^{n_{\mathrm{el}}+n_{\mathrm{nuc}}} \mathbf{w}^l_{\mathrm{nuc}}(\mathbf{h}_{ij}) \odot \mathbf{f}^l_{Z_{j-n_{\mathrm{el}}}} \right) \quad \text{with } 1 \leq l \leq L, \tag{6}$$

where σ_{ij} = same for same-spin pairs of electrons and σ_{ij} = op for pairs of electrons with opposite spin; $\mathbf{f}^0_{\mathrm{same}}$, $\mathbf{f}^0_{\mathrm{op}}$ and $\mathbf{f}^0_{Z_{j-n_{\mathrm{el}}}}, \ldots, \mathbf{f}^L_{Z_{j-n_{\mathrm{el}}}}$ denote trainable vectors of length n_{emb}. The embedding of the ith electron coordinates $\mathbf{r}_i$ is eventually defined as

$$\mathbf{x}_i = \mathbf{x}_i^L. \tag{7}$$

The embedding originally applied in PauliNet has an additional residual term in equation (6) and considers specific functions $\left(\mathbf{g}^l\right)_{l=0}^{L}$ for same-spin, opposite-spin and nuclear input channels.

Backflow transformation. Based on the embedding $\mathbf{x}(\mathbf{r};\mathbf{R})$, our model applies spin-dependent backflow shifts and factors to the single-electron orbitals in the Slater determinants (equation (2)). For simplicity, this section only considers the spin-up case.

Let $\eta^{\uparrow}_{\mathrm{gen}}$ denote a vector-valued function that is represented by a fully connected feedforward neural network, and $\omega_{\mathrm{bf}} \in \mathbb{R}$ a single spin-independent trainable weight. For the ith orbital and the jth electron in the dth Slater determinant, the backflow factor is computed as

$$\eta_i^{\uparrow,d}(\mathbf{x}_j) = 1 + \omega_{\mathrm{bf}} \widehat{\eta}_i^{\uparrow,d}(\eta^{\uparrow}_{\mathrm{gen}}(\mathbf{x}_j)), \tag{8}$$

where $\widehat{\eta}_i^{\uparrow,d}$ denotes an orbital-specific function that is also represented by a feedforward neural network.

The inner function $\eta^{\uparrow}_{\mathrm{gen}}$ can be seen as an extension of the embedding layer. Apart from electron spin orientation, it remains unchanged across electrons and determinants and is hence called a general backflow factor. The outer function $\widehat{\eta}_i^{\uparrow,d}$, on the other hand, is specific to the ith orbital in the dth determinant obtained from a CASSCF method. The motivation behind defining the backflow factor via the composition of these two functions was to maximize the number of neural networks weights than can possibly be shared across nuclear geometries. Note that for two distinct nuclear geometries, CASSCF does in general not yield the same number of unique orbitals. Although this does not raise an issue for the general backflow factor $\eta^{\uparrow}_{\mathrm{gen}}$, it implies that sharing the neural network weights that define $\widehat{\eta}_i^{\uparrow,d}$ across nuclear geometries is in general not possible in a meaningful way.

Based on ideas from ref. [27], the backflow shift is based on rotation-invariant features and rotation-equivariant pairwise differences. It is furthermore split into an electron–electron and an electron–nucleus part. Aside from the embedding, we also use side products of the embedding network as inputs and define the following feature vectors

$$\mathbf{f}_i^{\mathrm{el}} = (\mathbf{w}^L_{\sigma_{i1}}(\mathbf{h}_{i1}) \odot \mathbf{f}^L_{\sigma_{i1}}(\mathbf{x}_1^{L-1}), \ldots, \mathbf{w}^L_{\sigma_{in_{\mathrm{el}}}}(\mathbf{h}_{in_{\mathrm{el}}}) \odot \mathbf{f}^L_{\sigma_{in_{\mathrm{el}}}}(\mathbf{x}_{n_{\mathrm{el}}}^{L-1})) \tag{9}$$

and

$$\mathbf{f}_i^{\mathrm{nuc}} = (\mathbf{w}^L_{\mathrm{nuc}}(\mathbf{h}_{i(n_{\mathrm{el}}+1)}) \odot \mathbf{f}^L_{Z_1}, \ldots, \mathbf{w}^L_{\mathrm{nuc}}(\mathbf{h}_{i(n_{\mathrm{el}}+n_{\mathrm{nuc}})}) \odot \mathbf{f}^L_{Z_{n_{\mathrm{nuc}}}}) \tag{10}$$

for $i \in \{1, \ldots, n_{\mathrm{el}}\}$. All features are obtained without further cost at the end of the embedding loop (equation (6)). The electronic part of the shift for the ith electron is defined as

$$\mathbf{s}_i^{\mathrm{el}}(\mathbf{r}, \mathbf{x}_i, \mathbf{f}_i^{\mathrm{el}}) = \sum_{\substack{k=1 \\ k \neq i}}^{n_{\mathrm{el}}} \widehat{s}_{\mathrm{el}}(\mathbf{x}_i, \mathbf{f}_{ik}^{\mathrm{el}}) \frac{\mathbf{r}_{ik}}{1 + r_{ik}^3}. \tag{11}$$

In contrast to the backflow factor, we do not differentiate between different spins to reduce complexity. Similarly, the electron–nuclear shift is computed via

$$\mathbf{s}_i^{\mathrm{nuc}}(\mathbf{r}, \mathbf{R}, \mathbf{x}_i, \mathbf{f}_i^{\mathrm{nuc}}) = \sum_{k=n_{\mathrm{el}}+1}^{n_{\mathrm{nuc}}+n_{\mathrm{el}}} \widehat{s}_{\mathrm{nuc}}(\mathbf{x}_i, \mathbf{f}_{ik}^{\mathrm{nuc}}) \frac{\mathbf{r}_{ik}}{1 + r_{ik}^3}, \tag{12}$$

where $\widehat{s}_{\mathrm{el}}$ and $\widehat{s}_{\mathrm{nuc}}$ are represented by feedforward neural networks. A decay in the vicinity of nuclei ensures that the applied backflow shifts do not lead to a violation of the Kato cusp condition[28]. The complete backflow shift is computed by

$$\mathbf{s}_i(\mathbf{r}, \mathbf{R}, \mathbf{x}_i, \mathbf{f}_i^{\mathrm{el}}, \mathbf{f}_i^{\mathrm{nuc}}) = \omega_{\mathrm{sh}} \prod_n \tanh\left(2Z_n |\mathbf{r}_i - \mathbf{R}_n|\right)^2 \left(\mathbf{s}_i^{\mathrm{el}}(\mathbf{x}_i, \mathbf{f}_i^{\mathrm{el}}) + \mathbf{s}_i^{\mathrm{nuc}}(\mathbf{x}_i, \mathbf{f}_i^{\mathrm{nuc}})\right), \tag{13}$$

with a spin-independent trainable weight $\omega_{\mathrm{sh}} \in \mathbb{R}$.

Jastrow factor. With two spin-dependent scalar functions $J_\uparrow$ and $J_\downarrow$ that are represented by fully connected feedforward neural networks, the Jastrow factor (equation (1)) is defined as

$$J(\mathbf{x}(\mathbf{r};\mathbf{R})) = \sum_{i=1}^{n_\uparrow} J^\uparrow(\mathbf{x}_i) + \sum_{i=n_\uparrow+1}^{n_{\mathrm{el}}} J^\downarrow(\mathbf{x}_i). \tag{14}$$

Cusp correction. The Kato cusp condition[28] is a necessary condition for eigenstates of the Hamiltonian H. It ensures that the local energies of a wavefunction ψ are finite by forcing the kinetic energy term $\nabla^2\psi$ to diverge in such a way that it exactly cancels the divergence caused by the potential energy term $Z_n/|\mathbf{r}_i - \mathbf{R}_n|$ when the ith electron approaches the nth nucleus. The orbitals $\varphi_i^{\uparrow,d}$ ($\varphi_i^{\downarrow,d}$) yielded by a CASSCF method (equation (2)) in general do not lead to enhanced Slater determinants that satisfy the cusp condition. We follow the approach outlined in ref. [19] to address this issue: within a radius R_{cusp} around the nuclei, we replace the molecular orbitals by an exponentially decaying function that satisfies the Kato cusp condition, transitions smoothly into the orbitals at R_{cusp} and minimizes the variance of the local energy of this orbital. Cusp correction for the enhanced Slater determinants is performed after the initial set of orbitals has been obtained from a CASSCF calculation and remains fixed during the optimization of the wavefunction parameters θ.

To account for electron–electron cusps in the Jastrow factor (equation (1)), we use an explicit term

$$\gamma(\mathbf{r}) = \sum_{i=1}^{n_{\mathrm{el}}} \sum_{j=i+1}^{n_{\mathrm{el}}} \frac{r_{ij}}{r_{ij} + 1}, \tag{15}$$

similar to the one applied in ref. [1].

Variational Monte Carlo. We use a standard variational Monte Carlo approach to optimize our wavefunction ansatz. Let

$$H = E_{\mathrm{kin}} + E_{\mathrm{pot}} \tag{16}$$

denote the electronic Hamiltonian as obtained within the Born–Oppenheimer approximation, with

$$E_{\mathrm{kin}} = -\frac{1}{2} \sum_i \nabla_{r_i}^2, \tag{17}$$

$$E_{\mathrm{pot}} = \sum_{i>j} \frac{1}{|\mathbf{r}_i - \mathbf{r}_j|} + \sum_{n>m} \frac{Z_n Z_m}{|\mathbf{R}_n - \mathbf{R}_m|} - \sum_{i,n} \frac{Z_n}{|\mathbf{r}_i - \mathbf{R}_n|}, \tag{18}$$

where E_{kin} accounts for the kinetic energy of the electrons and E_{pot} accounts for the attraction and repulsion between particles in the system. By the Rayleigh–Ritz variational principle, it holds for any wavefunction ψ_θ that its energy E_θ is greater than or equal to the ground-state energy E_0 of the eigenfunction of H associated to the smallest eigenvalue, that is,

$$E_\theta = \int \frac{\psi_\theta(\mathbf{r}) H \psi_\theta(\mathbf{r})}{\Omega_\theta} \mathrm{d}\mathbf{r} \geq E_0, \tag{19}$$

where $\Omega_\theta = \int \psi_\theta(\mathbf{r})^2 \mathrm{d}\mathbf{r}$ denotes a normalization factor.

To evaluate the energy E_θ for a given set of parameters θ, we use Markov chain Monte Carlo integration (MCMC) and sample electron coordinates according to the probability density

$$p_\theta(\mathbf{r}) = \frac{\psi_\theta(\mathbf{r})^2}{\Omega_\theta}. \tag{20}$$

This allows us to express the total energy E_θ as the expected value of a local energy

$$E_{\mathrm{loc}}(\mathbf{r}) = \frac{H\psi_\theta(\mathbf{r})}{\psi_\theta(\mathbf{r})}, \tag{21}$$

in the sense that

$$E_\theta = \int E_{\mathrm{loc}}(\mathbf{r}) p_\theta(\mathbf{r}) d\mathbf{r} = \langle E_{\mathrm{loc}} \rangle \approx \frac{1}{N} \sum_{k=1}^{N} E_{\mathrm{loc}}(\mathbf{r}^k), \tag{22}$$

where N denotes the number of sampled electron coordinates and $\mathbf{r}^k \sim p_\theta$.

To optimize the parameters θ, we use K-FAC[20] which was already implemented in FermiNet[4] to minimize the local energy $E_{\mathrm{loc}}(\mathbf{r}^k)$ at the location of electron coordinates $\mathbf{r}^k$. DeepErwin also supports the limited-memory Broyden–Fletcher–

Goldfarb–Shanno algorithm[29], which is another second-order method, as well as standard first-order stochastic gradient descent as implemented by the Adam algorithm[21]. After each optmization epoch, (that is, after each sample of coordinates was used exactly once for a gradient descent step), the coordinates $\mathbf{r}^k$ are resampled to reflect the updated probability distribution p_θ. By applying this procedure iteratively for a large number of epochs, we eventually obtain parameters θ such that the wavefunction ψ_θ closely approximates the wavefunction of the ground state for the considered molecule.

To increase numerical stability, the implementation of our ansatz does not directly model ψ_θ, but the logarithm of its square. The local energy can then be computed as

$$E_{\mathrm{loc}}(\mathbf{r}) = E_{\mathrm{pot}}(\mathbf{r}) - \frac{1}{4}\nabla_{\mathbf{r}}^2\phi_\theta(\mathbf{r}) - \frac{1}{8}(\nabla_{\mathbf{r}}\phi_\theta)^2(\mathbf{r}), \tag{23}$$

where $\phi_\theta = \log(\psi_\theta^2)$.

Although the local energy already contains second derivatives (with respect to $\mathbf{r}$), calculating the gradient with respect to θ does not require the calculation of third derivatives, as H is Hermitian[30]. Precisely, it holds that

$$\nabla_\theta E_\theta = \langle E_{\mathrm{loc}}\nabla_\theta\phi_\theta\rangle - \langle E_{\mathrm{loc}}\rangle\langle\nabla_\theta\phi_\theta\rangle \tag{24}$$

$$\approx \frac{1}{N}\sum_{k=1}^{N} E_{\mathrm{loc}}(\mathbf{r}^k)\nabla_\theta\phi_\theta(\mathbf{r}^k) - \frac{1}{N^2}\left(\sum_{k=1}^{N} E_{\mathrm{loc}}(\mathbf{r}^k)\right)\left(\sum_{k=1}^{N}\nabla_\theta\phi_\theta(\mathbf{r}^k)\right), \tag{25}$$

for N samples of electron coordinates $\mathbf{r}^k \sim p_\theta$.

To approximate the distribution of electron coordinates defined by the density p_θ, we typically use about 2,000 independent MCMC chains (walkers), where each walker is initialized before optimization with a large number (~1,000) of burn-in steps. During optimization, walkers are updated after every epoch with a small number (~10) of extra MCMC steps. To precisely estimate the energy E_θ after optimization has concluded, we apply a similar procedure, but collect the local energies for all walkers from roughly 1,000 intermediate steps in the respective MCMC chain.

For single steps in the MCMC chains, we use a Metropolis–Hastings algorithm[31]. Given a current walker state $\mathbf{r}$, a proposal state $\mathbf{r}^{\mathrm{prop}}$ for the Metropolis–Hastings algorithm is generated according to the probability density

$$p(\mathbf{r}^{\mathrm{prop}}|\mathbf{r}) = \prod_{i=1}^{n_{\mathrm{el}}} p_{\mathcal{N}}\left(\mathbf{r}_i^{\mathrm{prop}}|\mu = \mathbf{r}_i, \sigma^2 = (d_i)^2\right), \tag{26}$$

where $p_{\mathcal{N}}$ denotes the density of a three-dimensional normal distribution and the variance parameters d_i are defined for fixed parameters d_0, $d_{\min}$, $d_{\max}$, $\delta \geq 0$ as

$$d_i = \delta \min\left(d_{\min} + \frac{|\mathbf{r}_i - \mathbf{R}_n|}{d_0}, d_{\max}\right), \tag{27}$$

where $\mathbf{R}_n$ denotes the position of the nucleus closest to $\mathbf{r}_i$, that is, $n = \mathrm{argmin}_{j=1,\ldots,n_{\mathrm{nuc}}}|\mathbf{r}_i - \mathbf{R}_j|$. We then consider the canonical acceptance probability

$$p_{\mathrm{acc}}(\mathbf{r}^{\mathrm{prop}}|\mathbf{r}) = \min\left\{1, \frac{p(\mathbf{r}|\mathbf{r}^{\mathrm{prop}})p_\theta(\mathbf{r}^{\mathrm{prop}})}{p(\mathbf{r}^{\mathrm{prop}}|\mathbf{r})p_\theta(\mathbf{r})}\right\}, \tag{28}$$

and for a sample $\alpha \in [0, 1]$ from a uniform distribution, the proposed sample $\mathbf{r}^{\mathrm{prop}}$ is accepted in the case $p_{\mathrm{acc}}(\mathbf{r}^{\mathrm{prop}}|\mathbf{r}) > \alpha$ and rejected otherwise.

The parameters d_i defined in equation (27) can be seen as step size parameters that regulate the average distance between electron coordinates $\mathbf{r}$ and a proposal $\mathbf{r}^{\mathrm{prop}}$. In general, smaller step sizes lead to higher acceptance rates for proposed samples. The parameter δ is a general step size parameter that is gradually adapted during optimization to yield an average acceptance rate of 50%. The second factor in equation (27) was specifically designed to take into account that wavefunctions are usually most complex in the proximity of a nucleus; that is, the step size d_i is chosen for each electron i to depend on its distance to the closest ion to encourage smaller steps close to the nuclei (where the wavefunction varies rapidly) and larger steps (when an electron is further away from the nuclei).

Note that the step sizes d_i, and thus the proposal probability, depend on the electron positions $\mathbf{r}$. In general it is therefore not true that $p(\mathbf{r}^{\mathrm{prop}}|\mathbf{r}) = p(\mathbf{r}|\mathbf{r}^{\mathrm{prop}})$. However, due to our choice for the acceptance probability p_{acc}, the so-called detailed balance condition is still satisfied; that is, being in a state $\mathbf{r}$ and transitioning to $\mathbf{r}^{\mathrm{prop}}$ is as probable as being in $\mathbf{r}^{\mathrm{prop}}$ and transitioning to $\mathbf{r}$.

Forces. For a given molecule, let ψ_0 denote the wavefunction of the ground state, that is, ψ_0 is the eigenfunction of the H associated with the smallest eigenvalue. To calculate the electronic forces $\mathbf{F}_m$ acting on the mth nuclei, we apply the Hellmann–Feynman theorem[23] and compute

$$\mathbf{F}_m = -\nabla_{\mathbf{R}_m}E_0 = -\frac{1}{\Omega_0}\int \psi_0(\mathbf{r})((\nabla_{\mathbf{R}_m}H)\psi_0)(\mathbf{r})\mathrm{d}\mathbf{r} \tag{29}$$

$$= Z_m\frac{1}{\Omega_0}\int \psi_0(\mathbf{r})^2\left(\sum_i \frac{\mathbf{r}_i - \mathbf{R}_m}{|\mathbf{r}_i - \mathbf{R}_m|^3}\right)\mathrm{d}\mathbf{r} - \sum_{n\neq m} Z_n\frac{\mathbf{R}_n - \mathbf{R}_m}{|\mathbf{R}_n - \mathbf{R}_m|^3} \tag{30}$$

$$\approx Z_m\left(\frac{1}{N}\sum_k\sum_i \frac{\mathbf{r}_i^k - \mathbf{R}_m}{|\mathbf{r}_i^k - \mathbf{R}_m|^3} - \sum_{n\neq m} Z_n\frac{\mathbf{R}_n - \mathbf{R}_m}{|\mathbf{R}_n - \mathbf{R}_m|^3}\right), \tag{31}$$

for N samples of electron coordinates $\mathbf{r}^k \sim \psi_0(\mathbf{r})^2/\Omega_0$, and where $\Omega_0 = \int\psi_0(\mathbf{r})^2\mathrm{d}\mathbf{r}$ denotes the L^2-norm of ψ_0.

As it is no longer necessary to compute derivatives of the wavefunction, evaluating equation (31) should be relatively easy. However, unlike the local energy (equation (22)), which has zero variance for eigenstates of the Hamiltonian, naive Monte Carlo sampling cannot be applied in this case due to the divergence of $|\mathbf{r}_i - \mathbf{R}_k|^{-3}$ when the ith electron approaches the kth nucleus.

This issue can be addressed by observing that for each diverging term on one side of a nucleus, there is an equally diverging term with the opposite sign on the other side of the nucleus. Different variance reduction methods have been proposed to exploit this property, such as fitting the force density close to a nucleus with a function that is constrained to be zero at $\mathbf{r}_i = \mathrm{R}_k$ (ref. [32]) or antithetic sampling[32,33].

We minimize the variance of force samples by combining antithetic sampling with a truncated $1/r$ potential: for each sample of electron coordinates $\mathbf{r}^k$ yielded by a Markov chain, we also consider an additional sample $\widehat{\mathbf{r}}^k$ in which each electron within a distance of R_{core} to the closest nucleus is mirrored to the opposite side of this nucleus.

$$\widehat{\mathbf{r}}^{k,i} = \mathbf{r}_i^k + 2(\mathbf{R}_m - \mathbf{r}_i^k) \tag{32}$$

for $i \in \{1, \ldots, n_{\mathrm{el}}\}$ and $m = \mathrm{argmin}_n|\mathbf{r}_i - \mathbf{R}_n|$; therefore, whenever an electron comes close to a nucleus and thus generates a large force in one direction, there will always be a cancelling contribution in the opposite direction. Note that the samples $\widehat{\mathbf{r}}^{k,i}$ do not necessarily have the same probability as the original sample $\mathbf{r}^k$. Their contribution to the Monte Carlo estimate is thus weighted by $\frac{\psi(\widehat{\mathbf{r}}^{k,i})^2}{\psi(\mathbf{r}^k)^2}$. Furthermore, we avoid numerical instabilities by replacing the raw Coulomb forces with a scaled version that decays towards zero, as electrons approach a nucleus:

$$\frac{\mathbf{r}_i^k - \mathbf{R}_m}{|\mathbf{r}_i^k - \mathbf{R}_m|^3} \longrightarrow \frac{\mathbf{r}_i^k - \mathbf{R}_m}{|\mathbf{r}_i^k - \mathbf{R}_m|^3}\tanh^3\left(\frac{|\mathbf{r}_i^k - \mathbf{R}_m|}{R_{\mathrm{cut-off}}}\right) \tag{33}$$

Shared optimization of model parameters. For N distinct nuclear geometries $\mathbf{R}^1, \mathbf{R}^2, \ldots, \mathbf{R}^N$, let θ^{sh} denote the set of model parameters that are shared across all geometries, and $\widehat{\theta}^k$ denote the set of parameters that are specific to the kth set of nuclear coordinates $\mathbf{R}^k$. The full set of parameters for the kth geometry can then be written as $\theta^k = (\theta^{\mathrm{sh}}, \widehat{\theta}^k)$, and the associated realization of the wavefunction model is denoted by ψ_{θ^k}. During each shared optimization epoch, we consider a single nuclear geometry $\mathbf{R}^k$ and update the respective model weights θ^k with respect to the local energies of the wavefunction ψ_{θ^k} at MCMC walker positions that were sampled from the probability density p_{θ^k} (equation (20)). That is, during each shared optimization epoch, not only the geometry-specific weights are updated, but also the shared weights θ^{sh}, and therefore the wavefunctions for all nuclear geometries.

A straightforward way of deciding which geometry to consider for a shared optimization epoch is to employ a simple round-robin scheme. Note that for each eigenstate of the Hamiltonian, the local energies defined in equation (21) have zero variance. Therefore, another approach to select geometries for a shared optimization epoch is to use the standard deviation of local energies as a proxy of how closely a realization of the wavefunction model already approximates the wavefunction of the ground state. By selecting the geometry for which the current wavefunction has the highest standard deviation in the local energies, we ensure that geometries for which the parameters have not yet been well optimized get more attention during optimization. In practice, we train a few initial epochs using the round-robin scheme to obtain a reliable starting point for each wavefunction and then switch to the standard-deviation-based scheme, to ensure homogeneous convergence of the accuracy across all geometries.

In our implementation of the shared optimization scheme, we use a single instance of the optimizer to update the set of shared weights θ^{sh}, as well as an additional instance for each of the geometry-specific sets of model parameters. In particular, due to the fact that geometry-specific weights receive fewer gradient descent updates than shared weights, the learning rate for the geometry-specific optimizers are usually chosen about ten times larger than the learning rate for the optimizer of the shared weights (Supplementary Table 1).

After the shared optimization process, the wavefunctions can again be treated as fully independent realizations of our wavefunction model and are not constrained by any additional dependencies. In particular, this means that the evaluation of their exact energy can easily be parallelized across geometries.

Reference calculations. To validate our deep learning method, we compared the obtained energies to reference values, which we computed using the MOLPRO package[34,35]. We employed both single- and multi-reference explicitly correlated F12 methods: coupled cluster with singles, doubles and perturbative triples (CCSD(T)-F12)[36] and MRCI-F12[37]. The basis set cc-pVQZ-F12[38] was used for all geometries of H_4^+, H_6 and C_2H_4. Due to convergence issues with this basis set for some geometries of H_{10}, the smaller cc-pVTZ-F12 basis set was employed for this molecular system. For the MRCI-F12 calculations, a CASSCF reference was used with a full-valence active space (H_4^+: three electrons in three orbitals abbreviated as (3, 3), H_6: (6,6), H_{10}: (10,10)). A full-valence active space was prohibitively expensive for C_2H_4, which is why we resorted to an active space of (2,2). The recommended GEM_BETA coefficients were used (that is, for MRCI-F12(Q)/cc-pVQZ-F12 calculations GEM_BETA = 1.5 a_0^{-1} and for MRCI-F12(Q)/cc-pVTZ-F12 calculations GEM_BETA = 1.4 a_0^{-1})[39]. The Davidson-corrected[40] energy values (MRCI-F12(Q)) were extracted, as provided by the energyd variable in MOLPRO.

We note that these accurate methods and basis sets were only used to validate our results, but not as a starting point for our deep learning method. The orbitals used as a starting point for our method were generated using the CASSCF method implemented in PySCF[41], using a 6-311G Pople basis set.

Computational settings for DeepErwin. Reference values for the results shown in Fig. 2 were obtained via MRCI-F12(Q)/cc-pVQZ-F12. A detailed description of these calculations can be found in the previous section.

In all cases, we used the described method from the previous section as a reference; however, two geometries were excluded from the reference set for the initialization from the H_{10} run, as these were also included in the pretrained geometry grid.

The main hyperparameters used for all computations are listed in Supplementary Table 1. Detailed counts of total, trainable and shared parameters for the used wavefunction models for the four main molecules considered in our numerical experiments are compiled in the Supplementary Table 2.

Data availability

All data in this manuscript were generated using the Python package DeepErwin or the quantum-chemistry code MOLPRO as described in Methods. All data required to perform the reported calculations as well as the processed data that was used to generate figures are available on Code Ocean[42]. Source data are provided with this paper.

Code availability

The DeepErwin package alongside a detailed documentation is available on the Python Package Index (PyPI) and GitHub (https://github.com/mdsunivie/deeperwin) under the MIT license. All codes and configuration files that were used to perform the reported calculations are also available on Code Ocean[42].

Received: 27 May 2021; Accepted: 16 March 2022;
Published online: 19 May 2022

References

1. Han, J., Zhang, L. & E, W. Solving many-electron Schrödinger equation using deep neural networks. *J. Comput. Phys.* **399**, 108929 (2019).
2. Hermann, J., Schätzle, Z. & Noé, F. Deep-neural-network solution of the electronic Schrödinger equation. *Nat. Chem.* **12**, 891–897 (2020).
3. Manzhos, S. Machine learning for the solution of the Schrödinger equation. *Mach. Learn. Sci. Technol.* **1**, 013002 (2020).
4. Pfau, D., Spencer, J. S., Matthews, A. G. D. G. & Foulkes, W. M. C. Ab initio solution of the many-electron Schrödinger equation with deep neural networks. *Phys. Rev. Res.* **2**, 033429 (2020).
5. Wilson, M., Gao, N., Wudarski, F., Rieffel, E. & Tubman, N. M. Simulations of state-of-the-art fermionic neural network wave functions with diffusion Monte Carlo. *Phys. Rev. Res.* **4**, 013021 (2021).
6. Bartlett, R. J. & Musiał, M. Coupled-cluster theory in quantum chemistry. *Rev. Mod. Phys.* **79**, 291–352 (2007).
7. Spencer, J. S., Pfau, D., Botev, A. & Foulkes, W. M. C. Better, faster fermionic neural networks. Preprint at https://arxiv.org/abs/2011.07125 (2020).
8. Unke, O. T. et al. Machine learning force fields. *Chem. Rev.* **121**, 10142–10186 (2021).
9. Behler, J. Four generations of high-dimensional neural network potentials. *Chem. Rev.* **121**, 10037–10072 (2021).
10. Kirkpatrick, J. et al. Pushing the frontiers of density functionals by solving the fractional electron problem. *Science* **374**, 1385–1389 (2021).
11. Westermayr, J. & Marquetand, P. Machine learning for electronically excited states of molecules. *Chem. Rev.* **121**, 9873–9926 (2020).
12. Schütt, K. T., Gastegger, M., Tkatchenko, A., Müller, K.-R. & Maurer, R. J. Unifying machine learning and quantum chemistry with a deep neural network for molecular wavefunctions. *Nat. Commun.* **10**, 5024 (2019).
13. Bogojeski, M., Vogt-Maranto, L., Tuckerman, M. E., Müller, K.-R. & Burke, K. Quantum chemical accuracy from density functional approximations via machine learning. *Nat. Commun.* **11**, 5223 (2020).
14. Faber, F. A. et al. Prediction errors of molecular machine learning models lower than hybrid DFT error. *J. Chem. Theory Comput.* **13**, 5255–5264 (2017).
15. Devlin, J., Chang, M.-W., Lee, K. & Toutanova, K. Bert: pre-training of deep bidirectional transformers for language understanding. Preprint at https://arxiv.org/abs/1810.04805 (2018).
16. Tan, C. et al. A survey on deep transfer learning. In *International Conference on Artificial Neural Networks* 270–279 (Springer, 2018).
17. Matthews, D. A. Analytic gradients of approximate coupled cluster methods with quadruple excitations. *J. Chem. Theory Comput.* **16**, 6195–6206 (2020).
18. Schütt, K. et al. SchNet: a continuous-filter convolutional neural network for modeling quantum interactions. In *Proc. 31st Conference on Neural Information Processing Systems* (eds Guyon, I. et al.) 992–1002 (Curran Associates, 2017).
19. Ma, A., Towler, M. D., Drummond, N. D. & Needs, R. J. Scheme for adding electron–nucleus cusps to Gaussian orbitals. *J. Chem. Phys.* **122**, 224322 (2005).
20. Martens, J. & Grosse, R. Optimizing neural networks with kronecker-factored approximate curvature. In *International Conference on Machine Learning* 2408–2417 (PMLR, 2015).
21. Kingma, D. P. & Ba, J. Adam: a method for stochastic optimization. Preprint at https://arxiv.org/abs/1412.6980 (2014).
22. Alijah, A. & Varandas, A. J. C. H_4^+: what do we know about it? *J. Chem. Phys.* **129**, 034303 (2008).
23. Feynman, R. P. Forces in molecules. *Phys. Rev.* **56**, 340–343 (1939).
24. Peierls, R. E. & Peierls, R. S. *Quantum Theory of Solids* (Oxford Univ. Press, 1955).
25. Pulay, P. Ab initio calculation of force constants and equilibrium geometries in polyatomic molecules. *Mol. Phys.* **17**, 197–204 (1969).
26. Gao, N. & Günnemann, S. Ab-initio potential energy surfaces by pairing GNNs with neural wave functions. In *International Conference on Learning Representations* (2022).
27. Ríos, P. L., Ma, A., Drummond, N. D., Towler, M. D. & Needs, R. J. Inhomogeneous backflow transformations in quantum Monte Carlo calculations. *Phys. Rev. E* **74**, 066701 (2006).
28. Kato, T. On the eigenfunctions of many-particle systems in quantum mechanics. *Commun. Pure Appl. Math.* **10**, 151–177 (1957).
29. Liu, D. C. & Nocedal, J. On the limited memory bfgs method for large scale optimization. *Math. Prog.* **45**, 503–528 (1989).
30. Ceperley, D., Chester, G. V. & Kalos, M. H. Monte Carlo simulation of a many-fermion study. *Phys. Rev. B* **16**, 3081–3099 (1977).
31. Hastings, W. K. Monte carlo sampling methods using Markov chains and their applications. *Biometrika* **57**, 97–109 (1970).
32. Chiesa, S., Ceperley, D. M. & Zhang, S. Accurate, efficient, and simple forces computed with quantum monte carlo methods. *Phys. Rev. Lett.* **94**, 036404 (2005).
33. Kalos, M. H. & Whitlock, P. A. *Monte Carlo Methods* (Wiley, 1986); https://cds.cern.ch/record/109491
34. Werner, H.-J., Knowles, P. J., Knizia, G., Manby, F. R. & Schütz, M. MOLPRO: a general-purpose quantum chemistry program package. *Wiley Interdiscip. Rev. Comput. Mol. Sci.* **2**, 242–253 (2012).
35. Werner, H.-J. et al. MOLPRO, version 2012.1. A package of ab initio programs (MOLPRO, 2012); https://www.molpro.net
36. Adler, T. B., Knizia, G. & Werner, H.-J. A simple and efficient CCSD(T)-F12 approximation. *J. Chem. Phys.* **127**, 221106 (2007).
37. Shiozaki, T., Knizia, G. & Werner, H.-J. Explicitly correlated multireference configuration interaction: MRCI-F12. *J. Chem. Phys.* **134**, 034113 (2011).
38. Peterson, K. A., Adler, T. B. & Werner, H.-J. Systematically convergent basis sets for explicitly correlated wavefunctions: the atoms H, He, B–Ne, and Al–Ar. *J. Chem. Phys.* **128**, 084102 (2008).
39. Hill, J. G., Mazumder, S. & Peterson, K. A. Correlation consistent basis sets for molecular core-valence effects with explicitly correlated wave functions: the atoms B–Ne and Al–Ar. *J. Chem. Phys.* **132**, 054108 (2010).
40. Langhoff, S. R. & Davidson, E. R. Configuration interaction calculations on the nitrogen molecule. *Int. J. Quantum Chem.* **8**, 61–72 (1974).
41. Sun, Q. et al. PySCF: the Python-based simulations of chemistry framework. *Wiley Interdiscip. Rev. Comput. Mol. Sci.* **8**, e1340 (2018).
42. Scherbela, M., Reisenhofer, R., Gerard, L., Marquetand, P. & Grohs, P. DeepErwin—a framework for solving the Schrödinger equation with deep neural networks. *CodeOcean* https://doi.org/10.24433/CO.8193370.v1 (2022).

Acknowledgements

We gratefully acknowledge financial support from the following grants: Austrian Science Fund FWF-I-3403 (L.G.), FWF-M-2528 (R.R.) and WWTF-ICT19-041 (L.G.). The

computational results have been achieved using the Vienna Scientific Cluster (VSC). The funders had no role in study design, data collection and analysis, decision to publish or preparation of the manuscript.

Author contributions

P.G., P.M. and R.R. conceived the project. M.S., L.G. and R.R. developed the detailed method. M.S. and L.G. wrote the Python code with contributions from R.R. The numerical experiments were designed and performed by M.S., L.G. and P.M. with support from R.R. R.R., M.S. and L.G. wrote the manuscript with input from P.G. and P.M. P.G. supervised the project. R.R. and P.G. obtained funding.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s43588-022-00228-x.

Correspondence and requests for materials should be addressed to Rafael Reisenhofer.

Peer review information *Nature Computational Science* thanks Huan Tran, Linfeng Zhang and the other, anonymous, reviewer(s) for their contribution to the peer review of this work. Peer reviewer reports are available. Primary Handling Editor: Jie Pan, in collaboration with the *Nature Computational Science* team.

Reprints and permissions information is available at www.nature.com/reprints.

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

nature communications

Article

https://doi.org/10.1038/s41467-023-44216-9

Towards a transferable fermionic neural wavefunction for molecules

Received: 22 March 2023

Accepted: 5 December 2023

Published online: 02 January 2024

Check for updates

Michael Scherbela[1,4], Leon Gerard[2,4] & Philipp Grohs[1,2,3]

Deep neural networks have become a highly accurate and powerful wavefunction ansatz in combination with variational Monte Carlo methods for solving the electronic Schrödinger equation. However, despite their success and favorable scaling, these methods are still computationally too costly for wide adoption. A significant obstacle is the requirement to optimize the wavefunction from scratch for each new system, thus requiring long optimization. In this work, we propose a neural network ansatz, which effectively maps uncorrelated, computationally cheap Hartree-Fock orbitals, to correlated, high-accuracy neural network orbitals. This ansatz is inherently capable of learning a single wavefunction across multiple compounds and geometries, as we demonstrate by successfully transferring a wavefunction model pre-trained on smaller fragments to larger compounds. Furthermore, we provide ample experimental evidence to support the idea that extensive pre-training of such a generalized wavefunction model across different compounds and geometries could lead to a foundation wavefunction model. Such a model could yield high-accuracy ab-initio energies using only minimal computational effort for fine-tuning and evaluation of observables.

Accurate predictions of quantum mechanical properties for molecules is of utmost importance for the development of new compounds, such as catalysts, or pharmaceuticals. For each molecule the solution to the Schrödinger equation yields the wavefunction and electron density, and thus in principle gives complete access to its chemical properties. However, due to the curse of dimensionality, computing accurate approximations to the Schrödinger equation quickly becomes computationally intractable with increasing number of particles. Recently, deep-learning-based Variational Monte Carlo (DL-VMC) methods have emerged as a high-accuracy approach with favorable scaling $\mathcal{O}(N^4)$ in the number of particles N[1]. These methods use a deep neural network as ansatz for the high-dimensional wavefunction, and minimize the energy of this ansatz to obtain the ground-state wavefunction. Based on two major architectures for the treatment of molecules in first quantization, PauliNet[1] and FermiNet[2], several improvements and applications have emerged. On the one hand, enhancements of architecture, optimization and overall approach have led to substantial improvements in accuracy or computational cost[3–7]. On the other hand, these methods have been adapted to many different systems and observables: model systems of solids[8,9], real solids[10], energies and properties of individual molecules[1,2,5,11], forces[12,13], excited states[14] and potential energy surfaces[13,15,16]. Furthermore, similar methods have been developed and successfully applied to Hamiltonians in second quantization[17,18].

We want to emphasize that DL-VMC is an ab-initio method, that does not require any input beyond the Hamiltonian, which is defined by the molecular geometry. This differentiates it from surrogate models, which are trained on results from ab-initio methods to either predict wavefunctions[19,20] or observables[21].

Despite the improvements in DL-VMC, it has not yet been widely adopted, in part due to the high computational cost. While DL-VMC offers favorable scaling, the method suffers from a large prefactor, caused by an expensive optimization with potentially slow convergence towards accurate approximations. Furthermore this

[1]Faculty of Mathematics, University of Vienna, Vienna, Austria. [2]Research Network Data Science, University of Vienna, Vienna, Austria. [3]Johann Radon Institute for Computational and Applied Mathematics, Austrian Academy of Sciences, Linz, Austria. [4]These authors contributed equally: Michael Scherbela, Leon Gerard. e-mail: philipp.grohs@univie.ac.at

optimization needs to be repeated for every new system, leading to prohibitively high computational cost for large-scale use. This can be partially overcome by sharing a single ansatz with identical parameters across different geometries of a compound, allowing more efficient computation of Potential Energy Surfaces (PES)[13,15,16]. However, these approaches have been limited to different geometries of a single compound and do not allow successful transfer to new compounds. A key reason for this limitation is that current architectures explicitly depend on the number of orbitals (and thus electrons) in a molecule. Besides potential generalization issues, this prevents a transfer of weights between different compounds already by the fact that the shapes of weight matrices are different for compounds of different size.

In this work we propose a neural network ansatz, which does not depend explicitly on the number of particles, allowing to optimize wavefunctions across multiple non-periodic, gas-phase compounds with multiple different geometric conformations. We find, that our model exhibits strong generalization when transferring weights from small molecules to larger, similar molecules. In particular we find that our method achieves high accuracy for the important task of relative energies. Our approach is inspired by the success of foundation models in language[22] or vision[23,24], where models are extensively pre-trained and then applied to new tasks—either without any subsequent training (referred to as zero-shot evaluation) or after small amount of training on the new task (referred to as fine-tuning). Zhang et al.[25] have shown that this paradigm can be successfully applied to wavefunctions, in their case for model Hamiltonians in second quantization.

In this work, we pre-train a first base-model for neural network wavefunctions in first quantization, and evaluate the pre-trained model by performing predictions on chemically similar molecules (in-distribution) and disparate molecules (out-of-distribution). We find that our ansatz outperforms conventional high-accuracy methods such as CCSD(T)-ccpVTZ and that fine-tuning our pre-trained model reaches this accuracy ≈ 20x faster, than optimizing a new model from scratch. When analyzing the accuracy as a function of pre-training resources, we find that results systematically and substantially improve by scaling up either the model size, data size or number of pre-training steps. These results could pave the way towards a foundation wavefunction model, to obtain high-accuracy ab-initio results of quantum mechanical properties using only minimal computational effort for fine-tuning and evaluation of observables.

Additionally we compare our results to GLOBE, a concurrent work[26], which proposes reparameterization of the wavefunction based on machine-learned, localized molecular orbitals. We find that for the investigated setting of re-using pre-trained weights our method in comparison achieves lower (and thus more accurate) absolute energies, higher accuracy of relative energies and is better able to generalize across chemically different compounds.

We use the following notation throughout this work: All vectors, matrices and tensors are denoted in **bold letters**, including functions with vectorial output. The i-th electron position for $i \in \{1, \ldots, n_{\text{el}}\}$ is denoted as $\mathbf{r}_i \in \mathbb{R}^3$. The set $\{\mathbf{r}_1, \ldots, \mathbf{r}_{n_\uparrow}\}$ of all electrons with spin up is abbreviated with $\{\mathbf{r}_\uparrow\}$, the set $\{\mathbf{r}_{n_\uparrow+1}, \ldots, \mathbf{r}_{n_{\text{el}}}\}$ of all spin-down electrons as $\{\mathbf{r}_\downarrow\}$. Similarly, the nuclear positions and charges of a molecule are denoted by $\mathbf{R}_I \in \mathbb{R}^3$ and $Z_I \in \mathbb{N}$, $I = 1, \ldots, N_{\text{atoms}}$, with the set of all nuclear positions and their corresponding charges denoted as $\{(\mathbf{R}, \mathbf{Z})\}$. Indices $i, k \in \{1, \ldots, n_{\text{el}}\}$ correspond to electrons and orbitals respectively, whereas $I, J \in \{1, \ldots, N_{\text{atoms}}\}$ correspond to nuclei. By $\langle \cdot, \cdot \rangle$ the dot product is denoted.

Results

In the following, we briefly outline our approach and how it extends existing work (A multi-compound wavefunction ansatz). We show the fundamental properties of our ansatz such as extensivity and equivariance with respect to the sign of reference orbitals. We demonstrate the transferability of the ansatz when pre-training on small molecules and re-using it on larger, chemically similar molecules. We also compare its performance against GLOBE, a concurrent work[26]. Lastly, we present a first wavefunction base model pre-trained on a large diverse dataset of 360 geometries and evaluate its downstream performance.

A multi-compound wavefunction ansatz

Existing high-accuracy ansätze for neural network wavefunctions ψ all exhibit the following structure:

$$\begin{aligned} \mathbf{h}_i &= \mathbf{h}_\theta(\mathbf{r}_i, \{\mathbf{r}_\uparrow\}, \{\mathbf{r}_\downarrow\}, \{(\mathbf{R}, \mathbf{Z})\}) \\ \mathbf{h}_i &\in \mathbb{R}^{D_{\text{emb}}}, \quad i = 1 \ldots n_{\text{el}} \end{aligned} \tag{1}$$

$$\begin{aligned} \Phi^d_{ik} &= \varphi^d_k(\mathbf{r}_i)\langle \mathbf{F}^d_k, \mathbf{h}_i \rangle \\ \varphi^d_k(\mathbf{r}_i) &: \mathbb{R}^3 \to \mathbb{R}, \quad \mathbf{F}^d_k \in \mathbb{R}^{D_{\text{emb}}} \\ i, k &= 1 \ldots n_{\text{el}}, \quad d = 1 \ldots N_{\text{det}} \end{aligned} \tag{2}$$

$$\psi = \sum_{d=1}^{N_{\text{det}}} \det\left[\Phi^d_{ik}\right]_{i,k=1\ldots n_{\text{el}}} \tag{3}$$

The neural network $\mathbf{h}_\theta$ in eq. (1) computes a D_{emb}-dimensional embedding $\mathbf{h}_i$ of electron i, by taking in information of all other particles, e.g., by using attention or message passing. Eq. (2) maps these high-dimensional embeddings onto $n_{\text{el}} \times N_{\text{det}}$ orbitals (indexed by k), using trainable backflow matrices $\mathbf{F}^d$ and typically trainable envelope functions $\varphi^d_k : \mathbb{R}^3 \to \mathbb{R}$. Eq. (3) evaluates the final wavefunction ψ as a sum of (Slater-)determinants of these orbitals, to ensure antisymmetry with respect to permutation of electrons.

By considering the interaction of all particles, in particular with the sets $\{\mathbf{r}_\uparrow\}$ and $\{\mathbf{r}_\downarrow\}$, the functions in Eq. (2) account for the interparticle correlation and therefore are able to better represent the true ground-state wavefunction. If the orbitals Φ^d_{ik} would only depend on $\mathbf{r}_i$ (instead of the many-body embedding $\mathbf{h}_i$), they would correspond to single-particle functions, e.g. Hartree-Fock orbitals. Existing methods, as proposed in Pfau et al.[2] or Hermann et al.[1], differ in the way the embedding $\mathbf{h}_i$ and the envelope functions φ^d_k are built. A popular choice for the embedding function $\mathbf{h}_\theta$ are continuous convolutions[1,5,26] or an attention mechanism[4]. For the envelope functions Hermann et al.[1] proposed to use orbitals obtained from a Hartree-Fock calculation, whereas Pfau et al.[2] relied on exponentially decaying envelopes, (i.e., $\varphi_k(\mathbf{r}) = \exp(-\alpha_{kI}|\mathbf{r} - \mathbf{R}|)$ with trainable parameter $\alpha_{kI} \in \mathbb{R}$), to ensure the boundary conditions far away from the nuclei. In our architecture, we mainly focus on the matrix $\mathbf{F}^d = [\mathbf{F}^d_1, \ldots, \mathbf{F}^d_{n_{\text{el}}}] \in \mathbb{R}^{n_{\text{el}} \times D_{\text{emb}}}$. While the mapping by $\mathbf{F}^d$ works well for the wavefunction of a single compound, it is fundamentally unsuited to represent wavefunctions for multiple different compounds at once, since its dimension $n_{\text{el}} \times D_{\text{emb}}$ depends explicitly on the number of electrons. There are several potential options, how this challenge could be overcome. A naïve approach would be to generate a fixed number of $N_{\text{orb}} \geq n_{\text{el}}$ orbitals and truncate the output to the required number of orbitals n_{el}, which may differ across molecules. While simple to implement, this approach is however fundamentally limited to molecules with $n_{\text{el}} \leq N_{\text{orb}}$. Another approach is to use separate matrices $\mathbf{F}^d_{\mathcal{G}}$ for each molecule or geometry $\mathcal{G}$, as was done in[13], but also this approach can fundamentally not represent wavefunctions for molecules that are larger than the ones found in the training set. A third approach would be to not generate all orbitals in a single pass, but generate the orbitals sequentially in an auto-regressive manner, by conditioning each orbital on the previously generated orbitals. While this approach has been successful in other domains such as language processing, it suffers from inherently poor parallelization due to its sequential nature. A final approach—chosen in this work—is to replace the tensor $\mathbf{F}$ with a trainable function $\mathbf{f}^o_\theta(\mathbf{c}_{Ik})$, which computes the

backflows based on some descriptor $\mathbf{c}_{Ik} \in \mathbb{R}^{N_{\text{basis}}}$ of the orbital k to be generated:

$$\varphi_\theta^d(\mathbf{r}_i,\mathbf{R}_I,\mathbf{c}_{Ik}) = \exp\left(-|\mathbf{r}_i - \mathbf{R}_I|\, g_\theta^{e,d}(\mathbf{c}_{Ik})\right) \tag{4}$$

$$\Phi_{ik}^d = \sum_{I=1}^{N_{\text{atoms}}} \varphi_\theta^d(\mathbf{r}_i,\mathbf{R}_I,\mathbf{c}_{Ik}) \left\langle \mathbf{f}_\theta^{o,d}(\mathbf{c}_{Ik}), \mathbf{h}_i \right\rangle \tag{5}$$

Similar to $\mathbf{f}_\theta^o$, the trainable function $g_\theta^{e,d}$ also maps some descriptors to scalar values for each orbital. While there are several potential descriptors $\mathbf{c}_{Ik}$ for orbitals, one particularly natural choice is to use outputs of computationally cheap, conventional quantum chemistry methods such as Density Functional Theory or Hartree-Fock. We compute orbital features based on the expansion coefficients of a Hartree-Fock calculation, by using orbital localization and a graph convolutional network (GCN), as outlined in the methods section "Obtaining orbital descriptors from Hartree-Fock". We then map these features to orbitals Φ_{ik}^d, which we call in the following *transferable atomic orbitals (TAOs)*, using odd and even functions $\mathbf{f}_\theta^o$ and g_θ^e as illustrated in Fig. 1.

Properties of our ansatz

These TAOs fulfill many properties, which are desirable for a wavefunction ansatz:

- **Constant parameter count**: In principle, the number of parameters in the ansatz is independent of system size. While it might still be necessary to increase the parameter count to maintain uniform accuracy for systems of increasing size, TAOs have no explicit relationship between parameter count and system size. This is in contrast to previous approaches[1,2,13] where the number of parameters grows explicitly with the number of particles, making it impossible to use a single ansatz across systems of different sizes. In particular, backflows and envelope exponents have typically been chosen as trainable parameters of shape $[N_{\text{orb}} \times N_{\text{det}}]$. In our ansatz the backflows $\mathbf{F}$ are instead computed by a single function $\mathbf{f}_\theta$ from multiple inputs $\mathbf{c}_{Ik}$.
- **Equivariant to sign of HF-orbital**: Orbitals of a HF-calculation are obtained as eigenvectors of a matrix and are thus determined only up to their sign (or their phase in the case of complex eigenvectors). We enforce that the functions $\mathbf{f}_\theta^o, g_\theta^e$ are odd and even with respect to $\mathbf{c}_{Ik}$. Therefore our orbitals Φ_{ik}^d are equivariant to a flip in the sign of the HF-orbitals used as inputs: $\Phi(-\mathbf{c}_{Ik}) = -\Phi(\mathbf{c}_{Ik})$. Therefore during supervised pre-training, the undetermined sign of the reference orbitals becomes irrelevant, leading to faster convergence as demonstrated in "Equivariance with respect to HF-phase".
- **Approximate locality**: When using localized HF-orbitals as input, the resulting TAOs are typically also localized. Localized HF-orbitals are orbitals which have non-zero orbital coefficients $\tilde{\boldsymbol{\alpha}}_{Ik}$ only on some subset of atoms. Using our architecture outlined in "Obtaining orbital descriptors from Hartree-Fock", this typically translates into local orbital features $\mathbf{c}_{Ik}$. Since we enforce the backflow $\mathbf{f}_\theta^o$ to be odd (and thus $\mathbf{f}_\theta^o(\mathbf{0}) = \mathbf{0}$), the resulting TAOs have zero contribution from atoms I with $\mathbf{c}_{Ik} = \mathbf{0}$. While the true wavefunction might not be fully decomposable into purely local contributions, many relevant chemical concepts —such as core electrons, bonds, lone pairs, or chemical groups— are intrinsically local concepts and locality has been successfully used as prior in many applications, for example in Neural Network Potentials[27,28]. This hints at the prior of using local orbitals to compose many-body wavefunctions, which in turn can still contain non-local electron-electron correlations captured via the fully connected, non-local electron embedding $\mathbf{h}_i$.
- **High expressivity**: We empirically find that our ansatz is sufficiently expressive to model ground-state wavefunctions to high accuracy. We demonstrate this both empirically (c.f. SI 1, 1 mHa energy deviation against PsiFormer for NH_3) and theoretically (c.f. SI 2) in the supplementary information. This stands in contrast to previous approaches based on incorporating ab-initio orbitals[1], which could not reach chemical accuracy even for small molecules.

Size consistency of the ansatz

One design goal of the ansatz is to allow transfer of weights from small systems to larger systems. In particular, if a large system consists of many small previously seen fragments, one would hope to obtain an energy which corresponds approximately to the sum of the fragment energies. One simple test case, are chains of equally spaced Hydrogen atoms of increasing lengths. These systems have been studied extensively using high-accuracy methods[29], because they are small systems which already show strong correlation and are thus challenging to solve. We test our method by pre-training our ansatz on chains of length 6 and 10, and then evaluating the model (with and without subsequent fine-tuning) for chain lengths between 2 and 28. Figure 2a shows that our ansatz achieves very high zero-shot-accuracy in the interpolation regime ($N_{\text{atoms}} = 8$) and for extrapolation to slightly larger or slightly smaller chains ($N_{\text{atoms}} = 4, 12$). Even when extrapolating to systems of twice the size ($N_{\text{atom}} = 20$), our method still outperforms a Hartree-Fock calculation and eventually converges to an energy close to the Hartree-Fock solution.

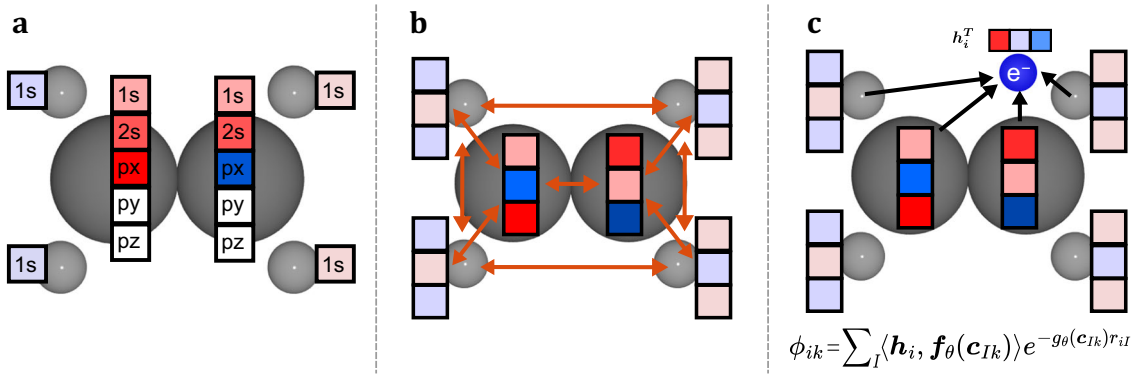

Fig. 1 | Illustration of the Transferable Atomic Orbitals, demonstrated on the C=C-bond of Ethene. a The input for each orbital are localized Hartree-Fock basis expansion coefficients $\tilde{\boldsymbol{\alpha}}_I$, corresponding to every atom I. **b** We learn a representation $\mathbf{c}_I$ of the orbital on every atom using a Graph Neural Network, exchanging information across atoms. **c** The orbital ϕ is evaluated for electron i by combining the electron-embedding $\mathbf{h}_i$ with the functions of the orbital representation $f_\theta(\mathbf{c}_I)$ and $g_\theta(\mathbf{c}_I)$.

https://doi.org/10.1038/s41467-023-44216-9

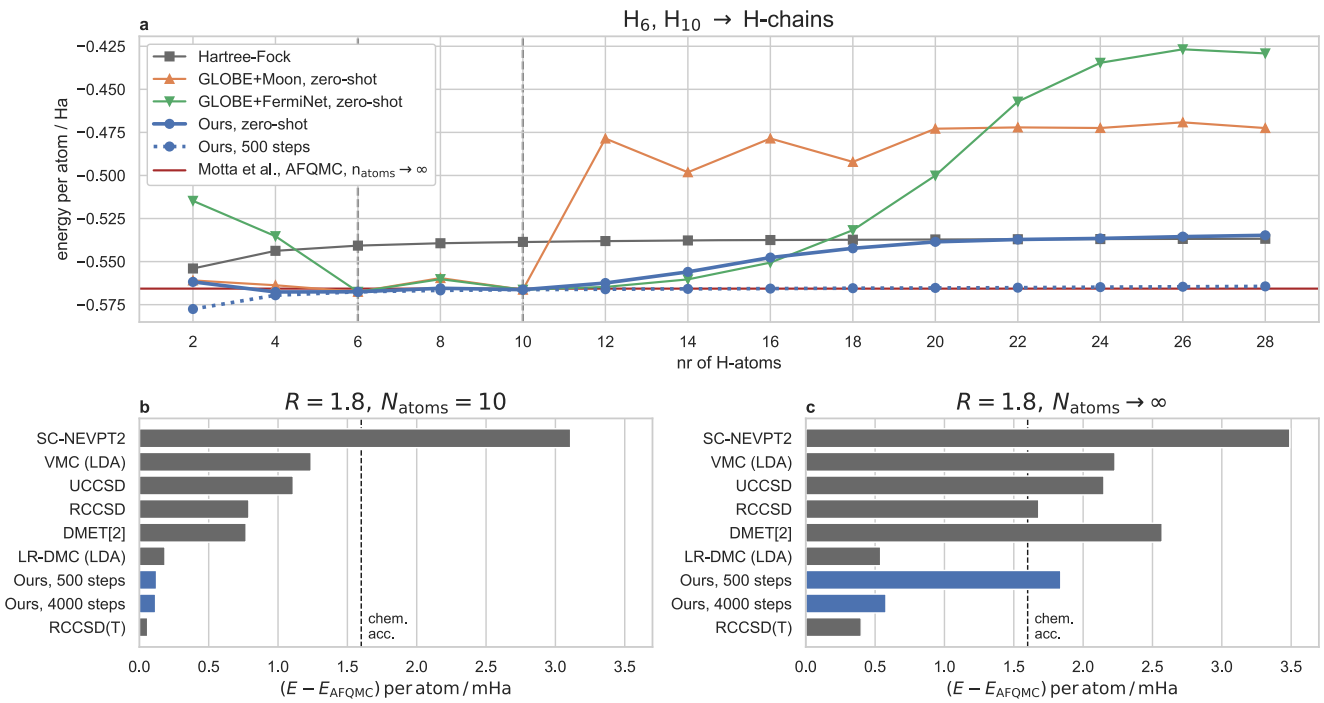

Fig. 2 | Transferability of the ansatz to chemically similar, larger systems, demonstrated on the example of hydrogen-chains. a Energy per atom as a function of chain-length. While GLOBE cannot successfully transfer to larger chains, our ansatz successfully predicts zero-shot energies (i.e. without fine-tuning) for up to 2x longer chains. **b**, **c** Comparison of our energies per atom after 500 and 4000 fine-tuning steps vs. high-accuracy methods from Motta et al.[29]. The methods compared in this work include: SC-NEVPT2, the strongly contracted variant of the n_{el} electron valence state second-order pertubation theory; VMC (LDA), variational Monte Carlo (local density approximation); UCCSD, couple cluster theory with full treatment of singles and doubles excitations; RCCSD and RCCSD(T), couple cluster theory with full treatment of singles and doubles and perturbative treatment of triple excitations using restricted Hartree-Fock as a reference state; DMET density-matrix embedding theory, LR-DMC (LDA) lattice-regularized diffusion Monte Carlo (local density approximation). **b** for the Hydrogen Chain with number of atoms $N_{atoms} = 10$ (**c**) for chain lengths extrapolated to the thermodynamic limit ($N_{atoms} \to \infty$).

To reach the accuracy of other correlated methods, we need a few fine-tuning steps for each new system. In Fig. 2b, c, we compare our results after 500 and 4000 fine-tuning steps against all high-accuracy methods from Motta et al.[29], which can obtain energies extrapolated to the thermodynamic limit (TDL). We compare for the two system sizes, investigated in ref. 29: 10 atoms in Fig. 2b, and extrapolation to $N_{atoms} = \infty$ in Fig. 2c. For H_{10}, our method is in near perfect agreement with their reference method AFQMC, deviating only by 0.1 mHa, nearly independent of the number of fine-tuning steps. This high-accuracy result is expected, since our model has also been pre-trained on chains of length 10 (albeit with different inter-atomic distances), and DL-VMC has previously been shown to achieve very high-accuracy on this system[1]. When extrapolating to the TDL, our zero-shot energies are not competitive with high-accuracy methods, but instead yield energy errors comparable to Hartree-Fock, as seen in Fig. 2a. However, fine-tuning the ansatz for only 500 steps, yields energies that already outperform most methods studied in ref. 29 and fine-tuning for 4000 steps yields a deviation of 0.6 mH/atom vs. AFQMC, on par with specialized methods such as LR-DMC.

This good performance stands in stark contrast to other approaches such as GLOBE+ FermiNet or GLOBE+Moon, studied in ref. 26: Both GLOBE-variants yield 5-6x higher errors in the interpolation regime and both converge to much higher energies for larger chains. While our approach yields Hartree-Fock-like energies for very long chains, GLOBE+FermiNet and GLOBE+Moon yield results that are outperformed even by assuming a chain of non-interacting H-atoms, which would yield an energy per atom of -0.5 Ha. For modest extrapolations ($N_{atoms} = 12$ to $N_{atoms} = 20$) our zero-shot results yield 3–20x lower errors than GLOBE+Moon.

Equivariance with respect to HF-phase

Due to using even and odd functions for the TAOs, our orbitals are equivariant with respect to a change of sign of the Hartree Fock orbitals. Therefore, a sign change of the HF-orbitals during HF-pre-training has no effect on the optimization of the wavefunction. One test case to assure this behavior is the rotation of a H_2O molecule, where we consider a set of 20 rotations of the same geometry, leading to a change of sign in the p-orbitals of the Oxygen atom (cf. Fig. 3). We evaluate our proposed architecture and compare it against a naïve approach, where we use a standard backflow matrix $\mathbf{F}$, instead of a trainable, odd function $\mathbf{f}^{o}_{\theta}$. In Fig. 3 we can see a clear spike in the HF-pre-training loss at the position of the sign flip for the standard backflow-type architecture, causing slower convergence during the subsequent variational optimization. After 16k optimization steps the effect diminishes and no substantial improvement on the accuracy can be observed. Although in this specific instance the orbital sign problem could also be overcome without our approach by correcting the phase of each orbital to align them across geometries, phase alignment is not possible in all circumstances. For example, there are geometry trajectories, where the Berry phase prevents such solutions[30].

Transfer to larger, chemically similar compounds

To test the generalization and transferability of our approach, we perform the following experiment: First, we train our ansatz on a dataset of multiple geometries of a single, small compound (e.g. 20 distorted geometries of Methane). For this training, we follow the usual procedure of supervised HF-pre-training and subsequent variational optimization as outlined in the methods section "Variational Monte Carlo". After 64k variational optimization steps, we

https://doi.org/10.1038/s41467-023-44216-9

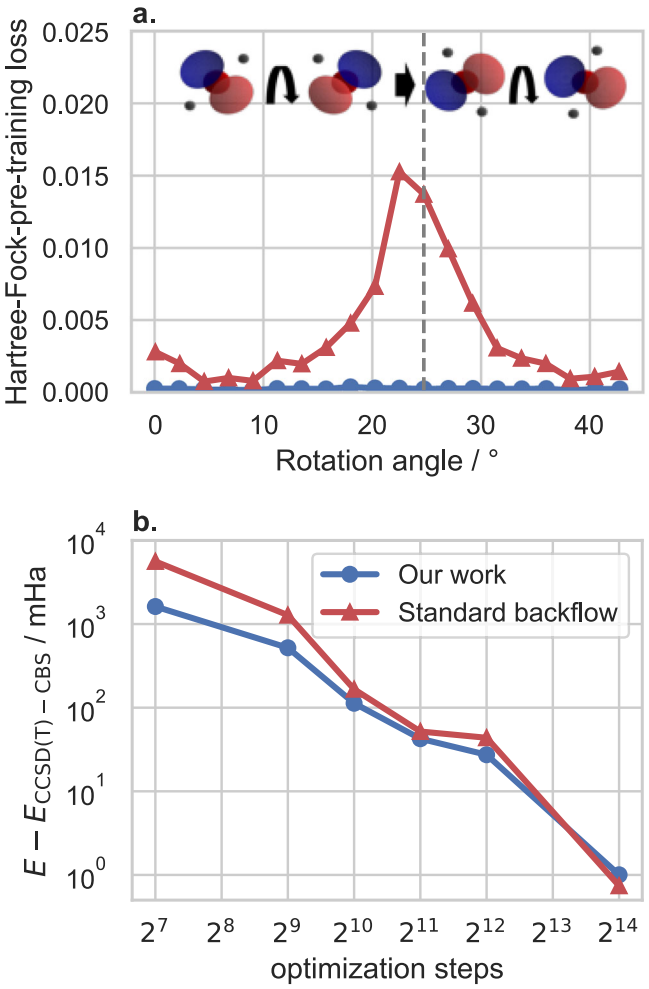

Fig. 3 | Accuracy when Hartree-Fock-pre-training against rotated H_2O molecules, which contain a change of sign in the Hartree-Fock-p-orbitals of the Oxygen atom (a). Comparing a shared optimization of a backflow-based neural network wavefunction (Standard backflow) against transferable atomic orbitals (Our work). **a** Hartree-Fock-pre-training loss of the last the last 100 Monte Carlo samples for 20 rotated geometries (**b**): Mean energy error vs. couple cluster reference calculations ($E_{CCSD(T)}$), averaged across all geometries.

then re-use the weights for different geometries of a larger compound (e.g. distorted geometries of Ethene). We fine-tune the model on this new geometry dataset for a variable number of steps and plot the resulting energy errors in Fig. 4. We do not require supervised HF-pre-training on the new, larger dataset. We perform this experiment for 3 pairs of test systems: Transferring from geometries of Hydrogen-chains with 6 atoms each, to chains with 10 atoms each, transferring from Methane to Ethene, and transferring from Ethene to Cyclobutadiene. These test systems are of interst, because they show strong correlation, despite being relatively small and computationally cheap systems. For example, even CCSD(T) overestimates the energy barriers of the Ethene- and Cyclobutadiene-PES by ≈10 mHa[13,31].

We compare our results to the earlier DeepErwin approach[13], which only partially reused weights, and GLOBE, a concurrent work[26] which reuses all weights. To measure accuracy we compare two important metrics: First, the mean energy error (averaged across all geometries g of the test dataset) $\frac{1}{N}\sum_g (E_g - E_g^{\text{ref}})$, which reflects the method's accuracy for absolute energies (cf. Fig. 4a). Second, the deviation of the relative energy between the highest and lowest point of the PES, i.e. $\Delta E - \Delta E_{\text{ref}} = (E_{\max} - E_{\min}) - (E_{\max}^{\text{ref}} - E_{\min}^{\text{ref}})$, plotted in Fig. 4b. Since different studies use different batch-sizes and different definitions of an epoch, we plot all results against the number of samples used for the energy estimation during variational optimization, which is very closely linked to computational cost.

Compared to other approaches, we find that our method yields substantially lower and more consistent energies. On the toy problem of H_6 to H_{10} our approach and GLOBE reach the same accuracy, while DeepErwin converges to higher energies. For the real-world molecules Ethene (C_2H_4) and Cyclobutadiene (C_4H_4) our approach reaches substantially lower (and thus more accurate) energies and much more consistent potential energy surfaces. After 64 mio. fine-tuning samples, our mean absolute energies are 16 mHa and 17 mHa lower than GLOBE, and our relative energies are 39 mHa and 20 mHa closer to the reference calculation. When inspecting the resulting Potential Energy Surface for Ethene (Fig. 4c), we find that we obtain qualitatively similar results as DeepErwin and MRCI, but obtain energies that are ≈ 6 mHa lower (and thus more accurate). GLOBE on the other hand does not yield the correct PES for this electronically challenging problem, since it overestimates the energy barrier at 90° twist angle by ≈50 mHa. We observe similar results on the Cyclobutadiene geometries, where our approach yields relative energies that are in close agreement to the reference method, while the GLOBE-results overestimate the energy difference by ≈20 mHa.

Towards a first foundation model for neural network wavefunctions

While the experiments in the previous section demonstrate the ability to pre-train our model and fine-tune it on a new system, the resulting pre-trained models are of little practical use, since they are only pre-trained on a single compound each and can thus not be expected to generalize to chemically different systems. To obtain a more diverse pre-training dataset, we compiled a dataset of 360 distorted geometries, spread across 18 different compounds. The dataset effectively enumerates all chemically plausible molecules with up to 18 electrons containing the elements H, C, N, and O. For details on the data generation see ""Dataset used for pre-training of multi-compound model". We pre-train a base-model for 500,000 steps on this diverse dataset and subsequently evaluate its performance, when computing Potential Energy Surfaces. We evaluate its error against CCSD(T) (extrapolated to the complete basis set limit) both for compounds that were in the pre-training dataset (with different geometries), as well as for new, larger, out-of-distribution compounds which were not present in the pre-training dataset. We compare the results against a baseline model, which uses the same architecture, but is trained from scratch. Instead of re-using the pre-trained weights, this baseline initializes its weights using the default method of supervised HF-pre-training for each specific molecule[2].

Figure 5 shows that fine-tuning our pre-trained model yields substantially lower energies than the usual optimization from a HF-pre-trained model. For example, for new large compounds, it only takes 1k fine-tuning steps of the pre-trained model, to reach the same accuracy as CCSD(T) with a 3Z basis set. The non-pretrained model has a 60x higher energy error after 1k optimization steps, and requires 20x more steps to reach this accuracy. As expected, the gains from pre-training diminish for long subsequent optimization, but after 32k optimization steps, the pre-trained model still demonstrates 3x lower energy errors than the model being trained from scratch.

To assess the accuracy of our method for relative energies, we use the pre-trained model to compute a potential energy surface of a carbon dimer. Figure 6 compares our energies (with and and without fine-tuning) against other state-of-the art conventional and deep-learning-based methods. We compare against CCSD(T) extrapolated to the complete basis-set limit, an FCI-QMC study of

https://doi.org/10.1038/s41467-023-44216-9

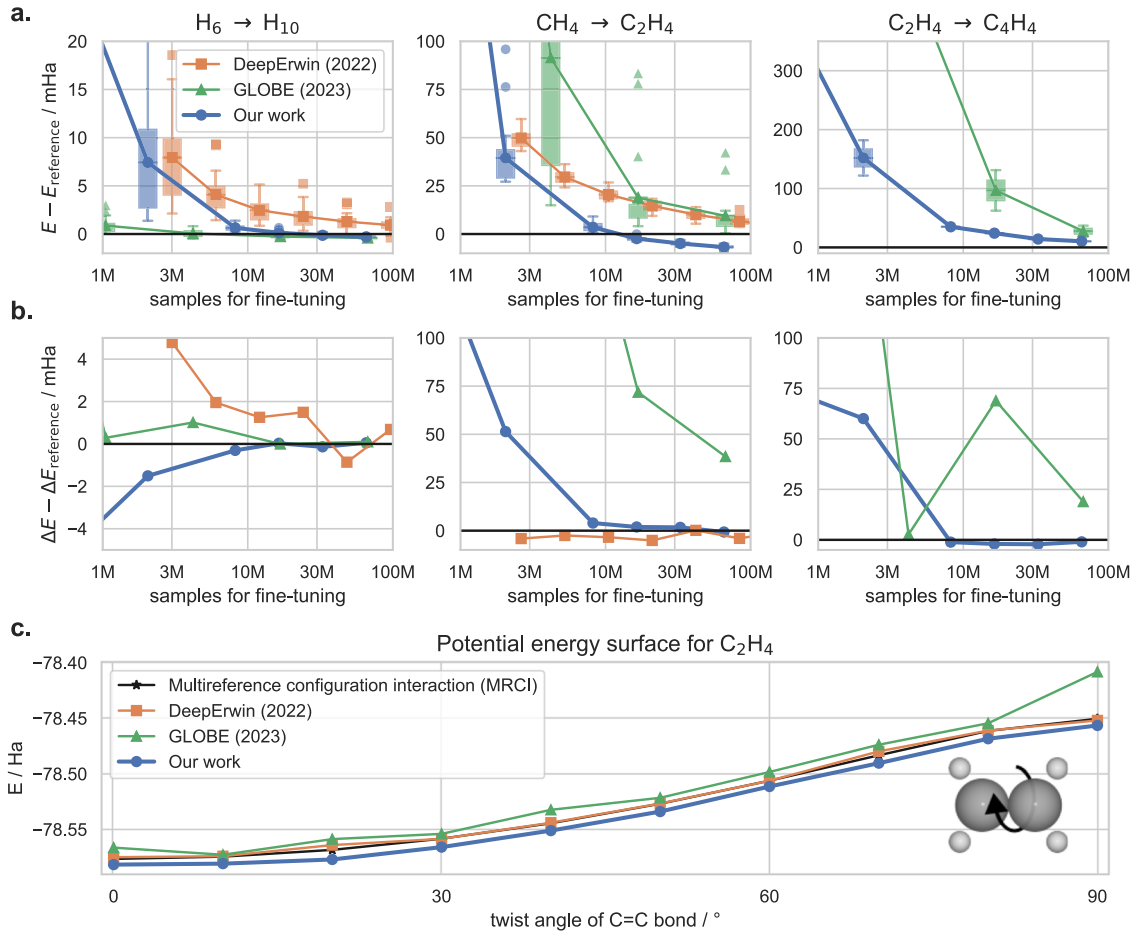

Fig. 4 | Accuracy when pre-training the model on small compounds and reusing it for larger compounds. Boxplots show the 25-75th percentile of energy deviations, connecting lines show mean energy deviations, whiskers span the non-outlier range (1.5 interquartile ranges above and below the boxes), energy deviations beyond the whiskers are plotted individually. **a** Mean energy vs reference energy E_{ref}, averaged across all geometries of the test set. **b** Deviation of relative energies. **c** Final Potential Energy Surface (PES) for the Ethene molecule for each method. For the H-chains and C_2H_4 we use MRCI results from[13] as reference energy, for Cyclobutadiene we use FermiNet results from[16] as reference.

the C_2-dimer by Booth et al.[32] and PsiFormer[4], the currently most accurate deep-learning based ansatz for absolute energies. We find that our approach without any fine-tuning steps correctly identifies the energy minimum at $d = 2.35$ bohr and even yields equilibrium energies that are lower than FCIQMC (cf. Fig. 6a). While the carbon dimer itself is not part of the pre-training dataset, several molecules with C = C bonds are, which explains the relatively high accuracy in this regime. When stretching the bond, our zero-shot energies overestimate the resulting energy by roughly 250mHa (cf. Fig. 6b), clearly highlighting this failure case in a regime of lacking pre-training data. However, after just 1k fine-tuning steps of our pre-trained base-model, we obtain the qualitatively correct PES. In particular in the electronically most challenging regime around $d = 3$ bohr, FCIQMC and CCSD(T) both systematically overestimate the relative energy by 20-30 mHa compared to PsiFormer, wheres our method only overestimates the energy by ca. 10 mHa (cf. Fig. 6c). We note that our approach uses only 1k optimization steps per geometry, compared to 100k (and twice the batch size) for PsiFormer, thus requiring GPU-hours instead of GPU-days.

Scaling behavior

In many domains, increasing the amount of pre-training, has led to substantially better results, even without qualitative changes to the architecture[33]. To investigate the scalability of our approach, we vary the three key choices, along which one could increase the scale of pre-training: The size of the wavefunction model, the number of compounds and geometries present in the pre-training-dataset, and the number of pre-training steps. Starting from a large model, trained on 18x20 geometries, for 256k pre-training steps, we independently vary each parameter. We test 3 different architectures sizes, with decreasing layer width and depth for the networks $\mathbf{f}_\theta$, g_θ, and $\mathbf{GCN}_\theta$ (Computational settings). We test 3 different training sets, with decreasing number of compounds in the training set, with 20 geometries each (Dataset used for pre-training of foundationmodel). Finally, we evaluate model-checkpoints at different amounts of pre-training, ranging from 64k steps to 512k steps. Figure 7 depicts the accuracy obtained by subsequently fine-tuning the resulting model for just 4000 steps on the evaluation set. In each case, increasing the scale of pre-training clearly improves evaluation results—both for the small in-distribution compounds, as well as the larger out-of-distribution compounds. We find a strong dependence of the accuracy on the model size and number of compounds in the pre-training dataset, and a weaker dependency on the number of pre-training steps. While our computational resources, currently prohibit us from training at larger scale, the results indicate that our approach may already be sufficient to train an accurate multi-compound, multi-geometry foundation model for wavefunctions.

Discussion

This work presents an ansatz for deep-learning-based VMC, which can in principle be applied to molecules of arbitrary size. We

https://doi.org/10.1038/s41467-023-44216-9

demonstrate the favorable properties of our ansatz, such as extensivity, zero-shot prediction of wavefunctions for similar molecules (Size consistency of the ansatz), invariance to the phase of orbitals (Equivariance with respect to HF-phase) and fast fine-tuning for larger, new molecules (Transfer to larger, chemically similar compounds). Most importantly, "Towards a first foundation model for neuralnetwork wavefunctions" is, to our knowledge, the first successful demonstration of a wavefunction, which is transferable across compounds and has successfully been trained on a diverse dataset of compounds and geometries. We demonstrate that the dominating deep-learning paradigm of the last years—pre-training on large data and fine-tuning on specific problems—can also be applied to the difficult problem of wavefunctions. While previous attempts[13,26] have failed to obtain high-accuracy energies from pre-trained neural network wavefunctions, we find that our approach yields accurate energies and does so at a fraction of the cost needed without pre-training. A typical inference run (batch-size 2048, one compute node with 2 GPUs) for a molecule with 3 heavy atoms takes ~30 min for zero-shot evaluation, or 1.5h for 1k fine-tuning steps and subsequent evaluation. A CCSD(T)-4Z calculation on a compute-node with 128 CPUs took ~30 min. Given that the per-iteration cost of DL-VMC scales as $\mathcal{O}(n_{\mathrm{el}}{}^4)$ vs. the $\mathcal{O}(n_{\mathrm{el}}{}^7)$ cost of CCSD(T), we expect our model to become competitive once pre-trained and applied to sufficiently large molecules. We furthermore demonstrate in "Scaling behavior" that results can be improved systematically by scaling up any aspect of the pre-training: Model size, data-size, or pre-training-steps.

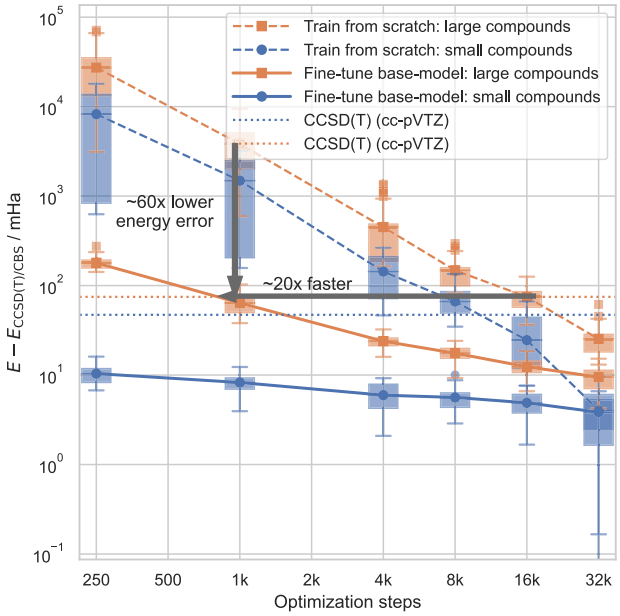

Fig. 5 | Fine-tuning of variationally pre-trained base-model (solid lines) vs. training a model from scratch (dashed lines) for 70 different geometries. Boxplots show the 25-75th percentile of energy deviations, connecting lines show mean energy deviations, whiskers span the non-outlier range (1.5 interquartile ranges above and below the boxes), energy deviations beyond the whiskers are plotted individually. Small compounds are in-distribution, with geometries similar to geometries in pre-training dataset. Larger compounds are out-of-distribution and are not present in the pre-training dataset.

Despite these promising results, there are many open questions and limitations which should be addressed in future work. First, we find that our ansatz currently does not fully match the accuracy of state-of-the-art single-geometry DL-VMC ansätze. While our approach consistently outperforms conventional variational methods such as MRCI or CCSD at finite basis set, larger, computationally more expensive DL-VMC models can reach even lower energies. For example, PsiFormer optimized for 100k steps on the carbon dimer, reaches ≈20 mHa lower absolute energies than our approach fine-tuned for 1k steps. Exchanging our message-passing-based electron-embedding, with recent attention based approaches[4] should lead to higher accuracy. Furthermore we have made several deliberate design choices, which each trade-off expressivity (and thus potentially accuracy) for computational cost: We do not exchange information across orbitals and we base our orbitals on computationally cheap HF-calculations. Including attention or message passing across orbitals (e.g. similar to ref. 26), and substituting HF for a trainable, deep-learning-based model should further increase expressivity. While we currently use HF-orbitals due to their widespread use and low computational cost, our method does not rely on a specific orbital descriptor. We could substitute HF for a separate model such as PhisNet[20] or SchnOrb[34] to compute orbital descriptors $\mathbf{c}_{lk}$, leading to a fully end-to-end machine-learned wavefunction. Second, while we include useful physical priors such as locality, we do not yet currently use the invariance of the Hamiltonian with respect to rotations, inversions or spin-flip. E3-equivariant networks have been highly successful for neural network force-fields, but have not yet been applied to

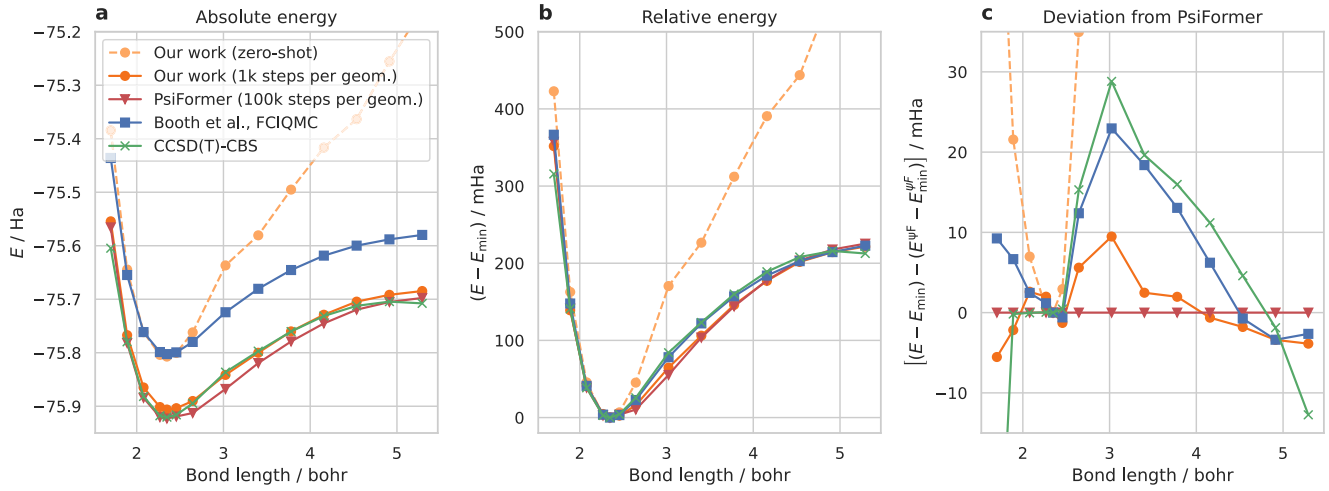

Fig. 6 | Potential energy surface of C_2. a Absolute energies, (**b**) Energies of each method relative to the energy minimum at $d = 2.35$ bohr, (**c**) Deviation of relative energies from the relative energies obtained by PsiFormer (ψF).

https://doi.org/10.1038/s41467-023-44216-9

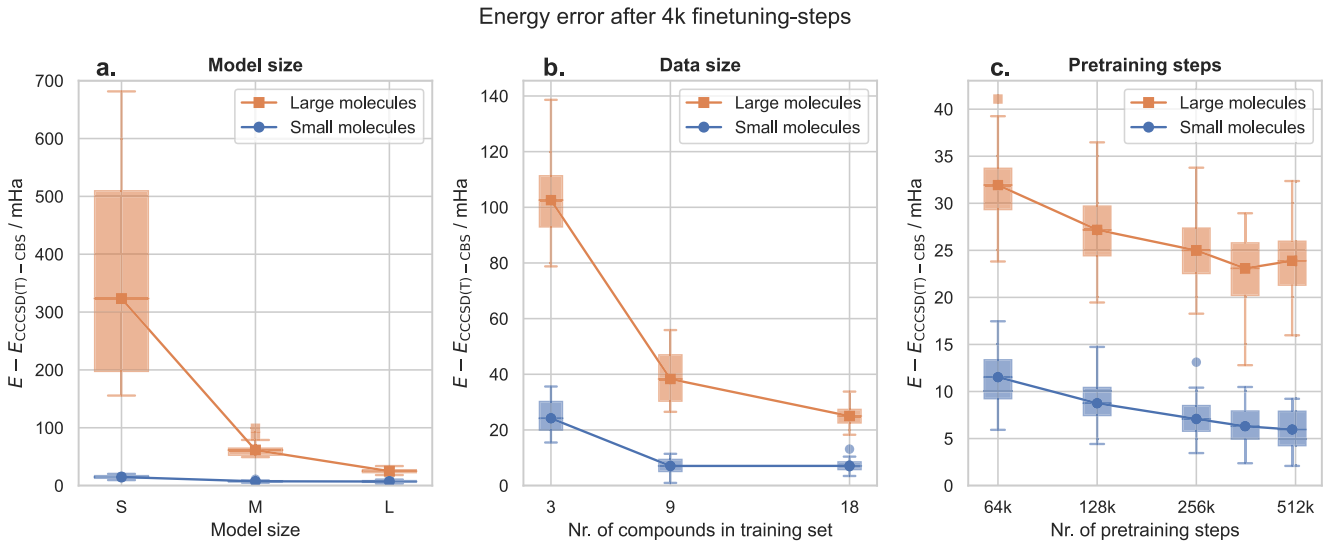

Fig. 7 | Error when fine-tuning the pre-trained model for 4000 steps on small in-distribution geometries and larger out-of-distribution geometries. Boxplots show the 25-75th percentile of energy deviations, connecting lines show mean energy deviations, whiskers span the non-outlier range (1.5 interquartile ranges above and below the boxes), energy deviations beyond the whiskers are plotted individually. **a** The energy error for increasing the model size of the transferable atomic orbitals. The small model uses no hidden layers and no graph convolutional network. The medium-sized model uses one hidden layer of width 64 and 128 for g_θ and for f_θ respectively, and one iteration of the graph convolutional network. The large model uses two iterations of the graph convolutional network, f_θ and g_θ with hidden dimension 256 and 128. **b** The energy error when using a pre-trained model on a dataset with either 3, 9 or 18 compounds. **c** The energy error with increasing number of pretraining steps.

wavefunctions due to the hitherto unsolved problem of symmetry breaking[15]. Using HF-orbitals as symmetry breakers, could open a direct avenue towards E3-equivariant neural network wavefunctions. Third, while we use locality of our orbitals as a useful prior, we do not yet use it to reduce computational cost. By enforcing sparsity of the localized HF-coefficients, one could limit the evaluation of orbitals to a few participating atoms, instead of all atoms in the molecule. While the concurrent GLOBE approach enforces its orbitals to be localized at a single position, our approach naturally lends itself to force localization at a given number of atoms, allowing for a deliberate trade-off of accuracy vs. computational cost. Lastly, we observe that our method performs substantially better, when dedicating more computational resources to the pre-training, which makes it likely that future work will be able to scale up our approach. To facilitate this effort we open source our code, dataset as well as model parameters.

Methods

Variational Monte Carlo

Considering the Born-Oppenheimer approximation, a molecule with n_{el} electrons and N_{atoms} nuclei can be described by the time-independent Schrödinger equation

$$\hat{H}\psi = E\psi \tag{6}$$

with the Hamiltonian

$$\hat{H} = -\frac{1}{2}\sum_i \nabla^2_{\mathbf{r}_i} + \sum_{i>j}\frac{1}{|\mathbf{r}_i - \mathbf{r}_j|} + \sum_{I>J}\frac{Z_I Z_J}{|\mathbf{R}_I - \mathbf{R}_J|} - \sum_{i,I}\frac{Z_I}{|\mathbf{r}_i - \mathbf{R}_I|} \tag{7}$$

By $\mathbf{r} = (\mathbf{r}_1, \ldots, \mathbf{r}_{n_\uparrow}, \ldots, \mathbf{r}_{n_{\mathrm{el}}}) \in \mathbb{R}^{3\times n_{\mathrm{el}}}$ we denote the set of electron positions divided into $n_\uparrow$ spin-up and $n_\downarrow = n_{\mathrm{el}} - n_\uparrow$ spin-down electrons. The solution to the electronic Schrödinger equation ψ needs to fulfill the anti-symmetry property, i.e. $\psi(\mathcal{P}\mathbf{r}) = -\psi(\mathbf{r})$ for any permutation $\mathcal{P}$ of two electrons of the same spin. Finding the groundstate wavefunction of a system corresponds to finding the solution to Eq. (6), with the lowest eigenvalue E_0. Using the Rayleigh-Ritz principle, an approximate solution can be found through minimization of the loss

$$\mathcal{L}(\psi_\theta) = \mathbb{E}_{\mathbf{r}\sim\psi^2_\theta(\mathbf{r})}\left[\frac{(\hat{H}\psi_\theta)(\mathbf{r})}{\psi_\theta(\mathbf{r})}\right] \geq E_0, \tag{8}$$

using a parameterized trial wavefunction ψ_θ. The expectation value in Eq. (8) is computed by drawing samples $\mathbf{r}$ from the unnormalized probability distribution $\psi^2_\theta(\mathbf{r})$ using Markov Chain Monte Carlo (MCMC). The application of the Hamiltonian to the wavefunction can be computed using automatic differentiation and the loss is minimized using gradient based minimization. A full calculation typically consists of three steps:

(i) **Supervised HF-pre-training**: Minimization of the difference between the neural network ansatz and a reference wavefunction (e.g. a Hartree-Fock calculation) $||\psi_\theta - \psi^{\mathrm{HF}}||$. This is the only part of the procedure which requires reference data, and ensures that the initial wavefunction roughly resembles the true groundstate. While this step is in principle not required, it substantially improves the stability of the subsequent variational optimization.

(ii) **Variational optimization**: Minimization of the energy (Eq. (8)) by drawing samples from the wavefunction using MCMC, and optimizing the parameters θ of the ansatz using gradient based optimization.

(iii) **Evaluation**: Evaluation of the energy by evaluating Eq. (8) without updating the parameters θ, to obtain unbiased estimates of the energy.

To obtain a single wavefunction for a dataset of multiple geometries or compounds, only minimal changes are required. During supervised and variational optimization, for each gradient step we pick one geometry from the dataset. We pick geometries either in a

round-robin fashion, or based on the last computed energy variance for that geometry. We run the Metropolis Hastings algorithm[35] for that geometry to draw electron positions $\mathbf{r}$ and then evaluate energies and gradients. For each geometry we keep a distinct set of electron samples $\mathbf{r}$.

Obtaining orbital descriptors from Hartree-Fock

As discussed in "A multi-compound wavefunction ansatz", our ansatz effectively maps uncorrelated, low-accuracy Hartree-Fock orbitals, to correlated, high-accuracy neural network orbitals. The first step in this approach is to obtain orbital descriptors $\mathbf{c}_k$ for each orbital k, based on a Hartree-Fock calculation.

The Hartree-Fock method uses a single determinant as ansatz, composed of single-particle orbitals ϕ_k^{HF}:

$$\psi^{\mathrm{HF}}(\mathbf{r}_1,\ldots,\mathbf{r}_{n_{\mathrm{el}}}) = \det\left[\Phi_{ik}^{\mathrm{HF}}\right]_{i,k=1\ldots n_{\mathrm{el}}} \tag{9}$$

$$\Phi_{ik}^{\mathrm{HF}} := \phi_k^{\mathrm{HF}}(\mathbf{r}_i) \tag{10}$$

For molecules, these orbitals are typically expanded in atom-centered basis-functions $\mu(\mathbf{r})$, with N_{basis} functions centered on each atom I:

$$\phi_k^{\mathrm{HF}}(\mathbf{r}) = \sum_{I=1}^{N_{\mathrm{atoms}}}\sum_{b=1}^{N_{\mathrm{basis}}} \alpha_{Ik,b}\,\mu_b(\mathbf{r}-\mathbf{R}_I), \tag{11}$$

The coefficients $\boldsymbol{\alpha}_{Ik} \in \mathbb{R}^{N_{\mathrm{basis}}}$ and the corresponding orbitals $\phi_k^{\mathrm{HF}}(\mathbf{r})$ are obtained as solutions of an eigenvalue problem and are typically delocalized, i.e. they have non-zero contributions from many atoms. However, since $\det[U\Phi] = \det[U]\det[\Phi]$, the wavefunction is invariant under linear combination of orbitals by a matrix U with $\det[U]=1$. One can thus choose localized orbital expansion coefficients

$$\tilde{\alpha}_{Ik,b} = \sum_{k'=1}^{N_{\mathrm{orb}}} \alpha_{Ik,b} U_{kk'} \tag{12}$$

corresponding to orbitals which are maximally localized according to some metric. We stress that such a transformation from canonical to localized orbitals is lossless: The localized orbitals represent exactly the same wavefunction as the canonical orbitals and thus the procedure involves no approximation. We localize orbitals purely to simplify the learning problem for the subsequent trainable functions $\mathbf{f}$, and g, which map orbital descriptors to backflows and exponents. If the orbitals for typical molecules can be composed of recurring local motives (which empirically holds true), this substantially simplifies the generalization of $\mathbf{f}$ and g to larger molecules, since their inputs will mostly consist of orbital coefficients already seen in smaller molecules. Several different metrics and corresponding localization schemes, such as Foster-Boys[36] or Pipek-Mezey[37], have been proposed to find the optimal transformation matrix U and are easily available as computationally cheap post-processing options in quantum chemistry codes. We use the Foster-Boys method as implemented in pySCF[38].

Due to the fundamentally local nature of atom-wise orbital coefficients $\tilde{\boldsymbol{\alpha}}_{Ik}$, which can be insufficient to distinguish orbitals, we use a fully connected graph convolutional neural network (GCN) to add context about the surrounding atoms. We interpret each atom as a node (with node features $\tilde{\boldsymbol{\alpha}}_{Ik}$) and use the set of all 3D inter-atomic distance vectors $\{\mathbf{R}_{IJ}\}$ as edge features:

$$\mathbf{c}_{Ik} = \mathbf{GCN}_{\theta,I}\left(\{\tilde{\boldsymbol{\alpha}}_{Jk}\}_{J=1\ldots N_{\mathrm{atoms}}}, \{\mathbf{R}_{JJ'}\}\right),$$
$$J,J' = 1\ldots N_{\mathrm{atoms}}$$

We embed the edge features using a Kronecker product of Gaussian basis functions (of means $\boldsymbol{\mu} \in \mathbb{R}^{D_{\mathrm{edge}}}$ and widths $\boldsymbol{\sigma} \in \mathbb{R}^{D_{\mathrm{edge}}}$) of the inter-atomic distance R_{IJ} and the concatenation of the 3D-distance vector with the constant 1. The embedded edge features are then mapped to a high-dimensional feature space with a multi-layer perceptron (MLP):

$$\tilde{\mathbf{e}}_{IJ} = \exp\left(-\frac{(\mathbf{R}_{IJ}-\boldsymbol{\mu})^2}{2\boldsymbol{\sigma}^2}\right) \otimes \left[1|\mathbf{R}_{IJ}\right] \tag{13}$$

$$\mathbf{e}_{IJ} = \mathrm{MLP}(\tilde{\mathbf{e}}_{IJ}) \tag{14}$$

$$\begin{aligned} &\mathbf{c}_{Ik}^0 = \tilde{\boldsymbol{\alpha}}_{Ik} \\ &\tilde{\mathbf{e}}_{IJ} \in \mathbb{R}^{4D_{\mathrm{edge}}}, \quad \mathbf{c}_{Ik}^0 \in \mathbb{R}^{N_{\mathrm{basis}}} \end{aligned} \tag{15}$$

Each layer l of the GCN consist of the following update rules

$$\mathbf{u}_{Ik}^l = \sum_J \mathbf{c}_{Jk}^l \odot \left(\mathbf{W}_{\mathrm{e}}{}^l \mathbf{e}_{IJ}\right), \tag{16}$$

$$\mathbf{c}_{Ik}^{l+1} = \sigma\left(\mathbf{W}_{\mathrm{c}}{}^l \mathbf{c}_{Ik}^l + \mathbf{W}_{\mathrm{u}}{}^l \mathbf{u}_{Ik}^l\right), \tag{17}$$

with trainable weight matrices $\mathbf{W}_{\mathrm{e}}{}^l$, $\mathbf{W}_{\mathrm{c}}{}^l$, $\mathbf{W}_{\mathrm{u}}{}^l$ and the SiLU activation function σ[39]. After L iterations we use the final outputs as orbitals features:

$$\mathbf{c}_{Ik} := \mathbf{c}_{Ik}^L \tag{18}$$

Mapping orbital descriptors to wavefunctions

To obtain entries Φ_{ik} of the Slater determinant, we combine a high-dimensional electron embedding $\mathbf{h}_i$ with a function of the orbital descriptor $\mathbf{c}_{Ik}$:

$$\mathbf{h}_i = \mathbf{h}_\theta(\mathbf{r}_i, \{\mathbf{r}_\uparrow\}, \{\mathbf{r}_\downarrow\}, \{(\mathbf{R},\mathbf{Z})\}) \tag{19}$$

$$\varphi_\theta^d(\mathbf{r}_i,\mathbf{R}_I,\mathbf{c}_{Ik}) = \exp\left(-|\mathbf{r}_i-\mathbf{R}_I|\, g_\theta^{\mathrm{e},d}(\mathbf{c}_{Ik})\right) \tag{20}$$

$$\Phi_{ik}^d = \sum_{I=1}^{N_{\mathrm{atoms}}} \varphi_\theta^d(\mathbf{r}_i,\mathbf{R}_I,\mathbf{c}_{Ik}) \left\langle \mathbf{f}_\theta^{\mathrm{o},d}(\mathbf{c}_{Ik}), \mathbf{h}_i \right\rangle \tag{21}$$

The functions $\mathbf{GCN}_\theta^{\mathrm{o}}$, $\mathbf{f}_\theta^{\mathrm{o}}$, and g_θ^{e} are trainable functions, which are enforced to be odd and even with respect to change in sign of their argument $\mathbf{c}$:

$$\begin{aligned} &\text{Even } g_\theta^{\mathrm{e}}: \\ &g_\theta^{\mathrm{e}}(\mathbf{c}) := g_\theta(\mathbf{c}) + g_\theta(-\mathbf{c}) \end{aligned} \tag{22}$$

$$\begin{aligned} &\text{Odd } \mathbf{f}_\theta^{\mathrm{o}}: \\ &\mathbf{f}_\theta^{\mathrm{o}}(\mathbf{c}) := \mathbf{f}_\theta(\mathbf{c}) - \mathbf{f}_\theta(-\mathbf{c}) \end{aligned} \tag{23}$$

$$\begin{aligned} &\text{Odd } \mathbf{GCN}_\theta^{\mathrm{o}}: \\ &\mathbf{GCN}_\theta^{\mathrm{o}}(\boldsymbol{\alpha},\mathbf{R}) := \mathbf{GCN}_\theta(\boldsymbol{\alpha},\mathbf{R}) - \mathbf{GCN}_\theta(-\boldsymbol{\alpha},\mathbf{R}) \end{aligned} \tag{24}$$

To obtain electron embeddings $\mathbf{h}_i$ we use the message-passing architecture outlined in[5], which is invariant with respect to permutation of electrons of the same spin, or the permutation of ions. Note that during training, all samples in a batch come from the same geometry, and thus have the same values for $\mathbf{R}$, $\mathbf{Z}$, and $\tilde{\boldsymbol{\alpha}}$. While the embedding

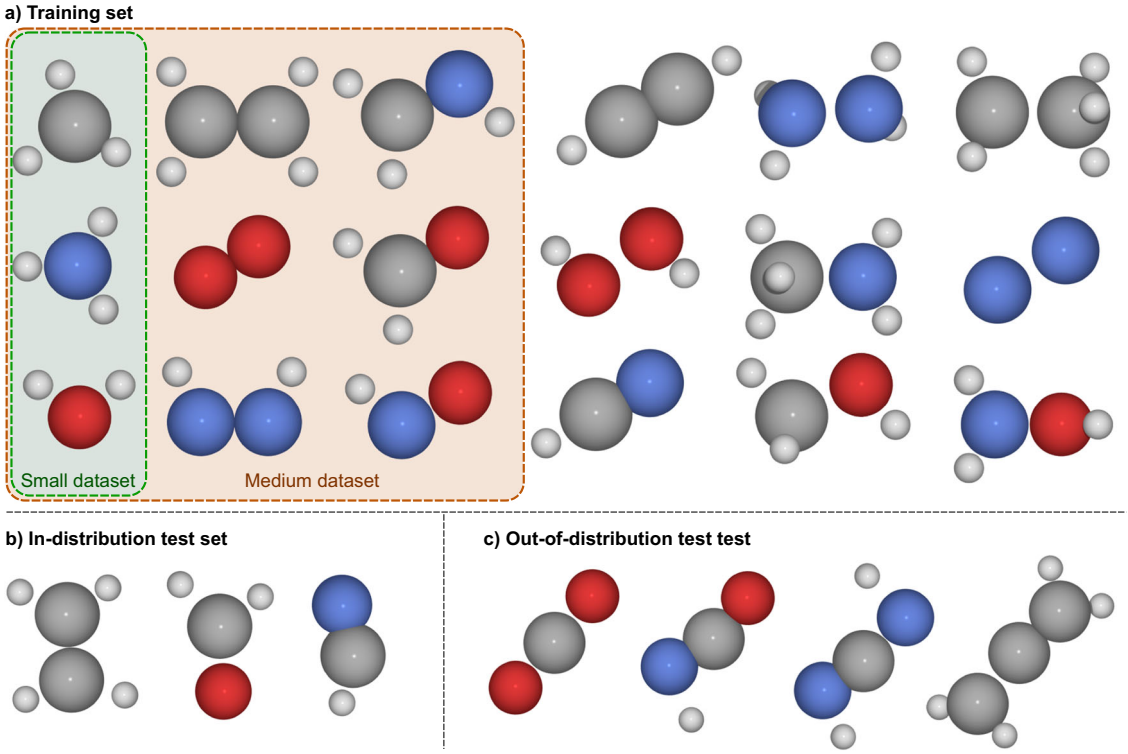

Fig. 8 | Compounds used for pre-training and evaluation of our model. Atom colors follow the usual convention of H = white, C = gray, N = blue, O = red. **a** The full training set, containing 18 compounds, each with 20 randomly distorted geometries. The small and medium sized training sets are subsets of this full training set, containing 3 and 9 compounds respectively. **b** The in-distribution test set consists of 3 compounds with 10 distorted geometries each. **c** The out-of-distribution test set consists of 4 compounds with 10 distortions each.

network $\mathbf{h}_\theta^{\text{embed}}$, needs to be re-evaluated for every sample, the networks $\mathbf{GCN}_\theta$, $\mathbf{f}_\theta$, and g_θ only need to be evaluated once per batch, substantially reducing their impact on computational cost.

Dataset used for pre-training of multi-compound model

We use RDKit[40] to generate all valid SMILES of molecules containing 1-3 atoms of the elements C, N, O. For each bond between atoms we allow single, double, and triple bonds. After saturating the molecules with Hydrogen, we perform force-field based geometry relaxation using RDKit. We obtain 18 compounds with 10-18 electrons, which we use for pre-training (cf. Fig. 8) and 35 compounds with 20-24 electrons, of which we use some for evaluation. Contrary to other datasets of small molecules such as GDB-7, our dataset also includes compounds which do not contain Carbon, such as the nitrogen dimer N_2 or hydrogen peroxide H_2O_2. To obtain a more diverse dataset we perturb each equilibrium geometry by applying Gaussian noise to the 3D coordinates. Since this can generate nonphysical geometries, we keep only geometries in which the perturbed inter-atomic distances are between 90–140% of the unperturbed distances.

Reference energies

Reference energies for H_2O in Fig. 3 were computed using DL-VMC for 100,000 steps[5]. Reference energies for H_{10} and C_2H_4 in Fig. 4 were computed using MRCI-F12(Q)[13]. Reference energies for C_4H_4 in Fig. 4 were computed using DL-VMC[16]. To compute reference energies for our multi-compound dastaset used in Fig. 5 and Fig. 7, we used pySCF[38] to perform CCSD(T) calculations using the cc-pCVXZ basis sets. We computed Hartree-Fock energies E_X^{HF} using basis-sets of valence $X = \{2, 3, 4\}$ and CCSD(T) energies $E_X^{\text{CCSD(T)}}$ using valence $X = \{2, 3\}$. To extrapolate to the complete-basis-set-limit, we followed[2] and fitted the following functions with free parameters $E_{\text{CBS}}^{\text{HF}}, E_{\text{CBS}}^{\text{corr}}, a, b, c$:

$$E_X^{\text{HF}} = E_{\text{CBS}}^{\text{HF}} + ae^{-bX}$$
$$E_X^{\text{corr}} := E_X^{\text{HF}} - E_X^{\text{CCSD(T)}} = E_{\text{CBS}}^{\text{corr}} + cX^{-3}$$
$$E_{\text{CBS}}^{\text{CCSD(T)}} = E_{\text{CBS}}^{\text{HF}} + E_{\text{CBS}}^{\text{corr}}$$

We note that neither CCSD itself, nor the perturbative (T) treatment, nor the CBS extrapolation are variational methods. The computed reference energies are therefore not variational and may underestimate the true groundstate energy.

Computational settings

For a more detailed summary and explanation of the high-dimensional embedding structure we refer to the original work[5]. In all experiments we relied on the second order optimizer K-FAC[41,42]. Key hyperparameters used in this work are summarized in Table 1. For the base model in "Towards a first foundation model for neural network wavefunctions" we increased the initial damping by 10x and ramped it down to 1×10^{-3} with an inverse scheduler. All runs reusing pre-trained weights, offset the learning rate scheduler by $o = 32,000$ steps, i.e. $\text{lr}(t) = \text{lr}_0(1+(t+o)/6000)^{-1}$. This leads to a 5x lower initial learning rate. All pre-training runs in "Transfer to larger, chemically similar compounds" used 64,000 optimization steps. The base model in "Towards a first foundation model for neural network wavefunctions" used 512,000 optimization steps due to the larger and more diverse training corpus.

The small- and medium-sized model for our ablation study in Fig. 7 differ from the large model by the number of hidden layers for $\mathbf{f}_\theta$ and g_θ, the number of neurons per layer, and the number of iterations of the $\mathbf{GCN}_\theta$: The small model uses no hidden layers and

Table 1 | Hyperparameter settings used in this work

HF-pre-training	Pre-training basis set	6-31G + p-functions for H
	Pre-training steps per geometry	100-500
Embedding	Hidden dimension of $\mathbf{h}_i$	256
	Dimension of SchNet convolution	32
	Nº iterations embedding	4
	Activation function	tanh
Transferable atomic orbitals	Nº determinants N_{det}	4
	Basis set	6-31G + p-functions for H
	Nº hidden layers $\mathbf{f}_\theta$	2
	Hidden dimension of $\mathbf{f}_\theta$	256
	Nº hidden layers g_θ	2
	Nº hidden dimension g_θ	128
	Nº iterations GCN	2
	Nº Gaussian basis functions	16
	Hidden edge embedding dimension D_{edge}	32
	Hidden node embedding dim.	16
	Activation function	SiLU
Markov Chain Monte Carlo	Nº walkers	2048
	Nº decorrelation steps	20
	Target acceptance prob.	50%
Variational pre-training	Optimizer	KFAC
	Damping	1×10^{-3}
	Norm constraint	3×10^{-3}
	Batch size	2048
	Initial learning rate lr_0	0.1
	Learning rate decay	$\mathrm{lr}(t)=\mathrm{lr}_0(1+t/6000)^{-1}$
	Optimization steps	64,000–512,000
Changes for fine-tuning	Learning rate decay	$\mathrm{lr}(t)=\mathrm{lr}_0(7+t/6000)^{-1}$
	Optimization steps	0–32,000

no graph convolutional network. The medium-sized model uses one hidden layer of width 64 for g_θ and 128 for $\mathbf{f}_\theta$, and one iteration of the graph convolutional network. The small, medium and large models respectively have 0.8 mio, 1.2 mio. and 2.0 mio parameters.

Data availability

All geometry- and energy-data is available on GitHub under https://github.com/mdsunivie/deeperwin. Model weights are available on figshare under https://doi.org/10.6084/m9.figshare.23585358.v1[43]. Source data are provided with this paper.

Code availability

All code is available on GitHub under https://github.com/mdsunivie/deeperwin and Zenodo (https://doi.org/10.5281/zenodo.10081846)[44].

References

1. Hermann, J., Schätzle, Z. & Noé, F. Deep-neural-network solution of the electronic Schrödinger equation. *Nat. Chem.* **12**, 891–897 (2020).
2. Pfau, D., Spencer, J. S., Matthews, A. G. D. G. & Foulkes, W. M. C. Ab initio solution of the many-electron Schrödinger equation with deep neural networks. *Phys. Rev. Res.* **2**, 033429 (2020).
3. Spencer, J. S., Pfau, D., Botev, A. & Foulkes, W. M. C. Better, faster fermionic neural networks. *arXiv* https://doi.org/10.48550/arXiv.2011.07125 (2020).
4. von Glehn, I., Spencer, J. S. & Pfau, D. A self-attention ansatz for ab-initio quantum chemistry. In *The Eleventh International Conference on Learning Representations* 10853–10892 (ICLR, 2023).
5. Gerard, L., Scherbela, M., Marquetand, P. & Grohs, P. Gold-standard solutions to the Schrödinger equation using deep learning: How much physics do we need? In *Advances in Neural Information Processing Systems* 10282–10294 (NeurIPS, 2022).
6. Towards the ground state of molecules via diffusion monte carlo on neural networks. *Nat. Commun.***14**, 1860 (2023).
7. Wilson, M., Gao, N., Wudarski, F., Rieffel, E. & Tubman, N. M. Simulations of state-of-the-art fermionic neural network wave functions with diffusion Monte Carlo. *arXiv* https://doi.org/10.48550/arXiv.2011.07125 (2021).
8. Cassella, G. et al. Discovering quantum phase transitions with fermionic neural networks. *Phys. Rev. Lett.* **130**, 036401 (2023).
9. Wilson, M. et al. Neural network ansatz for periodic wave functions and the homogeneous electron gas. *Phys. Rev. B* **107**, 235139 (2023).
10. Li, X., Li, Z. & Chen, J. Ab initio calculation of real solids via neural network ansatz. *Nat. Commun.* **13**, 7895 (2022).
11. Han, J., Zhang, L. & E, W. Solving many-electron Schrödinger equation using deep neural networks. *J. Comput. Phys.* **399**, 108929 (2019).
12. Qian, Y., Fu, W., Ren, W. & Chen, J. Interatomic force from neural network based variational quantum Monte Carlo. *J. Chem. Phys.***157**, 164104 (2022).
13. Scherbela, M., Reisenhofer, R., Gerard, L., Marquetand, P. & Grohs, P. Solving the electronic Schrödinger equation for multiple nuclear geometries with weight-sharing deep neural networks. *Nat. Comput. Sci.* **2**, 331–341 (2022).
14. Entwistle, M. T., Schätzle, Z., Erdman, P. A., Hermann, J. & Noé, F. Electronic excited states in deep variational Monte Carlo. *Nat. Commun.* **14**, 274 (2023).
15. Gao, N. & Günnemann, S. Ab-initio potential energy surfaces by pairing GNNs with neural wave functions. In *International Conference on Learning Representations* 10259–10281 (ICLR, 2022).
16. Gao, N. & Günnemann, S. Sampling-free inference for ab-initio potential energy surface networks. In *The Eleventh International Conference on Learning Representations* 10896–10965 (ICLR, 2023).
17. Carleo, G. & Troyer, M. Solving the quantum many-body problem with artificial neural networks. *Science* **355**, 602–606 (2017).
18. Kochkov, D. & Clark, B. K. Variational optimization in the AI era: computational graph states and supervised wave-function optimization. *arXiv* https://doi.org/10.48550/arXiv.2011.07125 (2018).
19. Schütt, K. T., Gastegger, M., Tkatchenko, A., Müller, K.-R. & Maurer, R. J. Unifying machine learning and quantum chemistry with a deep neural network for molecular wavefunctions. *Nat. Commun.* **10**, 5024 (2019).
20. Unke, O. et al. SE(3)-equivariant prediction of molecular wavefunctions and electronic densities. In *Advances in Neural Information Processing Systems* 14434–14447 (NeurIPS, 2021).
21. Batatia, I. et al. The design space of E(3)-equivariant atom-centered interatomic potentials. *arXiv* https://doi.org/10.48550/arXiv.2205.06643 (2022).
22. Brown, T. et al. Language models are few-shot learners. In *Advances in Neural Information Processing Systems*. (eds. Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. & Lin, H.) 1877–1901 (NeurIPS, 2020).

23. Radford, A. et al. Learning transferable visual models from natural language supervision. In Meila, M. & Zhang, T. (eds.) *Proceedings of the 38th International Conference on Machine Learning Research* 8748–8763 (PMLR, 2021).
24. Yuan, L. et al. Florence: A new foundation model for computer vision. *arXiv* https://doi.org/10.48550/arXiv.2011.07125 (2021).
25. Zhang, Y.-H. & Di Ventra, M. Transformer quantum state: a multi-purpose model for quantum many-body problems. *Phys. Rev. B* **107**, 075147 (2023).
26. Gao, N. & Günnemann, S. Generalizing neural wave functions. In Krause, A. et al. (eds.) *Proceedings of the 40th International Conference on Machine Learning*, vol. 202 of *Proceedings of Machine Learning Research* 10708–10726 (PMLR, 2023).
27. Behler, J. & Parrinello, M. Generalized neural-network representation of high-dimensional potential-energy surfaces. *Phys. Rev. Lett.* **98**, 146401 (2007).
28. Bartók, A. P., Payne, M. C., Kondor, R. & Csányi, G. Gaussian approximation potentials: the accuracy of quantum mechanics, without the electrons. *Phys. Rev. Lett.* **104**, 136403 (2010).
29. Motta, M. et al. Towards the solution of the many-electron problem in real materials: equation of state of the hydrogen chain with state-of-the-art many-body methods. *Phys. Rev. X* **7**, 031059 (2017).
30. Westermayr, J. & Marquetand, P. Machine learning for electronically excited states of molecules. *Chem. Rev.* **121**, 9873–9926 (2021).
31. Lyakh, D. I., Musiał, M., Lotrich, V. F. & Bartlett, R. J. Multireference nature of chemistry: The coupled-cluster view. *Chemical Reviews* **112**, 182–243 (2012).
32. Booth, G. H., Cleland, D., Thom, A. J. W. & Alavi, A. Breaking the carbon dimer: The challenges of multiple bond dissociation with full configuration interaction quantum Monte Carlo methods. *J. Chem. Phys.* **135**, 084104 (2011).
33. Hoffmann, J. et al. An empirical analysis of compute-optimal large language model training. In *Advances in Neural Information Processing Systems* (eds. Koyejo, S. et al.) 30016–30030 (NeurIPS, 2022).
34. Gastegger, M., McSloy, A., Luya, M., Schütt, K. T. & Maurer, R. J. A deep neural network for molecular wave functions in quasi-atomic minimal basis representation. *J. Chem. Phys.* **153**, 044123 (2020).
35. Hastings, W. K. Monte Carlo sampling methods using Markov chains and their applications. *Biometrika* **57**, 97–109 (1970).
36. Foster, J. M. & Boys, S. F. Canonical configurational interaction procedure. *Rev. Mod. Phys.* **32**, 300–302 (1960).
37. Pipek, J. & Mezey, P. G. A fast intrinsic localization procedure applicable for ab initio and semiempirical linear combination of atomic orbital wave functions. *J. Chem. Phys.* **90**, 4916–4926 (1989).
38. Sun, Q. et al. Recent developments in the PySCF program package. *J. Chem. Phys.***153**, 024109 (2020).
39. Elfwing, S., Uchibe, E. & Doya, K. Sigmoid-weighted linear units for neural network function approximation in reinforcement learning. *arXiv* https://doi.org/10.48550/arXiv.2011.07125 (2017)
40. Landrum, G. Rdkit: *Open-Source Cheminformatics* https://github.com/rdkit/rdkit (2009).
41. Martens, J. & Grosse, R. Optimizing neural networks with kronecker-factored approximate curvature. In *International Conference on Machine Learning*, 2408–2417 (PMLR, 2015).
42. Botev, A. & Martens, J. *KFAC-JAX* http://github.com/deepmind/kfac-jax (2022).
43. Gerard, L., Scherbela, M. & Grohs, P. *Pre-Trained Neural Wavefunction Checkpoints for the GitHub Codebase DeepErwin* https://figshare.com/articles/online_resource/Pre-trained_neural_wavefunction_checkpoints_for_the_GitHub_codebase_DeepErwin/23585358 (2023).
44. Scherbela, M., Gerard, L.& Grohs, P. *Deeperwin* https://github.com/mdsunivie/deeperwin/blob/master/README.md (2023).

Acknowledgements

We gratefully acknowledge financial support from the following grants: Austrian Science Fund FWF Project I 3403 (P.G.), WWTF-ICT19-041 (L.G.). The computational results have been achieved using the Vienna Scientific Cluster (VSC). The funders had no role in study design, data collection and analysis, decision to publish or preparation of the manuscript. Additionally, we thank Nicholas Gao for providing his results and data, Ruard van Workum for initial work on the python implementation for multi-compound optimization and Jan Hermann for fruitful discussions.

Author contributions

M.S., L.G., and P.G. conceived the overall idea. M.S. conceived and implemented the ansatz, built the dataset and designed the experiments. L.G. gave input on the ansatz and worked on implementation. M.S. and L.G. performed the experiments. M.S. and L.G. wrote the manuscript with input, supervision and funding from P.G.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information The online version contains supplementary material available at https://doi.org/10.1038/s41467-023-44216-9.

Correspondence and requests for materials should be addressed to Philipp Grohs.

Peer review information *Nature Communications* thanks Gero Friesecke and the other, anonymous, reviewers for their contribution to the peer review of this work. A peer review file is available.

Variational Monte Carlo on a Budget – Fine-tuning pre-trained Neural Wavefunctions

Michael Scherbela[*]
University of Vienna
michael.scherbela@univie.ac.at

Leon Gerard[*]
University of Vienna
leon.gerard@univie.ac.at

Philipp Grohs
University of Vienna
philipp.grohs@univie.ac.at

Abstract

Obtaining accurate solutions to the Schrödinger equation is the key challenge in computational quantum chemistry. Deep-learning-based Variational Monte Carlo (DL-VMC) has recently outperformed conventional approaches in terms of accuracy, but only at large computational cost. Whereas in many domains models are trained once and subsequently applied for inference, accurate DL-VMC so far requires a full optimization for every new problem instance, consuming thousands of GPUhs even for small molecules. We instead propose a DL-VMC model which has been pre-trained using self-supervised wavefunction optimization on a large and chemically diverse set of molecules. Applying this model to new molecules without any optimization, yields wavefunctions and absolute energies that outperform established methods such as CCSD(T)-2Z. To obtain accurate relative energies, only few fine-tuning steps of this base model are required. We accomplish this with a fully end-to-end machine-learned model, consisting of an improved geometry embedding architecture and an existing SE(3)-equivariant model to represent molecular orbitals. Combining this architecture with continuous sampling of geometries, we improve zero-shot accuracy by two orders of magnitude compared to the state of the art. We extensively evaluate the accuracy, scalability and limitations of our base model on a wide variety of test systems.

1 Introduction

Solving the Schrödinger equation is of utmost importance for the prediction of quantum chemical properties in chemistry. The time-independent Schrödinger equation in the Born-Oppenheimer approximation [1] for a molecule with N_{nuc} nuclei and n_{el} electrons is an eigenvalue problem with Hamiltonian H:

$$H\psi = E\psi, \qquad H = -\frac{1}{2}\sum_i \nabla^2_{\boldsymbol{r}_i} + \sum_{i>j}\frac{1}{r_{ij}} + \sum_{I>J}\frac{Z_I Z_J}{R_{IJ}} - \sum_{i,I}\frac{Z_I}{r_{iI}}. \tag{1}$$

By $\boldsymbol{R} = (\boldsymbol{R}_1, \dots, \boldsymbol{R}_{N_{\text{nuc}}}) \in \mathbb{R}^{N_{\text{nuc}}\times 3}$ and $\boldsymbol{Z} = (Z_1, \dots, Z_{N_{\text{nuc}}}) \in \mathbb{N}^{N_{\text{nuc}}}$ we denote the nuclear positions and charges. The electron positions are denoted by $\boldsymbol{r} = (\boldsymbol{r_1}, \dots, \boldsymbol{r}_{n_\uparrow}, \dots, \boldsymbol{r}_{n_{\text{el}}}) \in \mathbb{R}^{n_{\text{el}}\times 3}$ with $n_\uparrow$ spin-up electrons and $n_\downarrow$ spin-down electrons. The inter-particle difference and distance vectors are written as $\boldsymbol{R}_{IJ} = \boldsymbol{R}_I - \boldsymbol{R}_J$, $R_{IJ} = |\boldsymbol{R}_{IJ}|$, $\boldsymbol{r}_{iI} = \boldsymbol{r}_i - \boldsymbol{R}_I$, $r_{iI} = |\boldsymbol{r}_{iI}|$, $\boldsymbol{r}_{ij} = \boldsymbol{r}_i - \boldsymbol{r}_j$ and $r_{ij} = |\boldsymbol{r}_{ij}|$ with $I, J = 1, \dots, N_{\text{nuc}}$ and $i, j = 1, \dots, n_{\text{el}}$. The eigenvalues E of Eq. 1 represent

[*]Equal contribution, author order random

37th Conference on Neural Information Processing Systems (NeurIPS 2023).

the energy states of a molecule, whereas a special interest lies in finding the smallest eigenvalue E_0, called the ground-state energy. The corresponding high-dimensional wavefunction $\psi : \mathbb{R}^{n_{\text{el}} \times 3} \to \mathbb{R}$ can be found via the Rayleigh-Ritz principle, by minimizing

$$\mathcal{L}(\psi_\theta) = \mathbb{E}_{\boldsymbol{r} \sim \psi_\theta^2(\boldsymbol{r})} \left[\frac{H\psi_\theta(\boldsymbol{r})}{\psi_\theta(\boldsymbol{r})} \right] \geq E_0. \tag{2}$$

Due to electrons being fermions, the solution must fulfill the anti-symmetry property, stating that the sign of the wavefunction must change for any permutation $\mathcal{P}$ of two electrons of the same spin: $\psi(\boldsymbol{r}) = -\psi(\mathcal{P}\boldsymbol{r})$. Having access to the solution ψ, allows in principle a complete description of the considered molecule. Unfortunately, only for one electron systems there exists an analytical solution and due to the curse of dimensionality, with increasing number of particles, obtaining an accurate approximation of the wavefunction becomes intractable already for medium-sized molecules. This is because many high-accuracy approximation methods scale poorly with n_{el}. For example CCSD(T) – coupled cluster with its single-, double-, and perturbative triple-excitations variant – is considered the gold-standard reference in computational chemistry, but its computational cost scales as $\mathcal{O}({n_{\text{el}}}^7)$ [2]. Deep-learning-based Variational Monte Carlo (DL-VMC) has emerged as a promising alternative solution. A single step scales as $\mathcal{O}({n_{\text{el}}}^4)$ and it has surpassed the accuracy of many conventional methods such as CCSD(T), when applied to small molecules [3]. In DL-VMC, the wavefunction is represented by a neural network with trainable parameters θ and optimized via Eq. 2 using gradient based optimization. Since the expectation value of Eq. 2 cannot be computed analytically, it is approximated by sampling the electron positions during optimization and evaluation with Monte Carlo methods like Metropolis-Hastings [4].

Related work FermiNet by Pfau et al. [3] and its variants have emerged as the leading architecture for DL-VMC in first quantization. It can reach highly accurate energies, but typically requires tens of thousands of optimization steps for convergence. Many improvements have been proposed to further increase accuracy [5–7] and accelerate convergence [8]. Furthermore, DL-VMC has been extended to properties beyond energies [9–11] and systems beyond molecules [12–16]. Despite the favorable scaling of DL-VMC, computational cost is still high, even for small molecules, often requiring thousands of GPUhs to find ψ for a single small molecule [17]. This is because unlike typical machine learning applications – which train an expensive model once, and subsequently achieve cheap inference – in DL-VMC the minimization of Eq. 2 is typically done from scratch for every new system.

A promising line of research to scale-up expensive ab-initio solvers such as DL-VMC or CCSD(T) has been to develop proxy methods, which can be trained on outputs of ab-initio methods to directly predict molecular properties [18, 19] or wavefunctions [20, 21] from the molecular geometry. While these proxy methods can often reproduce the underlying ab-initio method with high fidelity and scale to millions of atoms [22], they are fundamentally limited by the accuracy of their reference method and need many high-accuracy samples for training, reiterating the need for scaleable, high-accuracy reference methods.

To enable DL-VMC methods to efficiently compute wavefunctions for many molecules, several methods have been proposed to amortize the cost of optimization, by learning a single wavefunction across multiple systems. This has been demonstrated to work for different geometries of a single molecule [10, 23, 24], and recently two approaches have been proposed to learn wavefunctions across entirely different molecules, each with their own limitations. Gao et al. [17] reparameterized the orbitals of a wavefunction, by using chemistry-inspired heuristics to determine orbital positions, managing to efficiently generalize wavefunctions across different geometries of a single molecule. However, when learning a single wavefunction across different molecules, their results deteriorated and transfer to new molecules proved difficult. Scherbela et al. [25] do not require heuristic orbital positions, but instead use orbital descriptors of a low-accuracy conventional method to parameterize DL-VMC orbitals. However, while their wavefunction ansatz successfully transfers to new molecules, their method requires a separate, iterative Hartree-Fock (HF) calculation for every new geometry.

Our contribution This work presents the first end-to-end machine learning approach, which successfully learns a single wavefunction across many different molecules with high accuracy. Our contributions are:

- A transferable neural wavefunctions, which requires neither heuristic orbital positions, nor iterative HF calculations. We achieve this by building on the architecture by Scherbela et al. [25] and an orbital prediction model by Unke et al. [20].
- A simplified and improved electron embedding architecture, leveraging expressive nuclear features from our orbital prediction model and a message passing step between nuclei.
- A chemically diverse dataset with up to 100 molecules based on QM7-X [26], a data augmentation method based on normal mode distortions, and successful training of a neural wavefunction on these with continuously sampled geometries. Additionally, we improve the initialization of electrons around the molecule, reducing the computational overhead.
- As a final result, an accurate neural wavefunction, which shows for the first time zero-shot capabilities, i.e. high-accuracy energy predictions without additional optimization steps, on new systems. In particular it achieves better absolute energies than well established, high-accuracy gold-standard reference methods, such as CCSD(T)-3Z on unseen systems without any finetuning (cf. Fig. 2a).

Overview of the paper In Sec. 2 we outline our method and the procedure to optimize a transferable wavefunction across molecules. In Sec. 3 we thoroughly test the accuracy of our obtained wavefunction, by analyzing absolute energies (Sec. 3.1), relative energies (Sec. 3.2), and the impact of design choices in an ablation study (Sec. 3.3). Throughout this work, we compare against other high-accuracy methods, in particular results obtained by state-of-the-art DL-VMC methods and CCSD(T). Finally we analyze the scalability and limitations of our base model, by applying it to a large-scale dataset in Sec. 3.4, before a discussion and outlook for future research in Sec. 4.

2 Methods

Our approach is divided into two parts (cf. Fig. 1): On the one hand, a wavefunction ansatz, containing an electron embedding, orbital embedding and a Slater determinant. On the other hand a method for geometry sampling based on normal-mode distortions.

2.1 Our wavefunction ansatz

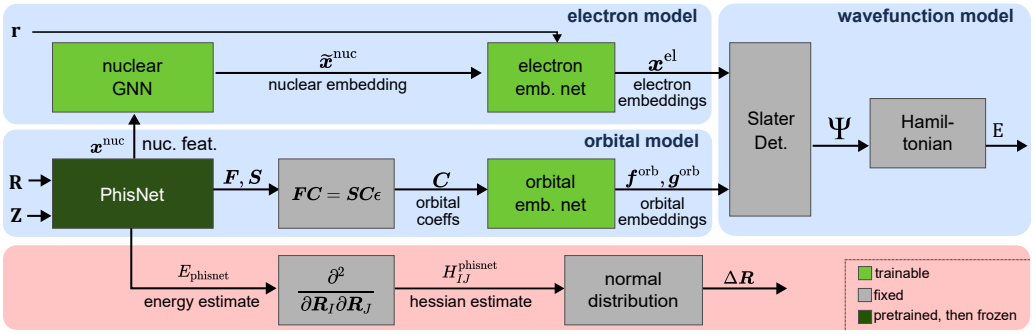

Figure 1: **Overview of our approach**: Wavefunction ansatz (top) and geometry sampling (bottom)

A single forward-pass for our wavefunction model

$$\psi = \sum_{d=1}^{N_{\text{det}}} \det\left[\phi_k^d(\boldsymbol{x}_i^{\text{el}})\right], \quad \phi_k^d(\boldsymbol{x}_i^{\text{el}}) = \sum_{I=1}^{N_{\text{nuc}}} \langle \boldsymbol{x}_i^{\text{el}}, \boldsymbol{f}_{Ikd}^{\text{orb}} \rangle e^{-r_{iI} g_{Ikd}^{\text{orb}}}, \quad i,k = 1,\dots,n_{\text{el}} \tag{3}$$

can again be divided into three blocks: An electron model acting on nuclear and electron coordinates, generating electron embeddings $\boldsymbol{x}^{\text{el}}$; an SE(3)-equivariant orbital model acting only on nuclear coordinates, generating orbital embeddings g^{orb} and $\boldsymbol{f}^{\text{orb}}$; a Slater determinant combining electron- and orbital-embeddings, and ensuring anti-symmetry of the wavefunction.

Message passing neural network Throughout this work we use message passing neural networks (MPNN), to operate on the graph of particles which are connected by edges containing information about their relative positions. The electron-electron, electron-nuclear and nuclear-nuclear edges are embedded with a multi-layer perceptron (MLP),

$$\boldsymbol{e}_{ij}^{\text{el-el}} = \text{MLP}\big([\boldsymbol{r}_{ij}, r_{ij}]\big) \qquad \boldsymbol{e}_{iI}^{\text{el-nuc}} = \text{MLP}\big([\boldsymbol{r}_{iI}, r_{iI}]\big) \qquad \boldsymbol{e}_{IJ}^{\text{nuc-nuc}} = \text{MLP}\big([\boldsymbol{R}_{IJ}, R_{IJ}]\big), \tag{4}$$

by using a concatenation ($[\cdot]$) of distance and difference vectors, and separate weights for each MLP. A single message passing step is decomposed into the following operations

$$\tilde{\boldsymbol{a}}_i^{\text{rec}} = \text{MessagePassing}(\boldsymbol{a}_i^{\text{rec}}, \{\boldsymbol{a}_j^{\text{send}}\}, \{\boldsymbol{e}_{ij}\}) \tag{5}$$

$$= \sigma\left(\text{Linear}(\boldsymbol{a}_i^{\text{rec}}) + \text{Linear}\big(\sum_j \text{Linear}(\boldsymbol{a}_j^{\text{send}}) \odot \text{Linear}(\boldsymbol{e}_{ij})\big)\right) \tag{6}$$

for a receiving particle $\tilde{\boldsymbol{a}}_i^{\text{rec}}$ and the set of sending particles $\{\boldsymbol{a}_j^{\text{send}}\}$, connected via their edges $\{\boldsymbol{e}_{ij}\}$. By σ we denote the non-linear activation and with $\odot$ the element-wise multiplication along the feature dimension. An MPNN is obtained by stacking MessagePassing layers

$$\text{MPNN}(\boldsymbol{a}_i, \{\boldsymbol{a}_j\}, \{\boldsymbol{e}_{ij}\}) = \text{MessagePassing}(\ldots \text{MessagePassing}(\boldsymbol{a}_i, \{\boldsymbol{a}_j\}, \{\boldsymbol{e}_{ij}\})) \tag{7}$$

In the following we use these message passing steps to model all inter-particle interactions.

SE(3)-equivariant orbital model The orbital model is a simplified version of PhisNet[20], a neural network predicting the overlap matrix $\boldsymbol{S}$ and the Fock matrix $\boldsymbol{F}$, via nuclear embeddings $\boldsymbol{x}^{\text{nuc}}$:

$$\boldsymbol{x}^{\text{nuc}} = \text{phisnet}_\theta(\boldsymbol{R}, \boldsymbol{Z}) \qquad \boldsymbol{S}_{IJ} = s_\theta(\boldsymbol{x}_I^{\text{nuc}}, \boldsymbol{x}_J^{\text{nuc}}) \qquad \boldsymbol{F}_{IJ} = f_\theta(\boldsymbol{x}_I^{\text{nuc}}, \boldsymbol{x}_J^{\text{nuc}}). \tag{8}$$

Here $\boldsymbol{x}^{\text{nuc}} \in \mathbb{R}^{N_{\text{nuc}} \times (L+1)^2 \times N_{\text{channels}}}$, and $\boldsymbol{S}_{IJ}$ and $\boldsymbol{F}_{IJ}$ are each in $\mathbb{R}^{N_{\text{basis}} \times N_{\text{basis}}}$. The basis-set size of the predicted orbitals is denoted by N_{basis} and the feature dimension of the nuclear embeddings by N_{channels}. The full overlap- and Fock-matrices are assembled from the corresponding blocks $\boldsymbol{S}_{IJ}$ and $\boldsymbol{F}_{IJ}$, leading to matrices of shape $[N_{\text{nuc}} N_{\text{basis}} \times N_{\text{nuc}} N_{\text{basis}}]$. Each layer of PhisNet is SE(3)-equivariant, ensuring that any 3D-rotation or inversion of the input coordinates $\boldsymbol{R}$, leads to an equivalent rotation of its outputs. This is done by splitting any feature vector into representations of varying harmonic degree $l = 0, \ldots, L$, each with components $m = -l, \ldots, l$. A detailed description of SE3-equivariant networks in general, as well as PhisNet in particular can be found in [20]. A list of changes and simplifications we made to PhisNet can be found in Appendix C.

The orbital embeddings (corresponding to orbital expansion coefficients in a conventional quantum chemistry calculation) are obtained by solving the generalized eigenvalue problem:

$$\boldsymbol{F}\boldsymbol{C}_k = \boldsymbol{S}\boldsymbol{C}_k \epsilon_k \qquad \hat{\boldsymbol{x}}_{Ik}^{\text{orb}} = \text{reshape}(\boldsymbol{C}_k, [N_{\text{nuc}}, N_{\text{basis}}])_I \tag{9}$$

It has been shown empirically that it is beneficial to obtain the orbital coefficients $\hat{\boldsymbol{x}}_{Ik}^{\text{orb}}$ as solutions to this generalized eigenvalue problem, rather than predicting them directly [27] as functions of $\boldsymbol{R}$ and $\boldsymbol{Z}$. This is because the orbital coefficients are neither unique, nor do they share the molecule's symmetry. The matrices $\boldsymbol{F}$ and $\boldsymbol{S}$ on the other hand are unique and transform equivariantly under E(3)-transformations of the molecule, leading to a well defined learning problem.

Following [25], we do not use the orbital energies ϵ_k and obtain the backflow factors $\boldsymbol{f}$ and exponents $\boldsymbol{g}$, by first localizing the resulting orbitals using the Foster-Boys localization [28] (cf. Appendix B) and subsequently using an MPNN and MLP acting on the orbital embeddings.

$$\widetilde{\boldsymbol{x}}_{Ik}^{\text{orb}} = \sum_{n=1}^{N_{\text{orb}}} U_{kn}^{\text{loc.}} \hat{\boldsymbol{x}}_{In}^{\text{orb}}, \qquad \boldsymbol{x}_{Ik}^{\text{orb}} = \text{MPNN}(\widetilde{\boldsymbol{x}}_{Ik}^{\text{orb}}, \{\widetilde{\boldsymbol{x}}_{Jk}^{\text{orb}}\}, \{\boldsymbol{e}_{IJ}^{\text{nuc-nuc}}\}) \tag{10}$$

$$\boldsymbol{f}_{Ik}^{\text{orb}} = \text{MLP}(\boldsymbol{x}_{Ik}^{\text{orb}}), \qquad \boldsymbol{f}_{Ik}^{\text{orb}} \in \mathbb{R}^{N_{\text{nuc}} \times N_{\text{orb}} \times N_{\text{det}} \times N_{\text{emb}}} \tag{11}$$

$$g_{Ik}^{\text{orb}} = \text{MLP}(\boldsymbol{x}_{Ik}^{\text{orb}}), \qquad g^{\text{orb}} \in \mathbb{R}^{N_{\text{nuc}} \times N_{\text{orb}} \times N_{\text{det}}} \tag{12}$$

Electron model The electron embedding is a message passing neural network. To incorporate the geometric information of the molecule considered, we leverage the equivariant prediction of the PhisNet nuclear embeddings $\boldsymbol{x}^{\text{nuc}}$, by first performing a message passing step between the nuclear embeddings

$$\hat{\boldsymbol{x}}_I^{\text{nuc}} = \text{MLP}(\boldsymbol{x}_I^{\text{nuc}}) \qquad \tilde{\boldsymbol{x}}_I^{\text{nuc}} = \text{MessagePassing}(\hat{\boldsymbol{x}}_I^{\text{nuc}}, \{\hat{\boldsymbol{x}}_J^{\text{nuc}}\}, \{\boldsymbol{e}_{IJ}^{\text{nuc-nuc}}\})$$

and then using these features to initialize the electron embeddings

$$\boldsymbol{x}_i^{\text{el},0} = \text{MessagePassing}(\mathbf{0}, \{\tilde{\boldsymbol{x}}_J^{\text{nuc}}\}, \{\boldsymbol{e}_{iJ}^{\text{el-nuc}}\}),$$

by using a zero vector $\mathbf{0}$ for the initial receiving electrons. This differentiates our electron model from previously proposed methods [3, 6, 8], leading to a better generalization when optimized across molecules (cf. Sec. 3.3). The final step of the embedding is a multi-iteration message passing between electron embeddings to capture the necessary electron-electron interaction

$$\boldsymbol{x}_i^{\text{el}} = \text{MPNN}(\boldsymbol{x}_i^{\text{el},0}, \{\boldsymbol{x}_j^{\text{el},0}\}, \{\boldsymbol{e}_{ij}^{\text{el-el}}\}),$$

resulting in a N_{emb}-dimensional representation for each electron $\boldsymbol{x}_i^{\text{el}} \in \mathbb{R}^{N_{\text{emb}}}, i = 1, \ldots, n_{\text{el}}$.

Overall E(3)-equivariance Like existing approaches [3, 8, 17] our overall wavefunction does not enforce E(3)-symmetry. This is because the wavefunction can have lower symmetry than the molecule [23], for example in the case of the excited states of a hydrogen atom. We therfore choose all parts of the network that act purely on nuclear coordinates (i.e. the PhisNet model and the energy/hessian estimate) to be equivariant under E(3)-transformations. All parts of the network acting on electron coordinates (in particular the electron model) break this symmetry by depending explicitly on the cartesian coordinates of the electrons. The architecture is thus only invariant under translations, but not under rotations or inversions. We bias the model towards approximately invariant energies using data augmentation as discussed in Sec. 2.3.

2.2 Sampling

Markov Chain Monte Carlo (MCMC) sampling of electron positions We use MCMC to draw samples $\boldsymbol{r}$ from the probability distribution $\psi(\boldsymbol{r})^2$, to evaluate the expectation value of Eq. 2. One notable difference compared to other works is our initialization $\boldsymbol{r}^0$ of the Markov Chain. In the limit of infinite steps, the samples are distributed according to ψ^2, but for a finite number of steps the obtained samples strongly depend on $\boldsymbol{r}^0$. This issue is typically addressed by a "burn-in", where MCMC is run for a fixed number of steps (without using the resulting samples) to ensure that $\boldsymbol{r}$ has diffused to state of high probability. Previous work has initialized $\boldsymbol{r}^0$ using a Gaussian distribution of the electrons around the nuclei. We find that this initialization is far from the desired distribution ψ^2 and thus requires $\approx 10^5$ MCMC steps to reach the equilibrium distribution. We instead initialize $\boldsymbol{r}^0$ by samples drawn from an exponential distribution around the nuclei, which much better approximates the correct distribution and thus equilibrates substantially faster (cf. Appendix A). We find that exponential initialization reduces the required number of burn-in steps by ca. 50%, reducing the computational cost of a 500-step zero- shot evaluation by ca. 5%.

Normal mode sampling of geometries Since DL-VMC is an ab-initio method, we do not require a labeled dataset of reference energies, but to obtain a transferable wavefunction, which generalized well to new systems, a diverse dataset of molecular geometries $\boldsymbol{R}$ is required. Starting with an initial set of geometries $\boldsymbol{R}^0$, we update $\boldsymbol{R}$ on the fly, by perturbing each geometry every 20 optimization steps by adding random noise $\Delta\boldsymbol{R}$ to the nuclear coordinates. Using uncorrelated, isotropic random noise for $\Delta\boldsymbol{R}$ would yield many non-physical geometries $\boldsymbol{R}'$, since the stiffness of different degrees of freedom can vary by orders of magnitude. Intuitively we want to make large perturbations along directions in which the energy changes slowly, and vice-versa. We achieve this by sampling $\Delta\boldsymbol{R}$ from a correlated normal distribution

$$\Delta\boldsymbol{R} \sim \mathcal{N}(\alpha(\boldsymbol{R}^0 - \boldsymbol{R}), \beta\boldsymbol{H}_{\text{phis}}^{-1}) \qquad \boldsymbol{R}' = \boldsymbol{R} + \Delta\boldsymbol{R}. \tag{13}$$

The bias term $\alpha(\boldsymbol{R}^0 - \boldsymbol{R})$ ensures that geometries stay sufficiently close to their starting point $\boldsymbol{R}^0$. The covariance matrix is chosen proportional to the pseudo-inverse of the hessian of the energy E^{phis}, which is predicted from the scalar component of the nuclear embeddings using a pre-trained MLP. By using $\boldsymbol{H}_{\text{phis}}^{-1}$ as covariance matrix, we take large steps along soft directions and small steps along stiff directions, thus avoiding unphysical geometries with very high energies.

$$E^{\text{phis}} = \sum_{I=1}^{N_{\text{nuc}}} \text{MLP}(\boldsymbol{x}_I^{\text{nuc}}), \qquad H_{I\zeta,J\xi}^{\text{phis}} = \frac{\partial^2 E^{\text{phis}}}{\partial \boldsymbol{R}_{I\zeta} \partial \boldsymbol{R}_{J\xi}}, \quad I, J = 1 \ldots N_{\text{nuc}} \quad \zeta, \xi = 1 \ldots 3 \tag{14}$$

After distorting the nuclear coordinates $\boldsymbol{R}$, we also adjust the electron positions $\boldsymbol{r}$, using the space-warp coordinate transform described in [29], which effectively shifts the electrons by a weighted average of the shift of their neighbouring nuclei. In addition to this distortion of the molecule, we also apply a random global rotation to all coordinates, to obtain a more diverse dataset.

2.3 Optimization

To obtain orbital descriptors (and energies to calculate the hessian of the energy) we pre-train PhisNet against the Fock matrix, the overlap matrix, the energy and the forces of Hartree-Fock calculations in a minimal basis set across 47k molecules. Further details of the loss function, dataset, and the adaptions to PhisNet can be found in Appendix C. For all subsequent experiments, we freeze the parameters of PhisNet. A full DL-VMC calculation to obtain a ground-state energy prediction can be divided into three consecutive steps:

1. **Supervised optimization using PhisNet**: Initially, the neural-network orbitals (cf. Eq. 3) are optimized to minimize the residual against orbitals obtained from PhisNet. It ensures that the initial wavefunction roughly resembles the true ground-state and is omitted when fine-tuning an already optimized base model.
2. **Variational optimization**: Minimization of the energy (cf. Eq. 2) by drawing samples from ψ^2 using MCMC and updating the wavefunction parameters using the KFAC optimizer [30].
3. **Evaluation**: For inference of the ground-state energy, we sample electron positions using MCMC, and evaluate the energy using Eq. 2 without updating θ.

To train a multi-geometry transferable wavefunction we further divide the variational optimization of a neural wavefunction into two steps:

1. **Pre-training**: A single wavefunction model is trained across many molecules and geometries. In every gradient step we only consider a single geometry per batch. The next geometry to optimize is chosen based on the energy variance as proposed by [10]. To sample continuously the space of molecular geometries we distort each geometry every 20 optimization steps as described in Sec. 2.2. We refer to evaluations of this model on new systems as "zero-shot".
2. **Fine-tuning**: A a small number of additional variational optimization steps is done using geometries of interest, starting from the weights of a pre-trained base model. This procedure yields a model that is specialized to the molecule at hand and typically yields more accurate energies on the specific problem than the raw pre-trained model.

3 Results

We pre-train our wavefunction model on a dataset of 98 molecules (699 conformers) for 256k optimization steps using the architecture and training procedure outlined in Sec. 2. Below we demonstrate the performance of this model, for zero-shot evaluations and after subsequent fine-tuning.

3.1 Accuracy of pre-trained model for absolute energies

To test the transfer capabilities of the model, we evaluate it on test-sets, which each contain 4 randomly chosen and perturbed molecules, grouped by molecule size (measured as the number of non-Hydrogen atoms). To avoid train/test leakage, we excluded all molecules that are part of these test sets from the training set (cf. Appendix D). Although the model has only been trained on molecules containing up to 4 heavy atoms, we evaluate its performance across the full range up to 7 heavy atoms. We find that for molecules containing up to 6 heavy atoms our method outperforms CCSD(T) with a 2Z basis set and outperforms CCSD(T) with a 4Z basis set after only 4k fine-tuning steps. This is a large improvement over the state of the art: Zero-shot evaluations by Gao et al. [17] did not manage to outperform a Hartree-Fock baseline, even on the toy system of Hydrogen-chains. Similarly, Scherbela et al. [25] achieve high accuracy after fine-tuning, but result that are worse than Hartree-Fock in a zero-shot setting. In contrast, our improvements to their method increase zero-shot accuracy by more than 2 orders of magnitude.

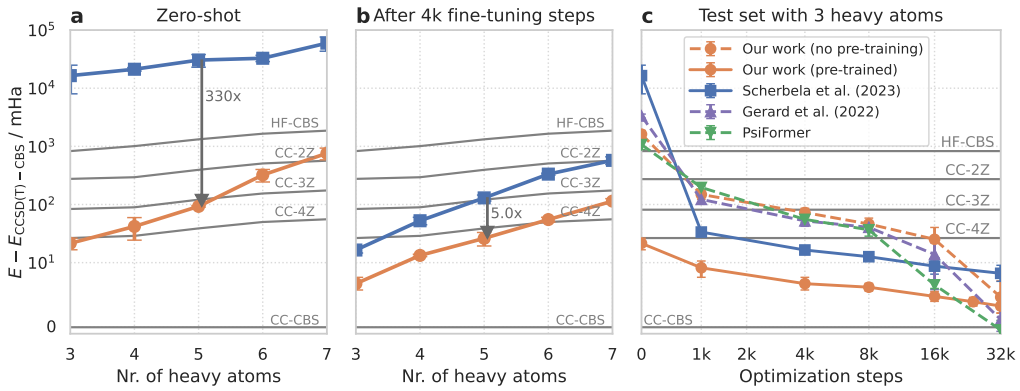

Figure 2: **Absolute energies**: Energies relative to CCSD(T)-CBS (complete basis set limit) when re-using the pre-trained model on molecules of varying size without optimization (a) and after fine-tuning (b). (c) depicts energy for the test set containing 3 heavy atoms as a function of optimization steps and compares against SOTA methods. Solid lines are with pre-training, dashed lines without. Gray lines correspond to conventional methods: Hartree-Fock in the complete basis set limit (HF-CBS), and CCSD(T) with correlation consistent basis basis sets of double to quadruple valence (CC-nZ).

We furthermore find that fine-tuning pre-trained wavefunctions can be more cost effective than well established conventional methods. For a typical molecule containing 5 heavy atoms, we require 9.5 node-hours for 4k fine-tuning steps, achieving accuracy that surpasse CCSD(T)-4Z. In contrast, CCSD(T)-4Z requires 10.5 node-hours and 16k steps of PsiFormer (achieving the same accuracy), require 53 node-hours. All run-times are listed in Tab. 3 of Appendix I.

To further test the accuracy of our method, we evaluate for 3 heavy atoms the performance with increasing fine-tuning steps (cf. Fig. 2c). We compare our work (with and without fine-tuning) against CCSD(T) and reference calculations done with state-of-the-art DL-VMC methods [6, 8]. For up to 16k optimization steps, our pre-trained model yields energy errors that are 1-2 orders of magnitude lower than other reference methods. After longer optimization the method by Gerard et al. [8] and PsiFormer [6] surpass our predictions. We hypothesize that this is not to blame on pre-training, but that our orbital prediction framework, which allows us to optimize across molecule, yields less expressive wavefunctions than a fully trainable backflow. This is demonstrated by the fact that a pre-trained and non-pre-trained model converge to the same energy in the limit of long optimization.

Like the absolute energies, also the variance of the local energies (another measure of wavefunction accuracy) is substantially improved by our method and results are depicted in Appendix E.

3.2 Accuracy for relative energies

While Sec. 3.1 demonstrates high accuracy for absolute energies, we find that relative energies (being the small difference between two large absolute energies) can be unsatisfactory in the zero-shot regime, and a small number of fine-tuning steps is required to reach quantitatively correct relative energies. Fig. 3 demonstrates this issue on 4 distinct systems, each highlighting a different challenge for our model. For each system, we evaluate our pre-trained base model without any system specific optimization (zero-shot), and after 4000 fine-tuning steps. The fine-tuning optimization is done separately for each of the 4 systems, analogously to Sec. 2.3, yielding 4 distinct wavefunctions that each represent the ground-state wavefunctions of all considered geometries per system. Results after more fine-tuning steps can be found in Appendix F.

Bicyclobutane conformers Fig. 3a depicts the energies of 5 conformers of bicyclobutane relative to the energy of its initial structure. The system is of interest, because CCSD(T) severely underestimates the energy of the dis-TS conformer by ≈ 60 mHa [31]. While our zero-shot results yield the correct sign for the relative energies, they are quantitatively far off from the gold-standard DMC reference calculation [31], in particular for the dis-TS geometry, where we overestimate the relative energy

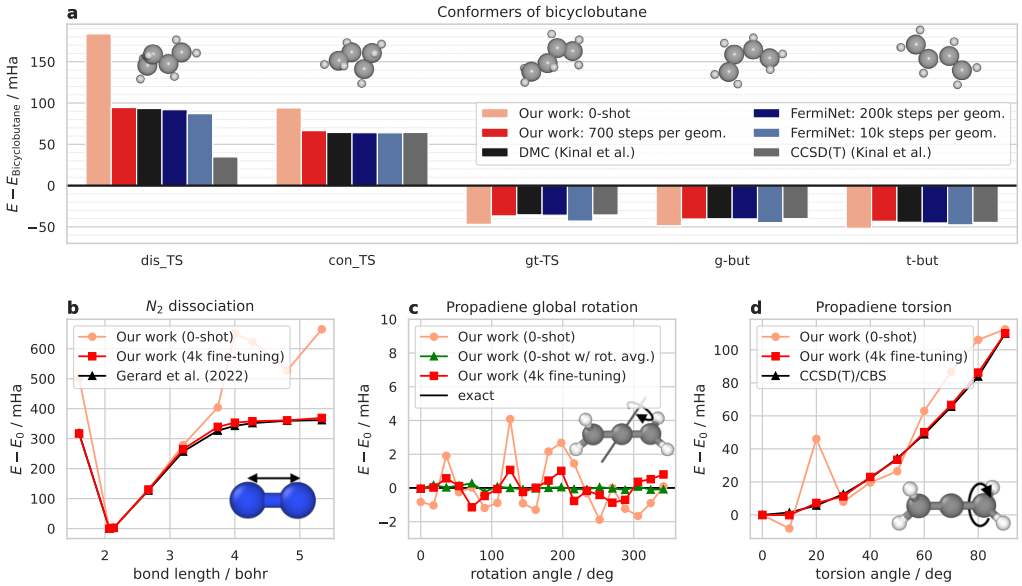

Figure 3: **Challenging relative energies**: Relative energies obtained with and without fine-tuning on 4 distinct, challenging systems, compared against high-accuracy reference methods. a) Relative energy of bicyclobutane conformers vs. the energy of bicyclobutane b) Potential energy surface (PES) of N_2, c) global rotation of propadiene d) relative energy of twisted vs. untwisted propadiene.

by 90 mHa. However when fine-tuning our model for only 700 steps per geometry (4k total), we obtain relative energies that are in close agreement with DMC (max. deviation 2.1 mHa). Comparing to FermiNet [5] we find that our results are more accurate than a FermiNet calculation after 10k steps (requiring twice our batch size; max. deviation 7.5 mHa), and slightly less accurate than a FermiNet calculation optimized for 200k steps (max. deviation 1.4 mHa). As opposed to CCSD(T) our model does not suffer from systematic errors, even for the challenging dis-TS geometry. A table of all relative energies can be found in Appendix F.

Nitrogen dissociation When evaluating the energy of an N_2 molecule at various bond-lengths we obtain high zero-shot accuracy near the equilibrium geometry ($d = 2.1$ bohr; in training set), but substantially lower accuracy at smaller or large bond lengths (cf. Fig. 3b). This is due to the lack of dissociated atoms in the pre-training dataset and again mostly remedied by finetuning. In the most challenging regime around $d = 3.7$, even with fine-tuning we obtain energies that are 12 mHa above high-accuracy results obtained by Gerard et al. [8], indicating the need for longer fine-tuning.

Global rotation of propadiene Energies of molecules are invariant under global translation or rotation of all particle positions. The wavefunction however is not invariant and neither is our wavefunction ansatz, which can lead to different energies for rotated copies of a molecule. When evaluating the energy of our model on 20 copies of propadiene (C_3H_4) rotated around a random axis, we find typical energy variations of ± 1 mHa, but also a individual outliers, deviating by up to 5 mHa. This highlights a dilemma facing all existing DL-VMC models: On the one hand, constraining the wavefunctions to be fully invariant under rotation (or even just invariant under symmetries of the Hartree-Fock orbitals) is too restrictive to express arbitrary ground-state wavefunctions [23]. One the other hand, our approach of biasing the model towards rotation-invariant energies by data augmentation, appears to be helpful but not to be fully sufficient. When evaluating an earlier checkpoint of our base model (trained for 170k epochs instead of 256k), we observe mean energy variations of ± 3 mHa and outliers of up to 30 mHa, indicating the positive impact of prolonged training with data augmentation. We again find 4000 fine-tuning steps split across all geometries are sufficient to reduce errors below chemical accuracy of 1.6 mHa.

In addition to augmenting data during training time, we can also augment the data during inference time, to obtain fully rotation-invariant energies, from an approximately rotation invariant model. We

achieve this by randomly rotating the molecule at every inference step and averaging across all rotated geometries. Since the Monte-Carlo estimate is already an average across many different samples anyways, this comes at no additional computational cost. Fig. 3c shows that with rotation averaging we obtain energies that are fully invariant, up to Monte-Carlo noise.

Twisted propadiene Twisting one of the C=C bonds of propadiene leads to a transition state with an energy difference of 110 mHa. Evaluating the energy without fine-tuning on an equidistant grid of torsion angles, we obtain the correct barrier height (112 mHa), but also deviations of up to 40 mHa. During pre-training we purposefully sampled twisted molecules, but only included equilibrium geometries, transition geometries, and one intermediate twist (cf. Appendix D). This seems sufficient for correct zero-shot barrier heights, but insufficient for high accuracy along the full path. Short fine-tuning (4k steps distributed across all 10 geometries) yields excellent agreement with CCSD(T): $\sim$0.2 mHa discrepancy for the barrier height and a maximum deviation of 2 mHa along the path.

3.3 Ablation studies

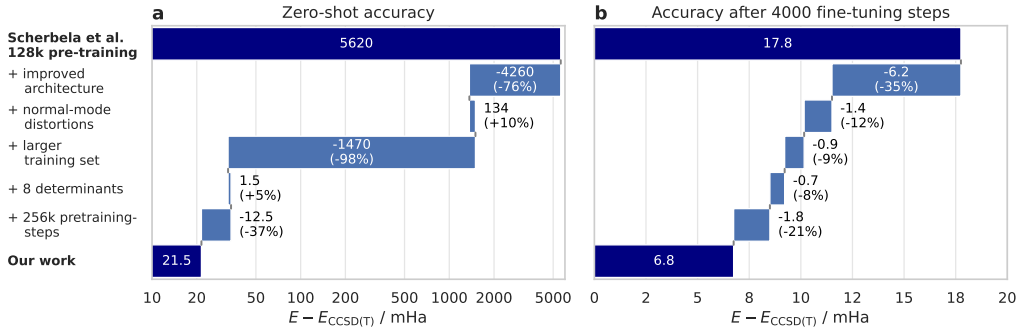

Figure 4: **Ablation study**: Breakdown of absolute energy difference between Scherbela et al. [25] and our work, when evaluating the respective base models on the test set consisting of 3 heavy atoms. Panel a shows accuracy gains for zero-shot evaluation, panel b the accuracy after 4k fine-tuning steps. First and last row depict their and our absolute energy respectively, intermediate bars show the subsequent improvements of various changes in mHa (and %). Note that panel a. is plotted on a logarithmic scale due to the large energy difference.

To analyze the relative importance of our changes, we break down the accuracy gap between our work and the prior work by Scherbela et al. [25] in Fig. 4. We start with their model checkpoint trained for 128k epochs on their dataset consisting of 18 different compounds. Next, we train a model using our improved architecture with their dataset and methodology, already reducing the zero-shot error by 76% and fine-tuning error by 35%. The next model additionally uses normal-mode distortions to augment the training dataset, decreasing the fine-tuning error by another 12%. Additionally increasing the training set size to our dataset containing 98 molecules and the corresponding torsional conformers, reduces the zero-shot error by 98% and yields modest improvements in the fine-tuning case. Increasing the number of determinants from 4 to 8 and increasing the number of pre-training steps from 128k to 256k improve fine-tuning accuracy by 8% and 21% respectively. The two largest contributions overall are the improved architecture – yielding substantial gains both in zero-shot and fine-tuning – as well as the larger training set, which is crucial for high zero-shot accuracy. This is consistent with [25], which found that both model size and training set size do improve performance, while pre-training duration shows diminishing returns after 256k steps. Since we pre-train the PhisNet model against Hartree-Fock references and do not update its parameters during optimization, the PhisNet model by itself contributes no substantial accuracy improvement. It impacts energy predictions indirectly, by providing pre-trained nuclear emebddings $\boldsymbol{x}^{\text{nuc}}$ and enabling fast geometry distortions and rotations during training and inference, which in turn improve accuracy.

3.4 Large-scale experiment

To showcase the scalability of our approach, we evaluate zero-shot predictions of the absolute energies for a subset of 250 molecules from the QM7 dataset [32], containing molecules with 14-58 electrons.

Since CCSD(T) calculations would be prohibitively expensive for a dataset of this size, in Fig. 5a we compare against density functional theory (DFT) reference calculations (PBE0+MBD, [26]). Fig. 5b breaks down the energy differences compared to DFT, by molecule size. For molecules with less than 5 heavy atoms we obtain energies that are lower than DFT by 10-120 mHa. Because our ansatz is variational (in contrast to DFT), our lower energies translate to a more accurate prediction of the true ground-state energy, confirming again our high zero-shot accuracy. With increasing system size the accuracy deteriorates, due to increasing extrapolation and out-of-distribution predictions, consistent with our results in Sec. 3.1. One reason for the large energy differences in larger molecules is the increasing inter-atomic distance within the molecules with increasing number of atoms (cf. Fig. 5c). While the largest inter-atomic distance observed in the training set is 11 bohr, the evaluation set contains distances up to 17 bohr. This issue could be overcome by employing a distance cutoff, as it is already applied in supervised machine-learned potential energy surface predictions [19, 33] or has just recently been incorporated into a neural wavefunction [17] via an exponential decay with increasing inter-particle distance. Another potential solution is to include larger molecules or separated molecule fragments in the pre-training molecule dataset, reducing the extrapolation regime.

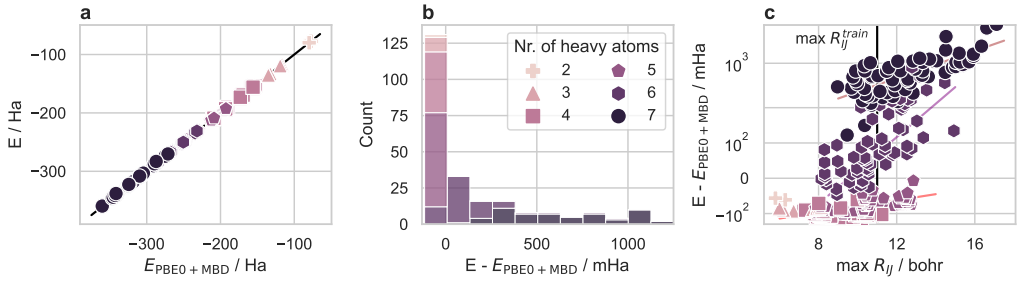

Figure 5: **Zero-shot on QM7**: a) Our zero-shot energies vs. DFT [26] b) Histogram of energy residuals (truncated at 1.25 Ha for clarity) c) Residuals vs. largest inter-molecular distance R_{IJ}.

4 Discussion

We have presented to our knowledge the first ab-initio wavefunction model, which achieves high-accuracy zero-shot energies on new systems (Sec. 3.1 and Sec. 3.4). Our pre-trained wavefunction yields more accurate total energies than CCSD(T)-2Z across all molecule sizes and outperforms CCSD(T)-3Z on molecules containing up to 5 heavy atoms, despite having been trained only on molecules containing up to 4 heavy atoms. We find that relative energies of our model are qualitatively correct without fine-tuning, but need on the order of 4000 fine-tuning steps to reach chemical accuracy of 1.6 mHa (Sec. 3.2). This is a substantial improvement over previous work, which so far has fallen in two categories: High-accuracy ansätze (such as [3, 6]) that cannot generalize across molecules and thus need ca. 10x more compute to reach the same accuracy, or methods that can generalize ([17, 25]), but yield orders of magnitude lower accuracy in the zero- or few-shot regime. We demonstrate in Sec. 3.3 that these improvements are primarily driven by an improved architecture (containing a more expressive electron embedding and an ML-based orbital model), improved geometry sampling, and a larger training dataset.

While results are encouraging in the zero- and few-shot regime, open questions for further research abound. The most pressing issues are currently limited zero-shot accuracy for relative energies, and potentially limited expressiveness of the ansatz in the regime of very long optimization. Zero-shot accuracy could be further improved by training on an even larger dataset, further improved geometry sampling (in particular of torsion angles), and an interaction cut-off to avoid previously unseen particle pairs for new large molecules. Furthermore SE(3)-symmetry of the wavefunction should be explored further, since currently only the orbital part of our architecture is SE(3)-equivariant. We experimented with a fully equivariant architecture, but found the resulting wavefunctions to not be expressive enough. To improve overall accuracy, attention based embeddings [6, 34] could be pursued. Additionally, we currently freeze the weights of the orbital embedding to simplify the architecture and avoid back-propagation through the iterative orbital localization procedure. Optimizing these weights in addition to the electron embedding will lead to a more expressive ansatz.

Acknowledgements

We gratefully acknowledge financial support from the following grants: Austrian Science Fund FWF Project I 3403 (P.G.), WWTF-ICT19-041 (L.G.). The computational results have been achieved using the Vienna Scientific Cluster (VSC). The funders had no role in study design, data collection and analysis, decision to publish or preparation of the manuscript.

References

[1] M. Born and R. Oppenheimer. "Zur Quantentheorie der Molekeln". In: *Annalen der Physik* 389.20 (1927), pp. 457–484. DOI: https://doi.org/10.1002/andp.19273892002.

[2] László Gyevi-Nagy, Mihály Kállay, and Péter R. Nagy. "Accurate Reduced-Cost CCSD(T) Energies: Parallel Implementation, Benchmarks, and Large-Scale Applications". In: *Journal of Chemical Theory and Computation* 17.2 (Feb. 2021), pp. 860–878. DOI: 10.1021/acs.jctc.0c01077.

[3] David Pfau et al. "Ab Initio Solution of the Many-Electron Schrödinger Equation with Deep Neural Networks". In: *Phys. Rev. Res.* 2.3 (Sept. 2020), p. 033429. DOI: 10.1103/PhysRevResearch.2.033429.

[4] W. K. Hastings. "Monte Carlo Sampling Methods Using Markov Chains and Their Applications". In: *Biometrika* 57.1 (Apr. 1970), pp. 97–109. DOI: 10.1093/biomet/57.1.97.

[5] James S. Spencer et al. "Better, Faster Fermionic Neural Networks". In: (2020).

[6] Ingrid von Glehn, James S. Spencer, and David Pfau. *A Self-Attention Ansatz for Ab-initio Quantum Chemistry*. Nov. 2022. DOI: 10.48550/arXiv.2211.13672.

[7] Weiluo Ren et al. "Towards the ground state of molecules via diffusion Monte Carlo on neural networks". In: *Nature Communications* 14.1 (Apr. 2023), p. 1860. DOI: 10.1038/s41467-023-37609-3.

[8] Leon Gerard et al. "Gold-Standard Solutions to the Schrödinger Equation Using Deep Learning: How Much Physics Do We Need?" In: *Advances in Neural Information Processing Systems*. Oct. 2022.

[9] M. T. Entwistle et al. "Electronic excited states in deep variational Monte Carlo". In: *Nature Communications* 14.1 (Jan. 2023), p. 274. DOI: 10.1038/s41467-022-35534-5.

[10] Michael Scherbela et al. "Solving the Electronic Schrödinger Equation for Multiple Nuclear Geometries with Weight-Sharing Deep Neural Networks". In: *Nature Computational Science* 2.5 (May 2022), pp. 331–341. DOI: 10.1038/s43588-022-00228-x.

[11] Yubing Qian et al. "Interatomic Force from Neural Network Based Variational Quantum Monte Carlo". In: *The Journal of Chemical Physics* 157.16 (Oct. 2022), p. 164104. DOI: 10.1063/5.0112344.

[12] Gino Cassella et al. "Discovering Quantum Phase Transitions with Fermionic Neural Networks". In: *Physical Review Letters* 130.3 (Jan. 2023), p. 036401. DOI: 10.1103/PhysRevLett.130.036401.

[13] Xiang Li, Zhe Li, and Ji Chen. "Ab initio calculation of real solids via neural network ansatz". In: *Nature Communications* 13.1 (Dec. 2022), p. 7895. DOI: 10.1038/s41467-022-35627-1.

[14] Giuseppe Carleo and Matthias Troyer. "Solving the Quantum Many-Body Problem with Artificial Neural Networks". In: *Science* 355.6325 (Feb. 2017), pp. 602–606. ISSN: 0036-8075, 1095-9203. DOI: 10.1126/science.aag2302.

[15] Giuseppe Carleo et al. "NetKet: A Machine Learning Toolkit for Many-Body Quantum Systems". In: *SoftwareX* 10 (July 2019), p. 100311. DOI: 10.1016/j.softx.2019.100311.

[16] Yuan-Hang Zhang and Massimiliano Di Ventra. "Transformer Quantum State: A Multipurpose Model for Quantum Many-Body Problems". In: *Physical Review B* 107.7 (Feb. 2023), p. 075147. DOI: 10.1103/PhysRevB.107.075147.

[17] Nicholas Gao and Stephan Günnemann. *Generalizing Neural Wave Functions*. Feb. 2023. DOI: 10.48550/arXiv.2302.04168.

[18] Simon Batzner et al. "E(3)-Equivariant Graph Neural Networks for Data-Efficient and Accurate Interatomic Potentials". In: *Nature Communications* 13.1 (May 2022), p. 2453. DOI: 10.1038/s41467-022-29939-5.

[19] Ilyes Batatia et al. "MACE: Higher Order Equivariant Message Passing Neural Networks for Fast and Accurate Force Fields". In: *Advances in Neural Information Processing Systems*. 2022.

[20] Oliver Unke et al. "SE(3)-Equivariant Prediction of Molecular Wavefunctions and Electronic Densities". In: *Advances in Neural Information Processing Systems*. 2021.

[21] M. Gastegger et al. "A Deep Neural Network for Molecular Wave Functions in Quasi-Atomic Minimal Basis Representation". In: *The Journal of Chemical Physics* 153.4 (July 2020), p. 044123. DOI: 10.1063/5.0012911.

[22] Albert Musaelian et al. “Scaling the Leading Accuracy of Deep Equivariant Models to Biomolecular Simulations of Realistic Size”. In: (Apr. 2023). DOI: 10.48550/arXiv.2304.10061. arXiv: 2304.10061.

[23] Nicholas Gao and Stephan Günnemann. “Ab-Initio Potential Energy Surfaces by Pairing GNNs with Neural Wave Functions”. In: *arXiv:2110.05064 [physics]* (Nov. 2021). arXiv: 2110.05064 [physics].

[24] Nicholas Gao and Stephan Günnemann. “Sampling-free Inference for Ab-Initio Potential Energy Surface Networks”. In: (2022). DOI: 10.48550/ARXIV.2205.14962.

[25] Michael Scherbela, Leon Gerard, and Philipp Grohs. “Towards a Foundation Model for Neural Network Wavefunctions”. In: *arXiv.org* (Mar. 2023).

[26] Johannes Hoja et al. “QM7-X, a comprehensive dataset of quantum-mechanical properties spanning the chemical space of small organic molecules”. In: *Scientific Data* 8.1 (Feb. 2021), p. 43. DOI: 10.1038/s41597-021-00812-2.

[27] Ruard van Workum, Joao Malhado, and Philipp Marquetand. *CASNet: Learning Complete Active Space Orbitals Using Message Passing Neural Networks*. June 2023. DOI: 10.26434/chemrxiv-2023-lwj87.

[28] János Pipek and Paul G. Mezey. “A Fast Intrinsic Localization Procedure Applicable for Ab Initio and Semiempirical Linear Combination of Atomic Orbital Wave Functions”. In: *The Journal of Chemical Physics* 90.9 (May 1989), pp. 4916–4926. DOI: 10.1063/1.456588.

[29] Claudia Filippi and C. J. Umrigar. “Correlated Sampling in Quantum Monte Carlo: A Route to Forces”. In: *Physical Review B* 61.24 (June 2000), R16291–R16294. DOI: 10.1103/PhysRevB.61.R16291.

[30] James Martens and Roger Grosse. “Optimizing neural networks with kronecker-factored approximate curvature”. In: *International conference on machine learning*. PMLR. 2015, pp. 2408–2417.

[31] Armağan Kinal and Piotr Piecuch. “Computational Investigation of the Conrotatory and Disrotatory Isomerization Channels of Bicyclo[1.1.0]butane to Buta-1,3-diene: A Completely Renormalized Coupled-Cluster Study”. In: *The Journal of Physical Chemistry A* 111.4 (Feb. 2007). Publisher: American Chemical Society, pp. 734–742. ISSN: 1089-5639. DOI: 10.1021/jp065721k. URL: https://doi.org/10.1021/jp065721k.

[32] Matthias Rupp et al. “Fast and Accurate Modeling of Molecular Atomization Energies with Machine Learning”. In: *Phys. Rev. Lett.* 108 (5 Jan. 2012), p. 058301. DOI: 10.1103/PhysRevLett.108.058301.

[33] Kristof T. Schütt et al. “SchNet: A Continuous-Filter Convolutional Neural Network for Modeling Quantum Interactions”. In: (June 2017). DOI: 10.48550/arXiv.1706.08566.

[34] Gabriel Pescia et al. *Message-Passing Neural Quantum States for the Homogeneous Electron Gas*. May 2023. DOI: 10.48550/arXiv.2305.07240.

[35] Qiming Sun et al. “Recent Developments in the PySCF Program Package”. In: *The Journal of Chemical Physics* 153.2 (July 2020), p. 024109. DOI: 10.1063/5.0006074.

[36] Mario Geiger and Tess Smidt. *E3nn: Euclidean Neural Networks*. July 2022. arXiv: 2207.09453 [cs].

[37] Diederik P. Kingma and Jimmy Ba. “Adam: A Method for Stochastic Optimization”. In: arXiv:1412.6980 (Jan. 2017). DOI: 10.48550/arXiv.1412.6980. arXiv: 1412.6980 [cs].

[38] Greg Landrum. *RDKit: Open-source cheminformatics*. https://github.com/rdkit/rdkit. GitHub repository. 2009.

[39] Frank Neese et al. “The ORCA Quantum Chemistry Program Package”. In: *The Journal of Chemical Physics* 152.22 (June 2020), p. 224108. DOI: 10.1063/5.0004608.

[40] David Pfau James S. Spencer and FermiNet Contributors. *FermiNet*. 2020. URL: http://github.com/deepmind/ferminet.

[41] Aleksandar Botev and James Martens. *KFAC-JAX*. Version 0.0.1. 2022. URL: http://github.com/deepmind/kfac-jax.

Supplementary material for Variational Monte Carlo on a Budget – Fine-tuning pre-trained Neural Wavefunction

A Electron MCMC initialization

To investigate the impact of the initial distribution of electron positions on the equilibration of the Markov Chain, we run two evaluations for a glycine molecule, using a pre-trained wavefunction. We perform no initial burn-in and use every 50th sample for energy evaluation. If the chain was perfectly equilibrated right after initialization, all sampled energies would fluctuate around the mean energy. However as Fig. 6b shows, it takes several thousand steps for the sampled energies to converge to the correct mean. This is particularly pronounced with Gaussian initialization of electron positions, which is the default in state-of-the-art DL-VMC codes such as FermiNet [3]. Using an exponential distribution of the initial electron positions much more closely resembles the correct electron density ψ^2 (cf. Fig. 6a) and thus reaches equilibrium substantially faster.

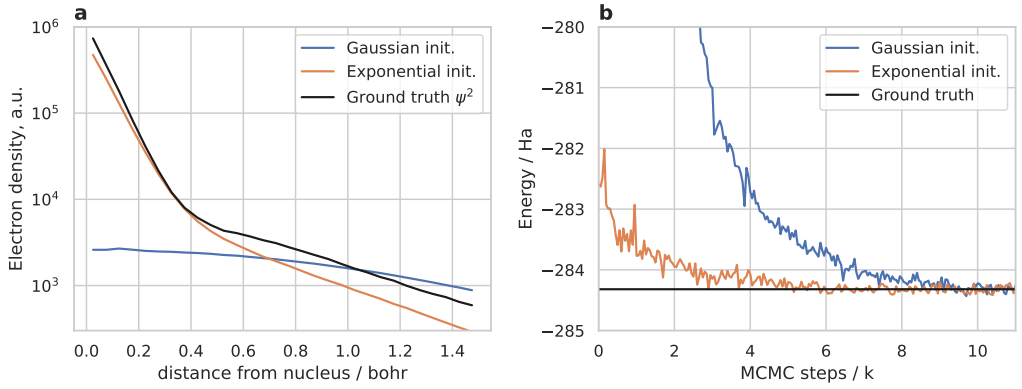

Figure 6: **Effect of electron initialization**: Initializing the electron positions using an exponential distribution instead of Gaussian, better fits the actual density (a), and thus leads to faster equilibration of observables during evaluation (b).

B Orbital localization

Our model uses orbital embeddings $\boldsymbol{x}^{\text{orb}}$ as inputs to parameterize the backflows $\boldsymbol{f}^{\text{orb}}$, and exponents $\boldsymbol{g}^{\text{orb}}$ of the orbitals. These orbital embeddings were introduced by Scherbela et al. [25] in the form of molecular orbital expansion coefficients, obtained from a self consistent Hartree-Fock calculation. In this setting, the coefficients $\boldsymbol{x}^{\text{orb}}$ are not uniquely defined, but only up to a linear transformation U with determinant ± 1

$$\boldsymbol{x}^{\text{orb}}_{Ik} = \sum_{n=1}^{N_{\text{orb}}} U_{kn} \hat{\boldsymbol{x}}^{\text{orb}}_{Ik}, \qquad U \in \mathbb{R}^{N_{\text{orb}} \times N_{\text{orb}}}, \qquad \det U = \pm 1. \tag{15}$$

This stems from the fact that the corresponding Hartree-Fock wavefunction is invariant under such a transformation. Consequently there is free choice, which linear combination of embeddings $\boldsymbol{x}^{\text{orb}}$ to choose from without any loss of information. We follow the approach of [25], by choosing U such that the corresponding Hartree-Fock orbitals are maximally localized according to the Foster-Boys

metric, i.e. minimize the spatial variance $\mathcal{L}$:

$$\phi_k(\boldsymbol{r}, U) = \sum_{I=1}^{N_{\text{nuc}}} \sum_{\mu=1}^{N_{\text{basis}}} b_{I\mu}(\boldsymbol{r}) U_{kn} \hat{x}^{\text{orb}}_{In\mu} \tag{16}$$

$$\mathcal{L}(U) = \sum_k \int \phi_k^2(\boldsymbol{r}, U) \boldsymbol{r}^2 d\boldsymbol{r} - \left(\int \phi_k^2(\boldsymbol{r}, U) \boldsymbol{r} d\boldsymbol{r} \right)^2 \tag{17}$$

Here $b_{I\mu}(\boldsymbol{r})$ denotes μ-th basis function of the Hartree-Fock expansion, centered on the I-th nucleus. In practice the integrals of Eq. 17 do not have to be evaluated explicitly, but can instead be computed via the overlap matrix $\boldsymbol{S}$. The minimization of $\mathcal{L}$ is typically done iteratively, requires on the order of 10 steps, and is readily implemented in many open-source quantum chemistry codes such as pySCF [35].

C Adaption of PhisNet

We heavily rely on PhisNet by Unke et al. [20] to obtain orbital descriptors without the need for a separate SCF calculation. Compared to their original work, we made several simplifications, which are motivated by the fact that we do not predict final high-accuracy orbitals in a large basis set, but only use PhisNet as a feature extractor by predicting orbitals in a minimal basis-set:

- **Layer Norm** We found deep variants of PhisNet to be unstable to train and mitigated the issue by adding an (equivariant) layer norm after each PhisNet module.
- **Simplified Fock matrix prediction** The original PhisNet implementation uses a final interaction between the node embeddings, before predicting the elements of the Fock matrix. We found this interaction to be superfluous for our purposes and left it out for simplicity.
- **Separate energy head** The original PhisNet computes energies via the eigenvalues obtained by diagonalization of the Fock matrix. We instead predict energies using a separate head on top of the scalar features of the node embeddings.
- **Smaller network** We changed the hyperparameters to obtain a smaller and faster version of PhisNet which obtained sufficient accuracy for our purposes. We used 2 layers (instead of 5) and $L_{\text{max}} = 2$ (instead of 4). This reduces the number of parameters from 17M to 3M.
- **Diverse training set** While the original work optimized separate models for each molecule (e.g. by training on different geometries of a molecular dynamics simulation), we optimize a single model to predict $\boldsymbol{F}$, $\boldsymbol{S}$, $\boldsymbol{E}$, and $\boldsymbol{\nabla E}$ across a dataset of 47k geometries sampled from QM7-X [26].
- **JAX re-implementation** We re-implemented PhisNet in JAX, using the e3nn library [36] to construct the SE(3)-equivariant operations.

We train the PhisNet-model on a dataset of 47k molecules from QM7X [26], using the Adam optimizer [37] on the following loss

$$\mathcal{L} = \sum_n \left(E^{\text{phis}}(\boldsymbol{R}^n, \boldsymbol{Z}^n) - E^{\text{ref},n} \right)^2 + \tag{18}$$

$$+ \sum_{nI\zeta} \left(\frac{\partial}{\partial R^n_{I\zeta}} E^{\text{phis}}(\boldsymbol{R}^n, \boldsymbol{Z}^n) - G^{\text{ref},n}_{I\zeta} \right)^2 + \tag{19}$$

$$+ \sum_{nIJ\mu\nu} \left(F^{\text{phis}}_{IJ\mu\nu}(\boldsymbol{R}^n, \boldsymbol{Z}^n) - F^{\text{ref},n}_{IJ\mu\nu} \right)^2 + \tag{20}$$

$$+ \sum_{nIJ\mu\nu} \left(S^{\text{phis}}_{IJ\mu\nu}(\boldsymbol{R}^n, \boldsymbol{Z}^n) - S^{\text{ref},n}_{IJ\mu\nu} \right)^2 . \tag{21}$$

Here E denotes energies, G denotes gradients of energies, F Fock matrices, and S overlap matrices. The indices I, J run over nuclei, the indices μ, ν over basis functions, and the index n over samples in a batch.

D Molecule datasets

Bicyclobutane For the Bicyclobutane to 1,3-butadiene transition we use the geometries from Kinal et al. [31] and compare against the reference energies stated in Spencer et al. [5].

N_2 For the N_2 potential energy surface with various bond-lengths we used the geometries including reference calculations from [8].

Propadiene The global rotation of 360° degrees for propadiene is performed on the geometry which is part of the test set for 3 heavy atoms. For the torsion experiment we used the equilibrium geometry and rotated the torsion angle by 90° degrees in steps of 10° degrees.

Zero-shot and fine-tuning dataset The results on zero-shot and few-shot predictions for increasing number of heavy atoms are performed on random subsets of molecules. For 5-7 heavy atoms we sample 4 unique and distorted molecules from QM7-X [26]. For 4 heavy atoms we use all geometries from the Bicyclobutane dataset. For 3 heavy atoms we use the ablation dataset.

Ablation dataset For the ablation study, we use one geometry per molecule from the out-of-distribution test set from Scherbela et al. [25], leading to a set of four distinct molecules. We ensure that these molecules are not part of the training set.

Large scale experiment For the large scale experiment we used a stratified random sample of 250 molecules from QM7 [32]. It contains all molecules with up to 4 heavy atoms, and additionally 65 randomly chosen molecules for 5, 6 and 7 heavy atoms each.

Pre-training dataset for transferable neural wavefunctions To train our pre-trained wavefunctions we use two datasets, consisting of 18 and 98 disparate molecules. For part of the ablation we use the dataset proposed in [25] and an extended version with 80 additional molecules. The additional compounds are a combination of all valid SMILES generated with RDKit [38] with 3 heavy atoms, allowing only Nitrogen, Oxygen and Carbon with single-, double- or triple-bonds, and all molecules up to four heavy atoms from QM7-X [26] (excluding molecule containing Fluorine). To prevent a train-test leakage, we remove Bicyclobutane (including all conformations) and the four molecules from the ablations dataset. Since the normal-mode-distortions by design do not generate strongly distorted geometries, we augment the 98-molecule-dataset with rotated dihedral angles. To generate a subset of all possible dihedral angles for a heavy-atom bond we first generate samples with equidistant angles for all possible dihedral angles and compute Hartree-Fock energies with a minimal basis-set. We include the equilibrium geometry and all extrema of the potential energy surface with respect to the rotation of a single dihedral angle if the energy of the extrema is significantly different to already included geometries of the same molecule. Additionally, we include the transition geometry towards the respective extrema and again only include energetic diverse states. Finally, to make sure that certain molecules are not underrepresented in the dataset we make sure that all molecules have at least 5 geometries that get distorted during pre-training by adding copies of the equilibrium geometry. Overall this yields 699 initial geometries $\boldsymbol{R}^0$ for pre-training.

E Energy variance

Fig. 7 depicts the energy variance obtained by various models. Analogously to the the energies presented in Fig. 2 of the main text, we also find that our pre-trained model achieves substantially lower energy variance compared to previous generalizing wavefunctions. Notably on the test set with 3 heavy atoms, the variance of our zero-shot model is on par with the variance of a PsiFormer model trained for 16k steps.

F Relative energies

To better depict the accuracy of the models' relative energies for the test systems in Fig. 3 of the main text, we depict the difference of relative energies between our model and a references method in Fig. 8. Furthermore we list numerical values vor all energies of Fig. 3a in Tab. 1. We find that on these

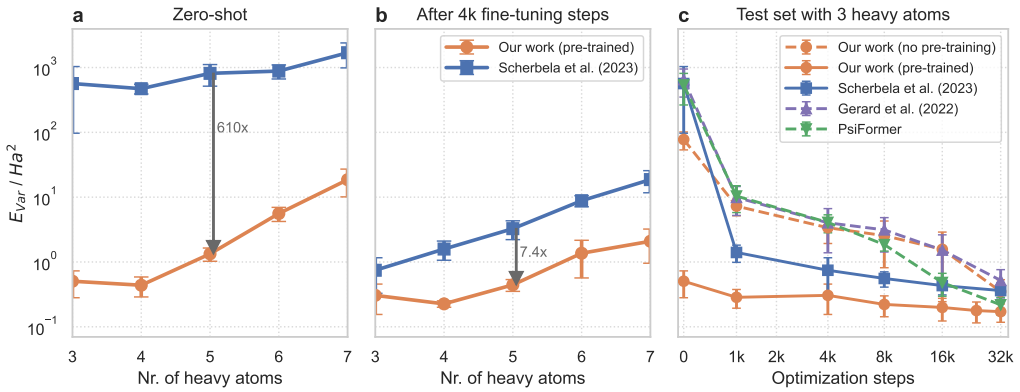

Figure 7: **Variance of energies**: Variance when re-using the pre-trained model on molecules of varying size without optimization (a) and after fine- tuning (b). For the test set containing 3 heavy atoms as a function of optimization steps, comparing against current state-of-the-art methods. Solid lines correspond to pre-trained models, dashed lines to models without pre-training (c).

challenging systems our model yields relative energy errors of up to 15 mHa when only fine-tuned for 400 steps per geometry. When fine-tuning our pre-trained model for 3200 steps per geometry, we obtain relative energies that are accurate within 2-3 mHa.

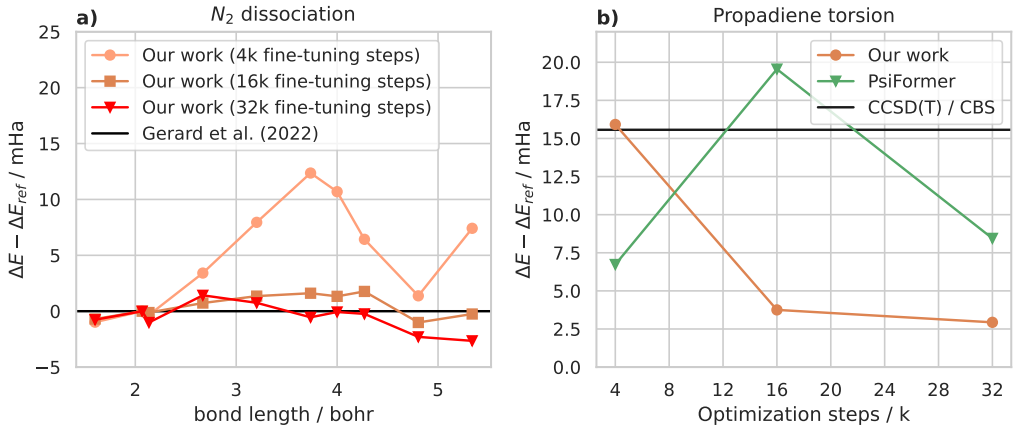

Figure 8: **Challenging relative energies**: The difference between relative energies of a reference method vs our work for two challenging systems. **a) Nitrogen dimer:** Difference of relative energies to the equilibrium geometry for our approach to reference calculation from Gerard et al. for N_2, **a) Propadiene:** Difference of relative energies of the transition barrier of twisted vs. untwisted propadiene. As reference method we choose PsiFormer with 64k optimization steps. This figure complements Fig. 3b,d of the main text with results for additional fine-tuning steps.

G Reference energies

CCSD(T) All CCSD(T) energies – except explicitly stated otherwise – were obtained using ORCA [39] starting from a restricted Hartree-Fock calculation. We use correlation consistent basis sets of the cc-pCVXZ family, with X in {2, 3, 4}. To extrapolate to the complete basis set limit (CBS), we use

structure	CCSD(T) [31]	DMC [31]	FermiNet 200k [5]	FermiNet 10k [5]	Our work zero-shot	Our work 700 per geom
con_TS	64.4	64.4	64.1	63.9	94.0	66.6
dis_TS	34.7	93.4	92.0	87.1	183.8	94.5
g-but	-40.0	-40.2	-40.3	-44.9	-48.5	-40.4
gt-TS	-35.5	-35.4	-35.9	-42.9	-46.9	-36.7
t-but	-44.6	-44.5	-45.3	-47.5	-51.8	-43.2

Table 1: Energies relative to the energy of bicyclobutane in mHa, including the zero-point vibrational energy correction from Kinal et al. [31]. This data complements Fig. 3a in the main text.

the approach outlined in [3] and fit the following functions with free parameters $E_{\text{CBS}}^{\text{HF}}, E_{\text{CBS}}^{\text{corr}}, a, b, c$:

$$E_X^{\text{HF}} = E_{\text{CBS}}^{\text{HF}} + ae^{-bX}$$
$$E_X^{\text{corr}} := E_X^{\text{HF}} - E_X^{\text{CCSD(T)}} = E_{\text{CBS}}^{\text{corr}} + cX^{-3}$$
$$E_{\text{CBS}}^{\text{CCSD(T)}} = E_{\text{CBS}}^{\text{HF}} + E_{\text{CBS}}^{\text{corr}}$$

We stress that although CCSD(T)-energies are often considered as "gold-standard", they do not necessarily represent the actual ground-state energy. There are many cases, where CCSD(T) either overestimates the true ground-state energy, or even underestimates it, because CCSD(T) does not yield upper bounds to the true ground-state energy.

PsiFormer For Fig. 2 we used the open-source FermiNet codebase [40]. The codebase didn't allow for inference calculation, therefore a slight fix was applied. All calculations were performed with the small settings as proposed in von Glehn et al. [6].

H Hyperparameters

A detailed description of the hyperparameter used in this work can be found below (cf. Tab. 2). For the mapping of the orbital descriptors to the electron embeddings to build the orbitals we rely on the hyperparameter from [25]. For optimization we rely on the second-order method KFAC [30] and use their Python implementation [41]. During the continuous sampling of the geometries we allow each geometry to perform a maximum of 20 steps of normal-mode distortion from the initial geometry and reset to the original one once the threshold is reached.

I Runtime and computational resources

Tab. 3 lists the run-times for our method and a comparable conventional reference method (CCSD(T)-4Z) for a typical molecule with 5 heavy atoms. We find that our fine-tuning our model for 4k steps yields lower absolute energies than CCSD(T)-4Z (cf. Fig. 2) at similar computational cost. This is in contrast to many earlier works that achieve highly accurate energies, but at computational cost that far surpasses the cost of conventional methods for small molecules.
Overall we used $\approx$ 5k GPUhs (A100) for development and training of our base models, and another 5k GPUhs (A40) on evaluations and fine-tuning. Additionally we required $\approx$ 20k CPUhs for CCSD(T) reference calculations.

Method	Runtime / h
Our work, zero-shot	1.0
Our work, 4k fine-tuning	9.5
PsiFormer, 16k steps	53.0
CCSD(T), 4Z basis	10.5

Table 3: **Runtime** of various quantum chemistry methods for C_4OH_6. All timings are in node hours, being 2 A100-hours for our work and PsiFormer, and 256 CPU-hours for CCSD(T).

J Code and data availability

All code, configuration files, geometries, datasets and obtained energies are available on GitHub under `https://github.com/mdsunivie/deeperwin`. Model weights are available on figshare under `https://doi.org/10.6084/m9.figshare.23585358.v1`.

Electron Embedding	Hidden dimension N_{emb}	256
	№ iterations	4
Nuclear Embedding	Hidden dimension $\tilde{\boldsymbol{x}}^{nuc}$	64
	№ layer MLP	1
Message passing	Activation function	SiLU
	№ layer edge embedding	3
	Dimension edge embedding	64
	Dimension linear layer	32
Markov Chain Monte Carlo	№ walkers	2048
	№ decorrelation steps	50
	Target acceptance prob.	50%
PhisNet [20]	Pre-trained against basis set	STO-6G
	№ iterations	2
	Harmonic degree L	2
	№ radial basis functions	128
	Hidden dimension of $\boldsymbol{x}^{nuc}$	128
	Distance cutoff (bohr)	30
Transferable atomic orbitals [25]	№ determinants N_{det}	8
	№ hidden layers $\boldsymbol{f}^{\text{orb}}$	2
	Hidden dimension of $\boldsymbol{f}^{\text{orb}}$	256
	№ hidden layers $\boldsymbol{g}^{\text{orb}}$	2
	Hidden dimension $\boldsymbol{g}^{\text{orb}}$	128
	№ iterations MPNN	2
	№ radial basis functions	16
	Hidden edge embedding dimension	32
	Hidden node embedding dimension	16
	Activation function	SiLU
Variational pre-training	Optimizer	KFAC
	Batch size	2048
	Norm constraint	3×10^{-3}
	Initial damping d_0	1
	Minimal damping d_{min}	0.001
	Damping rate decay	$\text{d}(t) = \text{d}_0 \exp(-t/20000)$
	Initial learning rate lr_0	0.1
	Learning rate decay	$\text{lr}(t) = \text{lr}_0(1 + t/6000)^{-1}$
	Optimization steps	128,000 - 256,000
Changes for fine-tuning	Learning rate decay	$\text{lr}(t) = \text{lr}_0(7 + t/6000)^{-1}$
	Optimization steps	0 - 32,000
Sampling geometries	Distortion energy β	0.005 Ha
	Max age	20
	Bias towards original geometry α	0.2

Table 2: Hyperparameter settings used in this work

Transferable Neural Wavefunctions for Solids

L. Gerard[†*], M. Scherbela[†*], H. Sutterud[‡*], W.M.C. Foulkes[‡], and P. Grohs[†, ¶]

[†] *Faculty of Mathematics, University of Vienna, Oskar-Morgenstern-Platz 1, A-1090 Vienna, Austria*
[‡] *Department of Physics, Imperial College London, South Kensington Campus, London SW7 2AZ*
[¶] *Johann Radon Institute for Computational and Applied Mathematics, Austrian Academy of Sciences, Altenbergerstrasse 69, 4040 Linz, Austria*
[*] *These authors contributed equally*

Abstract

Deep-Learning-based Variational Monte Carlo (DL-VMC) has recently emerged as a highly accurate approach for finding approximate solutions to the many-electron Schrödinger equation. Despite its favorable scaling with the number of electrons, $\mathcal{O}({n_{\text{el}}}^4)$, the practical value of DL-VMC is limited by the high cost of optimizing the neural network weights for every system studied. To mitigate this problem, recent research has proposed optimizing a single neural network across multiple systems, reducing the cost per system. Here we extend this approach to solids, where similar but distinct calculations using different geometries, boundary conditions, and supercell sizes are often required. We show how to optimize a single ansatz across all of these variations, reducing the required number of optimization steps by an order of magnitude. Furthermore, we exploit the transfer capabilities of a pre-trained network. We successfully transfer a network, pre-trained on $2 \times 2 \times 2$ supercells of LiH, to $3 \times 3 \times 3$ supercells. This reduces the number of optimization steps required to simulate the large system by a factor of 50 compared to previous work.

1 Introduction

Many interesting material properties, such as magnetism and superconductivity, depend on the material's electronic structure as given by the ground-state wavefunction. The wavefunction may in principle be found by solving the time-independent Schrödinger equation, but doing so with sufficient accuracy is challenging because the computational cost grows dramatically with the number of particles. The challenge is particularly pronounced in solid state physics, where accurate calculations for periodic systems require the use of large supercells — and consequently many particles — to minimize finite-size effects.

Over the past few decades, density functional theory (DFT) has emerged as the primary workhorse of solid state physics. When using local or semi-local exchange-correlation functionals such as LDA or PBE [1], DFT calculations have a favorable scaling of $\mathcal{O}({n_{\text{el}}}^3)$ or better, where n_{el} is the number of electrons in the system, and an accuracy that is often sufficient to help guide and predict experiments [1, 2]. However, the choice of functional is in practice an uncontrolled approximation and DFT sometimes yields quantitatively or even qualitatively wrong results, especially for strongly correlated materials [3, 4].

Another approach, known as variational Monte Carlo (VMC), uses an explicit parameterized representation of the full many-body wavefunction and optimizes the parameters using the variational principle. This method has a favorable scaling of $\mathcal{O}({n_{\text{el}}}^{3-4})$ [5, 6] but is limited in accuracy by the expressivity of the ansatz used. Recently, deep neural networks have been employed as wavefunction ansätze [6–8] and used to study a large variety of systems including small molecules [6, 9, 10], periodic model systems described by lattice Hamiltonians [7, 11–13], the homogeneous electron gas [14, 15], and Fermi liquids [16, 17]. Thanks to their flexibility and expressive power, deep-learning-based VMC (DL-VMC) approaches provide the best current estimates for the ground-state energies of several small molecules [9, 10]

Efforts to apply DL-VMC to real solids [18, 19] have been limited by the high computational cost involved. While a single calculation may be feasible, studying real solids requires many similar but distinct calculations. First, it is necessary to perform calculations involving increasingly larger supercells to estimate finite-size errors and extrapolate results to the thermodynamic limit. Second, twist-averaged boundary conditions (TABC) are used to accelerate the rate at which the finite-size errors reduce as the supercell size increases [20]. This requires averaging the results for each supercell over many calculations using different boundary conditions. Lastly, studying a given system often requires calculations for different geometries and lattice constants. Since most existing DL-VMC ansätze require optimizing a new wavefunction from scratch for each new system, the computational cost quickly becomes prohibitive even for systems of moderate size. For example, Li et al. proposed DeepSolid [18], an ansatz that can accurately model periodic wavefunctions with up to 100 electrons, but required over 80k GPU hours to study a single system.

In this work we implement a transferable DL-VMC ansatz for real solids that takes as input not only the electron positions but also other parameters of the system, such as its geometry or boundary condition. The key idea, based on [21] and detailed in Sec. 4.3, is to map computationally cheap, uncorrelated mean-field orbitals to expressive neural network orbitals that depend on the positions of all of the electrons. The transferability of the network orbitals allows us to optimize a single ansatz for many variations of unit-cell geometry, boundary condition and supercell size all at once. Because the ansatz learns to generalize across systems, we can use pre-trained models as highly effective initializers for new systems or larger supercells.

Compared to previous DL-VMC work without transferability, our approach yields more accurate results, gives access to denser twist averaging (reducing finite-size effects), and requires a fraction of the computational resources. For example, for lithium hydride, transferring a 32 electron calculation to one with 108 electrons yields more accurate results than previous work [18] at $\approx 1/50$ of the computational cost.

This paper is structured as follows. Sec. 2 describes the results of applying the transferable DL-VMC ansatz to three different systems: 1D hydrogen chains, 2D graphene, and 3D lithium hydride. In Sec. 3 we discuss the implications of the results, limitations of the ansatz, and possible future work. Sec. 4 explains the DL-VMC approach and our ansatz, as well as other technical details of our work.

2 Results

2.1 1D: Hydrogen chains

Chains of hydrogen atoms with periodic boundary conditions provide a simple 1D toy system that nevertheless exhibits rich physics such as dimerization, a lattice-constant-dependent metal-insulator transition, and strong correlation effects. A collaborative effort [3, 22] has obtained results for this system using a large variety of high-accuracy methods, providing a trustworthy benchmark.

Energy per atom The first test is to obtain the total energy per atom for a fixed atom spacing, $R = 1.8a_0$ (where a_0 is the Bohr radius), in the thermodynamic limit (TDL) attained as the number of atoms in the supercell tends to infinity. To this end, we train two distinct models on periodic supercells with $N_{\text{atoms}} = 4, 6, \ldots, 22$. The first model is trained at twist $k = 0$ (the Γ point) only. The second is trained using all twists from a Γ-centered four-point Monkhorst-Pack grid [23]. The three inequivalent twists are $k = 0, \frac{1}{4}, \frac{1}{2}$ in units of $2\pi/R$, and their weights are $w = 1, 2, 1$, respectively. Once the model has been pre-trained on these relatively short chains, we fine-tune it on larger chains with $N_{\text{atoms}} = 32, 38$. We use the extrapolation method described in [22] to obtain the energy E_∞ in the TDL. This entails fitting a polynomial of the form $E = E_\infty + E_1 N_{\text{atoms}}^{-1} + E_2 N_{\text{atoms}}^{-2}$. Previous authors have extrapolated the energy using only chain lengths of the form $N_{\text{atoms}} = 4n + 2,\ n \in \mathbb{N}$, which have filled electronic shells. We also report extrapolations using chain lengths $N_{\text{atoms}} = 4n$, which lead to partially filled shells.

Fig. 1a shows that all of our extrapolations (Γ-point filled shells, Γ-point unfilled shells, and TABC) are in good qualitative agreement with previous results obtained using methods such as lattice-regularized diffusion Monte Carlo (LR-DMC) [22] and DeepSolid [18], a FermiNet-based [6] neural wavefunction for solids. Quantitatively, we achieve slightly lower (and thus, by the variational principle, more accurate) energies than DeepSolid for all values of N_{atoms}. Using TABC, we obtain $E_\infty = -565.24(2)\,\text{mHa}$, which is $0.2 - 0.5\,\text{mHa}$ lower than the estimate obtained using LR-DMC and DeepSolid, and agrees within uncertainty with the extrapolated energy computed using the auxiliary-field quantum Monte Carlo (AFQMC) method [22]. Most notably, though, we obtain these results at a fraction of the computational cost of DeepSolid. Whereas DeepSolid required a separate calculation with 100,000 optimization steps for each value of N_{atoms} (and would have required even more calculations for twist-averaged energies), we obtain results for all 10 chain lengths and values of $N_{\text{atoms}} = 4, \ldots, 22$, with 3 twists for each system, using only 50,000 optimization steps in total. Furthermore, by re-using the model pre-trained on smaller chains, we obtain results for the larger chains with $N_{\text{atoms}} = 32, 38$ using only 2,000 additional steps of fine tuning. This reduces the cost of simulating the large chains by a factor of approximately 50. We note that, as expected, the use of TABC reduces the finite-size errors, allows to combine results for filled and unfilled shells in the extrapolation, and leads to faster and more uniform convergence of the energy per atom.

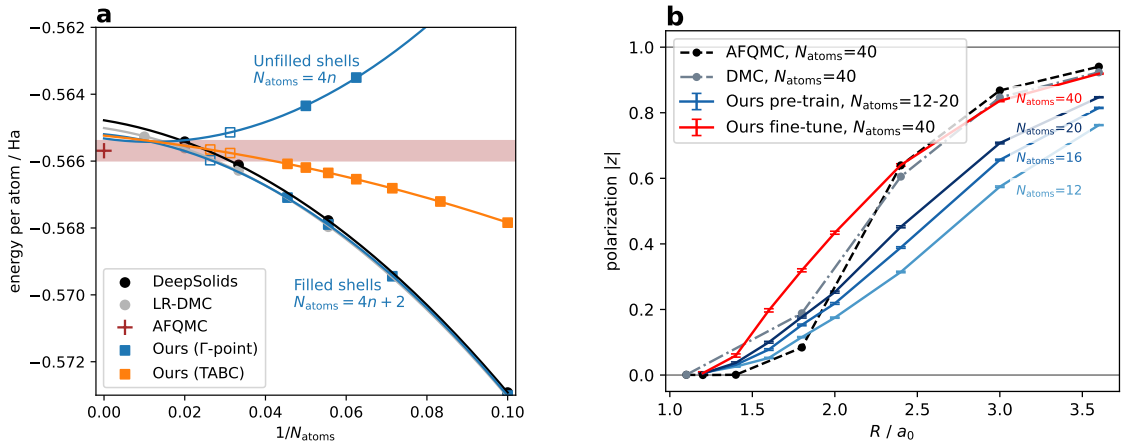

Figure 1: **1D Hydrogen chain**: **a**: Extrapolation of the energy per atom to the thermodynamic limit for $R = 1.8a_0$. Results obtained using DeepSolid (neural wavefunction), lattice-regularized diffusion Monte Carlo (LR-DMC), auxiliary field Monte Carlo (AFQMC), and our transferable neural wavefunction are shown. Open markers indicate energies computed by fine-tuning a model pre-trained on smaller supercells. The shaded area depicts the statistical uncertainty in the AFQMC result. Monte Carlo uncertainty of our results is $\approx 10\mu$Ha, well below the marker size. **b**: The complex polarization $|z|$ as a function of the inter-atomic separation, R, showing a phase transition between a metal at small R and an insulator at large R. AFQMC, DMC, and VMC results are taken from [3]. DeepSolid results are taken from [18].

Metal-insulator transition The 1D hydrogen chain exhibits a transition from an insulating phase at large inter-atomic separation, R, to a metallic phase at small R. The transition can be quantified by evaluating the complex polarization along the length of the chain

$$z = \left\langle e^{i \frac{2\pi}{R N_{\text{atoms}}} \sum_{i=1}^{n_{\text{el}}} x_i} \right\rangle, \tag{1}$$

where x_i is the position of electron i in the direction of the chain. The expectation value is defined as $\langle \ldots \rangle \equiv \int \Psi^*(\boldsymbol{r}) \ldots \Psi(\boldsymbol{r})\, \mathrm{d}\boldsymbol{r}$, where $\boldsymbol{r} = (\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots, \boldsymbol{r}_{n_{\text{el}}})$ is a $3n_{\text{el}}$-dimensional vector of electron positions, Ψ is the (approximate) ground-state wavefunction, and the integral is over all $3n_{\text{el}}$ electronic degrees of freedom. Although the polarization is easy to evaluate in principle, studying the transition is computationally costly because it requires many similar but distinct calculations:

multiple values of R are required to locate the transition; multiple twists k are required to obtain accurate twist-averaged polarizations; and multiple chain lengths N_{atoms} are required to allow extrapolation to the TDL. Even for a modest selection of all of these variations, studying the phase transition in detail requires hundreds of calculations. Using our transferable wavefunction, on the other hand, allows us to train a single model to represent the wavefunction for all parameter variations at once.

We trained a single ansatz to describe all 120 combinations of: (a) 3 distinct chain lengths, $N_{\text{atoms}} = 12, 16, 20$; (b) 5 symmetry-reduced k-points of an 8-point Γ-centered Monkhorst-Pack grid; and (c) 8 distinct atom spacings between $R = 1.2a_0$ and $R = 3.6a_0$. A total of 200k optimization steps were carried out, after which the complex polarization was evaluated using Eq. (1). To improve our estimates for $N_{\text{atoms}} \to \infty$, we fine-tuned this pre-trained model for 2k steps on chain lengths of $N_{\text{atoms}} = 40$ and a denser 20-point Monkhorst-Pack grid containing 11 symmetry-reduced twists. Fig. 1b shows that our approach qualitatively reproduces the results obtained using DMC and AFQMC. In agreement with Motta et al. [3], we observe a second-order metal-insulator transition. However, where Motta estimates the critical atom spacing $R_{\text{crit}} = 1.70(5)a_0$, our results are more consistent with $R_{\text{crit}} = 1.35(5)a_0$. A possible explanation for the disagreement is that our neural wave function may be less accurate (and may therefore produce relatively higher energies) for metals than insulators, disfavoring the metallic phase. Another possible explanation follows from the observation that, unlike the VMC method used here, the DMC and AFQMC methods yield biased estimates of the expectation values of operators, such as the complex polarization, that do not commute with the Hamiltonian [5, 24].

Also in agreement with Motta et al. [3], we find that the hydrogen chain has an antiferromagnetic ground state at large lattice constant R. The expected atomic spins are zero on every atom, but the spins on neighbouring atoms are antiferromagnetically correlated. As the lattice constant gets smaller and the system transitions to the metallic phase, these correlations decrease.

2.2 Graphene

To demonstrate the application of our transferable DL-VMC ansatz to a two-dimensional solid, we compute the cohesive energy of graphene in a 2×2 supercell and compare against the DL-VMC results of Li et al. [18]. We employ twist-averaged boundary conditions, apply structure-factor-based finite-size corrections [25] as detailed in the methods section, and add zero-point vibrational energies (ZPVE) (see Sec. 4.7). Li et al. restricted their calculation to a Monkhorst-Pack grid of 3×3 twists, yielding three symmetry-reduced twists in total. In our case, since we are able to compute multiple twists at once with minimal extra cost, we increase the grid size to 12×12. This increases the number of symmetry-reduced twists approximatly 6 times, from 3 to 19. The larger twist grid contains a subset of the twists considered by Li et al., allowing a direct comparison with their independent energy calculations.

In Tab. 1 (see also Sec. S4 for separate twist energies without ZPVE), we compare total energies calculated at each of the three twists on the 3×3 twist grid and cohesive energies obtained by averaging over the 3×3 and 12×12 twist grids. We find that our energies for k_1 and k_2 are lower than the energies obtained by DeepSolid by 1 mHa and 7 mHa / primitive cell respectively, while our energy for k_3 is higher by 4 mHa. Overall this leads to a twist-averaged energy which is 4 mHa / primitive cell lower than the DeepSolid energy. This is a direct consequence of how our approach divides the total number of optimization steps across twists. At every optimization step we randomly sample a symmetry-reduced twist to optimize next. The sampling probability is obtained by normalizing the weights assigned to the twists on the symmetry-reduced 12×12 Monkhorst-Pack grid. This procedure ensures that more optimization steps are spent on twists with high contribution to the final energy and fewer steps on less important twists. The k_3 twist is chosen in around $\sim$2.0% of all optimization steps, whereas the second twist is chosen two times more often. The wave function at k_2 is therefore optimized more stringently.

Furthermore, we obtain energies not only for the 3×3 Monkhorst-Pack-grid, but also for the full 12×12-grid, allowing us to assert convergence with respect to the k-point density. We stress that we only required a single neural network, optimized for 120k steps to obtain energies for all twists

(both the 12×12-grid and the 3×3-subset). DeepSolid on the other hand optimized for 900k steps in total, obtaining energies only for the 3×3-twist-grid.

We compare against experimental cohesive energies obtained from thermochemistry data for graphite [26] corrected for the small inter-layer binding energy of 3.5 mHa obtained using the Random Phase Approximation [27]. When computing cohesive energies and correcting for finite-size effects using a structure-factor-based correction and ZPVE (see Sec. 4.7) we obtain energies that are 7 mHa lower than experimental values, i.e. we predict slightly stronger binding than experiment. We hypothesize that this small discrepancy may be a finite size artifact stemming from the relatively small 2x2 supercell.

Table 1: **Total and cohesive energies of graphene** in Hartrees. The upper block of the table compares our results against the total energies computed by Li et al. [18] at the three symmetry-inequivalent twists on the 3×3 Monkhorst-Pack grid. The twists are expressed in the basis of the reciprocal lattice vectors. The lower block compares the twist-averaged cohesive energy per primitive cell with experimental results, showing the effect of increasing the size of the twist grid from 3×3 to 12×12. For the calculation of the cohesive energy we follow Li et al. [18] and take as the energy of a single carbon atom E= -37.84471 Ha [6]. All results include a structure-factor-based finite-size correction [25] and ZPVE (see Sec. 4.7). The experimental results are based on [26, 27].

	Twists	**Weight**	**DeepSolid**	**Our work**	**Exp.**
Total energy / Ha	$\boldsymbol{k}_1 = (0,\ 0)$	1/9	-76.1406	$-76.1414(2)$	-
	$\boldsymbol{k}_2 = (1/3, 1/3)$	2/3	-76.2342	$-76.2415(2)$	-
	$\boldsymbol{k}_3 = (2/3, 1/3)$	2/9	-76.2479	$-76.2432(2)$	-
Cohesive energy / Ha	Averaged 3×3	-	-0.5375	$-0.5413(2)$	$-0.538(2)$
	Averaged 12×12	-	-	$-0.5451(2)$	

With a network that has been trained across the entire Brillouin zone, we can evaluate observables along arbitrary paths in k space. Fig. 2 is a bandstructure-like diagram, showing how the total energy varies along a path passing through the high-symmetry k points $\Gamma = (0, 0)$, $M = (0, 1/2)$, and $K = (1/3, 2/3)$ in units of the supercell reciprocal lattice vectors. We use the pre-trained model from the 12x12 Monkhorst-Pack grid and transfer it to the bandstructure-like diagram with k-points previously unseen during optimization, requiring only a few additional optimization steps. We fine-tune the pre-trained model for the k-points on the path, using around 100 optimization steps per twist and then evaluate the energies along the path. Analogously to the Dirac cone visible in the one-electron bandstructure, also our many-electron bandstructure displays a characteristic cusp at the K point. However, since we are plotting the dependence of an n_{el}-electron energy on a many-body parameter (the twist), Fig. 2 is not directly comparable to a conventional one-electron bandstructure diagram.

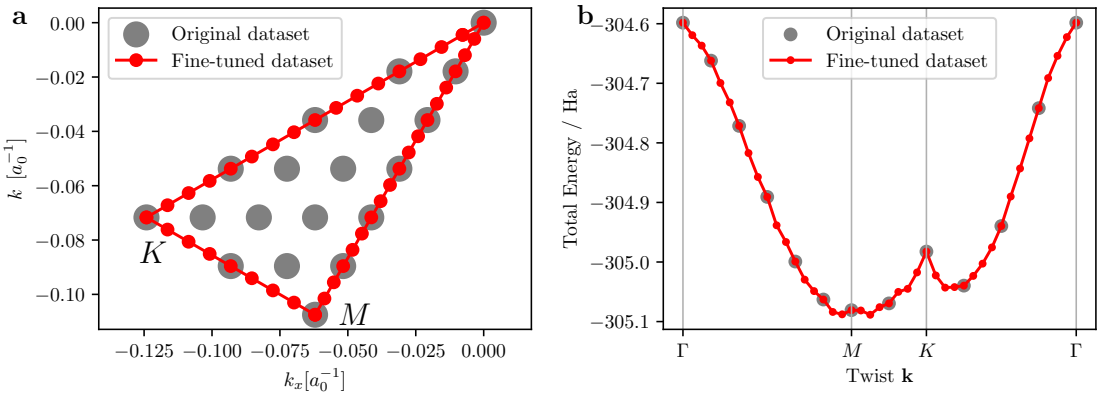

Figure 2: **Twist-dependent energy of Graphene a:** Grid of pretrained twists and path of fine-tuned values through Brillouin zone. **b:** Fine-tuned energies of graphene along path of twists across the Brillouin zone. Fine-tuned using shared optimization and around 100 additional optimization iterations per twist. Error bars are smaller than the size of the markers.

2.3 Lithium Hydride

We have also used the transferable DL-VMC ansatz to evaluate the energy-volume curve of LiH in the rock-salt crystal structure. As shown in Fig. 3 (see also Sec. S4), we obtain the energy-volume curve by fitting a Birch-Murhaghan equation of state to the total energies of a $2 \times 2 \times 2$ supercell at eight different lattice parameters. To reduce finite-size errors, the eight total energies are twist averaged using a $5 \times 5 \times 5$ Γ-centered Monkhorst-Pack grid and include structure-factor-based finite-size corrections. For comparison, Li et al. [18] performed a Γ-point calculation only and estimated finite-size errors by converging a Hartree-Fock calculation with an increasingly dense twist grid. To all results we add zero-point vibrational energies (ZPVE) taken from [28], making the calculated cohesive energy less negative by ≈ 8 mHa. The work of Li et al. [18] took no account of the ZPVE, explaining the slight difference between our depiction of their results, shown in Fig. 3, and their original publication [18].

We trained a single neural network wavefunction across 8 lattice constants and 10 symmetry-reduced twists, making 80 systems in total. In comparison, Li et al. [18] required a separate calculation for each geometry.

The Birch-Murnaghan fit gives an equilibrium lattice constant of $7.66(1)a_0$ (dotted orange line), which agrees well with the experimental value of $7.674(2)a_0$ [28]. Our Birch-Murnaghan estimate of the cohesive energy of $-177.3(1)$ mHa / primitive cell deviates from the experimental value of $-175.3(4)$ mHa by $-2.0(5)$ mHa. This marks an improvement over the DeepSolid results [18] of $-166.8(1)$ mHa, which differ from experiment by 8.5(5) mHa. Because we are able to optimize all systems at once, our results were obtained with roughly 5% of the compute required by DeepSolid.

Although we improve on the DeepSolid baseline, the cohesive energy might potentially still be impacted by finite-size effects because of the small size of the $2 \times 2 \times 2$ supercell used. To check this, we also studied a larger supercell containing $3 \times 3 \times 3$ primitive unit cells.

Previous work on molecules has shown that it is sometimes possible to transfer pre-trained neural wavefunctions to larger systems. For example, transferring parameters from a wavefunction pre-trained on small molecules to larger molecules allowed a reduction in the number of optimization steps by an order of magnitude compared to a random initialization of parameters [21, 29]. For our application, we transferred the neural wavefunction optimized to represent a $2 \times 2 \times 2$ simulation cell of LiH at multiple twists and lattice constants to a much larger $3 \times 3 \times 3$ supercell. The 108-electron $3 \times 3 \times 3$ system is one of the largest to have been studied using neural wavefunctions to date, with over three times more electrons than the 32-electron $2 \times 2 \times 2$ system used for pre-training. DeepSolid used $400,000$ optimization steps to get an estimate for the cohesive energy

and overestimated the energy by around 7 mHa / per primitive cell compared to the experimental results [18, 28]. By contrast, starting with the converged neural wavefunction for the $2 \times 2 \times 2$ supercell, we are able to calculate the cohesive energy for the $3 \times 3 \times 3$-supercell with only $8,000$ additional optimization steps shared across ten different twists. Using twist averaging, a structure-factor correction, and a ZPVE correction as before, we obtain a cohesive energy of -174.6 mHa / primitive cell, deviating from experiment by only 0.7(5) mHa / primitive cell. The magnitude of this deviation is close to the 0.4 mHa spread of experimental data obtained from different thermochemistry experiments [28]. Our twist-averaged $3 \times 3 \times 3$ calculation required only $\sim$2% of the computational resources used by Li et al. [18] for a single Γ-point calculation.

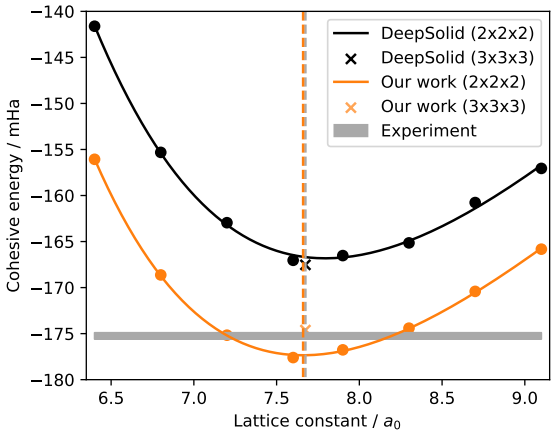

Figure 3: **Energy-volume curve of LiH per primitive cell** for a $2\times2\times2$ supercell as calculated using DeepSolid [18] and our transferable DL-VMC method. The DeepSolid results (black circles, with a Birch-Murnaghan fit represented as a black line) were obtained at a single twist, the Γ-point. Hartree-Fock corrections were applied, as discussed in [18], and a ZPVE correction added. Our results (orange circles, with a Birch-Murnaghan fit represented as an orange line) are twist averaged, using a $5 \times 5 \times 5$ Monkhorst-Pack grid per lattice constant. Structure-factor-based corrections were applied and a ZPVE correction added. The grey bar indicates the experimental uncertainty [28]. The statistical error bars are too small to be visible on this scale and therefore have beenn omitted. The vertical orange dashed line indicates the equilibrium lattice constant as calculated from the Birch-Murnaghan fit to our data. The orange cross shows the twist-averaged cohesive energy of a $3 \times 3 \times 3$ simulation cell, using again structure factor correction. This was obtained by transferring the network pre-trained for the $2 \times 2 \times 2$ system to a $3 \times 3 \times 3$ supercell, using only 8,000 additional optimization steps. A $5\times5\times5$ Monkhorst-Pack grid of twists was used. The black cross shows the result of DeepSolid's $3 \times 3 \times 3$ Γ-point calculation with a Hartree-Fock finite-size correction.

3 Discussion

Previous DL-VMC approaches were only capable of computing the wavefunction of a single system at once [18] or were limited to gas-phase molecules [21, 29]. In this work we propose a generalized neural network-based wavefunction for periodic systems. In addition to the electron positions, the network uses information about the Hartree-Fock one-electron eigenfunctions. Maximally localized Wannier functions are generated from the occupied Hartree-Fock orbitals and represented as expansions in a basis of local atomic-like orbitals. The expansion coefficients are then used as network inputs. By mapping cheap and low-accuracy orbital descriptors to highly accurate deep-learning-based orbitals, in a manner based on the ideas of Ref. [21], we are able to optimize a single neural wavefunction model across twists, lattice constants and supercell sizes all at once. This reduces the computational cost by an order of magnitude. Cost reductions are particularly important in solid-state simulations because large supercells must be studied to minimize finite-size effects.

We also investigate the transfer capabilities of a pre-trained wavefunction model. In particular, we transfer a pre-trained model to a system more than three times larger and find that energies converge using more than 50 times fewer optimization steps. This could pave the way to simulations of the large supercells required to study metals or perhaps even high-temperature semiconductors.

Furthermore, we find that our approach is able to represent qualitatively different wavefunctions within a single model. For example, for the hydrogen chain, a single ansatz with identical parameters can represent both the metallic state and the insulating state. A concurrent pre-print [30] exploring a related idea — pre-training a neural wavefunction embedding layer for a lattice model near a phase transition and then fine-tuning a final layer on either side of the transition — may shed some light on this success. Our embeddings can learn robust, transferable, features that allow efficient representation of the wavefunction in either phase.

Besides the transfer to larger systems, our method allows for efficient fine-grained twist-averaged calculations. Unlike previous DL-VMC methods, we use a single network to represent wavefunctions at multiple different twists, avoiding the computational overhead of training separate neural network-based wavefunctions at each twist. Our ansatz already allows the concurrent optimization of systems with varying numbers of particles, so this approach could be extended to grand-canonical twist averaging [31], in which the number of electrons in the supercell varies with the twist.

Our approach shares many of the limitations of other DL-VMC methods, including the sensitivity with regard to MCMC initialization. A standard practice in DL-VMC is to assign each electron a spin and initialize it close to the nuclei at the beginning of the calculation. If the electrons are initialized in an anti-ferromagnetic pattern, i.e., alternating the spins of neighboring atoms, but the ground state is ferromagnetic, as can be the case for the hydrogen chain when the inter-atomic separation is small (see Sec. 2.1), our approach tends to converge to local minima. FermiNet suffers from similar problems.

Another limitation arises from the allocation of compute budget between the multiple geometries or systems described by a single neural network. As discussed in Sec. 2.2, we allocate more compute during optimization to twists with a larger weight. This has a positive effect on twist-averaged results in general, because twists with higher contribution are converged to higher accuracy (see Tab. 1). However for individual twists, when plotting for example the band structure (see Fig. 2), not all twists are optimized to same accuracy potentially skewing results.

In comparison with other DL-VMC methods, the construction of our orbital matrix introduces an additional dependency on the positions of the nuclei (see Eq. (6)). Tests suggest that this worsens the empirical scaling of computational cost with the number of particles compared to FermiNet, and that the added cost is greatest for systems with a low ratio of electrons to nuclei. In the Supplementary Material, S1, we investigate empirically the effect on scaling of our newly proposed architecture. The overall scaling is in principle still dominated by the computation of the determinant and its derivative, implying that the time per iteration is $\mathcal{O}({n_{\text{el}}}^4)$, just as it is for FermiNet. For the system sizes investigated, however, we observe that our method requires approximately twice (the ratio is system dependent) as much run time per optimization step as FermiNet. This higher per-iteration cost is fortunately small relative to the orders of magnitude reduction in the total number of iterations steps required.

In summary, the approach introduced in this paper reduces the computational cost of DL-VMC simulations and allows them to be used to study larger supercells with more twists. By combining the present work with the efficient forward evaluation of the Laplacian of the wavefunction recently introduced by Li et al. [32] and using pseudo-potentials [33] to represent the core electrons, it should be possible to scale to even larger systems.

4 Methods

4.1 Notation

All vectors, matrices and tensors are denoted by **bold letters**, except for functions. We use lower-case indices $i, j = 1, \ldots, n_{\text{el}}$ for electron positions and upper case indices $I, J = 1, \ldots, N_{\text{atoms}}$ for

atom positions, where n_{el} and N_{atoms} are the numbers of electrons and atoms in the supercell. Orbitals are enumerated by the indices μ and ν, which range from 1 to n_{el}. The position of the i'th electron is $\boldsymbol{r}_i \in \mathbb{R}^3$. When i is not used as a subscript it denotes the imaginary unit. By $\boldsymbol{r} = (\boldsymbol{r}_1, \ldots, \boldsymbol{r}_{n_{\text{el}}})$ we denote the $3n_{\text{el}}$-dimensional vector of all electron positions. Similarly, nuclear positions and charges are represented by $\boldsymbol{R} = (\boldsymbol{R}_1, \ldots, \boldsymbol{R}_{N_{\text{atoms}}})$ and $\boldsymbol{Z} = (Z_1, \ldots, Z_{N_{\text{atoms}}})$. The matrix $\boldsymbol{L} \in \mathbb{R}^{3\times3}$ contains the supercell lattice vectors in its columns. The twist vector, which may always be reduced into the first Brillouin zone of the supercell, is denoted by $\boldsymbol{k}_{\text{s}}$. The dot product of two vectors $\boldsymbol{a}$ and $\boldsymbol{b}$ is written $\boldsymbol{a}\cdot\boldsymbol{b}$ and by $\odot$ we refer to the element-wise multiplication (Hadamard product).

4.2 Deep-learning Variational Monte Carlo

The time-independent Schrödinger equation for a solid takes the form

$$\hat{H}\Psi = E\Psi, \qquad \hat{H} = -\frac{1}{2}\sum_i \nabla^2_{\boldsymbol{r}_i} + \hat{V}_{\text{Coulomb}} \tag{2}$$

with the Hamiltonian in the Born-Oppenheimer approximation and Coulomb potential $\hat{V}_{\text{Coulomb}}$. A finite supercell is used to approximate the bulk solid, and the Coulomb potential is evaluated using the Ewald method, as described in [14, 34].

In this work we are interested in finding the lowest eigenvalue of the Schrödinger equation — the ground-state energy, E_0 — and the corresponding energy eigenfunction. To find an approximate solution, one can reformulate the Schrödinger equation as a minimization problem using the Rayleigh-Ritz variational principle. Given an arbitrary anti-symmetric trial wavefunction, $\Psi_{\boldsymbol{\theta}}$, with $\boldsymbol{\theta}$ denoting, for example, the trainable parameters of a neural network, the best attainable approximation to the ground state may be found by minimizing the energy expectation value

$$L(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{r}\sim\Psi^2_{\boldsymbol{\theta}}}\left[\frac{\hat{H}\Psi_{\boldsymbol{\theta}}}{\Psi_{\boldsymbol{\theta}}}\right] \geq E_0 \tag{3}$$

with respect to $\boldsymbol{\theta}$. An important constraint for the construction of the trial wavefunction arises from the Pauli exclusion principle, which states that the wavefunction must be antisymmetric with respect to the permutations of different electron coordinates [6]. In Sec. 4.3 we outline in detail the underlying architecture of our neural network-based wavefunction. As in previous work, we approximate the expectation value in Eq. (3) using Monte Carlo integration with samples drawn from the $3n_{\text{el}}$-dimensional probability density $|\Psi_{\boldsymbol{\theta}}(\boldsymbol{r})|^2$ [6, 8].

A list of all relevant hyperparameters can be found in the supplementary information S3.

4.3 Architecture

Overview Our ansatz can be broken down into the computation of periodic input features, the computation of embeddings $\boldsymbol{e}_{iJ}$ for each electron-nucleus pair, the computation of correlated orbitals, and the assembly of the final wavefunction Ψ_θ as a sum of Slater determinants. Each step serves a distinct purpose.

The input features enforce the periodic boundary conditions of the supercell. To capture correlation effects, we use a neural network to map single-electron coordinates to vectors in a latent space. These vectors, also known as embeddings, depend on the positions of all of the other electrons in a permutation equivariant way. Each embedding therefore contains information about the corresponding electron as well as its environment. The embeddings are subsequently mapped to many-electron orbitals as outlined below.

Ansatz Our wavefunction ansatz is a sum of Slater determinants multiplied by a Jastrow factor,

$$\Psi(\boldsymbol{r}, \boldsymbol{R}, \boldsymbol{Z}, \boldsymbol{k}_{\text{s}}) = e^{J(\boldsymbol{r})}\sum_{d=1}^{n_{\text{det}}} \det \boldsymbol{\Phi}_d(\boldsymbol{r}, \boldsymbol{R}, \boldsymbol{Z}, \boldsymbol{k}_{\text{s}}). \tag{4}$$

The optimization is free to adjust the relative normalizations of the determinants in the unweighted sum, making it equivalent to a weighted sum of normalized determinants, as might be used in a configuration-interaction expansion. The Jastrow factor $e^{J(\boldsymbol{r})}$ is node-less and follows the work of Hermann et al. [8], while the determinant enforces the fermionic antisymmetry. Instead of using single-particle orbitals in the determinant, as in most quantum chemical approaches, we follow other neural wavefunction methods [6] and promote every entry $\Phi_{d,i\mu}$ in the orbital matrix $\boldsymbol{\Phi}_d$ from a one-electron orbital, $\phi_{d,\mu}(\boldsymbol{r}_i)$, to a many-electron orbital, $\Phi_{d,i\mu}(\boldsymbol{r})$ (temporarily dropping the dependency on $\boldsymbol{R}$, $\boldsymbol{Z}$, and $\boldsymbol{k}_s$ for the sake of brevity). The many-electron orbitals are permutation equivariant, such that applying a permutation π to the electron position vectors permutes the rows of $\boldsymbol{\Phi}_d$ by π, i.e., $\Phi_{d,i\mu}(\boldsymbol{r}_{\pi(1)}, \ldots, \boldsymbol{r}_{\pi(n_{\mathrm{el}})}) = \Phi_{d,\pi(i)\mu}(\boldsymbol{r}_1, \ldots, \boldsymbol{r}_{n_{\mathrm{el}}})$. This ensures that the determinant has the correct fermionic symmetry. Each entry is constructed as a linear combination of atom-centered functions with permutation equivariant dependencies on both electrons and atoms

$$\Phi_{d,i\mu}(\boldsymbol{r}, \boldsymbol{R}, \boldsymbol{Z}, \boldsymbol{k}_{\mathrm{s}}) = e^{i\boldsymbol{k}_{\mathrm{s}}\cdot\boldsymbol{r}_i} \sum_{J=1}^{N_{\mathrm{atoms}}} \varphi_{d\mu}(\boldsymbol{r}_i, \{\boldsymbol{r}\}, \boldsymbol{R}_J, \{\boldsymbol{R}\}). \tag{5}$$

Here, $\{\boldsymbol{r}\}$ and $\{\boldsymbol{R}\}$ denotes the (permutation invariant) set of electron and atom positions, respectively. The phase factor enforces the twisted boundary conditions, as explained in Sec. 4.7. To construct the $\varphi_{d\mu iJ} \equiv \varphi_{d\mu}(\boldsymbol{r}_i, \{\boldsymbol{r}\}, \boldsymbol{R}_J, \{\boldsymbol{R}\})$ using a neural network, we use an adaptation of the recently proposed transferable atomic orbital ansatz [21, 29]. The orbitals are written as the inner product of an electron-nuclear embedding $\boldsymbol{e}_{iJ} \in \mathbb{R}^{n_{\mathrm{emb}}}$ and an orbital embedding $\boldsymbol{W}_{d\mu J} \in \mathbb{C}^{n_{\mathrm{emb}}}$, multiplied by an exponential envelope,

$$\varphi_{d\mu iJ} = (\boldsymbol{W}_{d\mu J} \cdot \boldsymbol{e}_{iJ}) e^{-a_{d\mu J}||\boldsymbol{s}_{iJ}||^{\mathrm{per}}}. \tag{6}$$

where $a_{d\mu J}$ is a learnable decay rate, $\boldsymbol{s}_{iJ}$ is the vector from nucleus J to electron i, expressed in the basis of the supercell lattice vectors, and $||\boldsymbol{s}_{iJ}||^{\mathrm{per}}$ is the modulus of $\boldsymbol{s}_{iJ}$ in a periodic norm explained below. Both the orbital embedding $\boldsymbol{W}_{d\mu J}$ and the decay length $a_{d\mu J}$ depend on the orbital μ and atom J and are different for each determinant d.

To obtain $\boldsymbol{W}_{d\mu J}$ and $a_{d\mu J}$ in a transferable way, we do not parameterize them directly but represent them as functions of some orbital-specific descriptor $\tilde{\boldsymbol{c}}_{\mu J} \in \mathbb{R}^{d_{\mathrm{orb}}}$:

$$\boldsymbol{W}_{d\mu J} = f_d^{\mathrm{W}}(\tilde{\boldsymbol{c}}_{\mu J}), \qquad a_{d\mu J} = f_d^{\mathrm{a}}(\tilde{\boldsymbol{c}}_{\mu J}), \tag{7}$$

with $f^{\mathrm{W}} : \mathbb{R}^{d_{\mathrm{orb}}} \to \mathbb{C}^{n_{\mathrm{det}} \times d_{\mathrm{emb}}}$ and $f^{\mathrm{a}} : \mathbb{R}^{d_{\mathrm{orb}}} \to \mathbb{R}^{n_{\mathrm{det}}}$ denoting simple multi-layer perceptrons. The orbital embedding includes information about single-particle orbitals of the system calculated with a mean-field method, which is key for the transferability of the ansatz. The inputs are the orbital features $\tilde{\boldsymbol{c}}_{\mu J} \in \mathbb{R}^{d_{\mathrm{orb}}}$, which are concatenations of the expansion coefficients of the localized mean-field orbitals in an atom-centred basis set, the twist $\boldsymbol{k}_{\mathrm{s}}$, the mean position of orbital μ, and the position of atom J, with a combined dimensionality of d_{orb}. While all parameters and intermediate computations of our network are real-valued, the last layer of f^{W} is complex-valued to allow the network to represent complex-valued wavefunctions.

An important difference with respect to previous neural network-based wavefunctions is the use of electron-nuclear embeddings $\boldsymbol{e}_{iJ}$, which describe the interaction between electron i and nucleus J. Other architectures such FermiNet, but also the more closely related transferable atomic orbital ansatz [21], use embeddings to represent the interactions of a single electron i with all nuclei instead. However, when the embeddings are both invariant under permutation of nuclei (which we require for efficient transferability) and invariant under translation of particles by a supercell lattice vector (which we require to enforce boundary conditions), they become periodic on the primitive lattice (see the supplementary information S2), not just the supercell lattice. This is too restrictive to represent correlation beyond a single primitive cell. We therefore opt to use electron-nucleus embeddings that are equivariant under permutation of nuclei at some additional computational cost explained in Sec. S2.

Input We require our representation of the difference vectors $\boldsymbol{r}_{ij} = \boldsymbol{r}_i - \boldsymbol{r}_j$, $\boldsymbol{r}_{iI} = \boldsymbol{r}_i - \boldsymbol{R}_I$ and $\boldsymbol{r}_{IJ} = \boldsymbol{r}_I - \boldsymbol{R}_J$ to be periodic with respect to the supercell lattice. This is accomplished using

the approach introduced by Cassella et al. [14]. The first step is to transform the coordinates into supercell fractional coordinates with $\boldsymbol{s}_{ij} = \boldsymbol{L}^{-1}\boldsymbol{r}_{ij}$, $\boldsymbol{s}_{iI} = \boldsymbol{L}^{-1}\boldsymbol{r}_{iI}$ and $\boldsymbol{s}_{IJ} = \boldsymbol{L}^{-1}\boldsymbol{r}_{IJ}$. Periodic versions of the difference vectors are then obtained by applying sine and cosine element-wise,

$$\omega(\boldsymbol{s}) := [\sin(2\pi\boldsymbol{s}), \cos(2\pi\boldsymbol{s})], \qquad \omega : \mathbb{R}^3 \to \mathbb{R}^6, \tag{8}$$

$$\boldsymbol{x}_{ij} := \omega(\boldsymbol{s}_{ij}), \quad \boldsymbol{x}_{iJ} := \omega(\boldsymbol{s}_{iJ}), \quad \boldsymbol{x}_{IJ} := \omega(\boldsymbol{s}_{IJ}), \tag{9}$$

whereas square brackets denotes the concatenation operator. For the distance, we use the following periodic norm

$$(||\boldsymbol{s}||^{\mathrm{per}})^2 = \sum_{l,p=1}^{3} \Big(\big(1 - \cos(2\pi s_l)\big) A_{lp} \big(1 - \cos(2\pi s_p)\big) + \sin(2\pi s_l) A_{lp} \sin(2\pi s_p) \Big) \tag{10}$$

for a vector $\boldsymbol{s} \in \mathbb{R}^3$ with the lattice metric $\boldsymbol{A} := LL^T$. This norm is used to define the periodic distance features:

$$x_{ij} = ||\boldsymbol{s}_{ij}||^{\mathrm{per}}, \quad x_{iJ} = ||\boldsymbol{s}_{iJ}||^{\mathrm{per}}, \quad x_{IJ} = ||\boldsymbol{s}_{IJ}||^{\mathrm{per}}. \tag{11}$$

Embedding The periodic input features are used to generate high-dimensional embeddings $\boldsymbol{e}_{iJ}$ for the construction of the orbital matrix. The following embedding is a slight adaption of the appraoch used in the recently proposed Moon architecture [35]. We start by aggregating the electron-electron features into message vectors $\boldsymbol{m}_i^0$ for each electron i

$$\boldsymbol{m}_i^0 = \sum_{j=1}^{n_{\mathrm{el}}} \Gamma^{\text{e-e}}(x_{ij},\, \boldsymbol{x}_{ij}) \odot \sigma\left(\boldsymbol{W}^{\mathrm{m}} \tilde{\boldsymbol{x}}_{ij} + \boldsymbol{b}^{\mathrm{m}}\right), \tag{12}$$

and compute the initial electron embeddings $\boldsymbol{h}_i^0$ as a trainable function of these messages

$$\boldsymbol{h}_i^0 = \sigma\left(\boldsymbol{W}^0 \boldsymbol{m}_i^0 + \boldsymbol{b}^0\right). \tag{13}$$

The matrices $\boldsymbol{W}^{\mathrm{m}}$, $\boldsymbol{W}^0$ and vectors $\boldsymbol{b}^{\mathrm{m}}$, $\boldsymbol{b}^0$ are trainable parameters, σ is an activation function which is applied elementwise and $\odot$ denotes the elementwise product. The filter function $\Gamma^{\text{e-e}}$

$$\Gamma^{\text{e-e}}(x_{ij},\, \boldsymbol{x}_{ij}) = \sigma\big(\boldsymbol{W}^{\mathrm{env}} \boldsymbol{x}_{ij} + \boldsymbol{b}\big) \odot \exp\big(-x_{ij}^2 \boldsymbol{\alpha}\big), \tag{14}$$

ensures an exponential decay with a trainable vector of length-scales, $\boldsymbol{\alpha}$ and a trainable matrix $\boldsymbol{W}^{\mathrm{env}}$. Furthermore, the input features $\tilde{\boldsymbol{x}}_{ij} = [x_{ij}, \boldsymbol{x}_{ij}, \boldsymbol{k}_{\mathrm{s}}]$ make the embedding twist dependent to allow for better transferability across twists.

To initialize the atomic features, we first one-hot encode the nuclear charges $\boldsymbol{Z}$ into a matrix $\tilde{\boldsymbol{H}} \in \mathbb{R}^{N_{\mathrm{atoms}} \times n_{\mathrm{species}}}$. We then initialize the atom embeddings $\boldsymbol{H}_I^0$ analogously to the electron embeddings, by aggregating atom-atom features for each atom I

$$\boldsymbol{H}_I^0 = \sum_{J=1}^{N_{\mathrm{atoms}}} \Gamma^{\text{a-a}}(x_{IJ},\, \boldsymbol{x}_{IJ}) \odot \sigma\big(\boldsymbol{W}^{\mathrm{a}} \tilde{\boldsymbol{H}}_J + \boldsymbol{b}^{\mathrm{a}}\big), \tag{15}$$

using a trainable weight matrix $\boldsymbol{W}^{\mathrm{a}}$ and bias vector $\boldsymbol{b}^{\mathrm{a}}$. We then incorporate electron-atom information by contracting across all electrons

$$\boldsymbol{H}_I^1 = \sum_{i=1}^{n_{\mathrm{el}}} \boldsymbol{e}_{iI}^0 \odot \left(\boldsymbol{W}^{\text{e-a}}\, \Gamma^{\text{e-a}}(x_{iI},\, \boldsymbol{x}_{iI})\right) \tag{16}$$

$$\boldsymbol{e}_{iI}^0 = \sigma\big(\boldsymbol{h}_i^0 + \boldsymbol{H}_I^0 + \boldsymbol{W}^{\mathrm{edge}} \tilde{\boldsymbol{x}}_{iI} + \boldsymbol{b}^{\mathrm{edge}}\big), \tag{17}$$

with $\tilde{\boldsymbol{x}}_{iI} = [x_{iI}, \boldsymbol{x}_{iI}, \boldsymbol{k}_{\mathrm{s}}]$ and trainable matrices $\boldsymbol{W}^{\text{e-a}}, \boldsymbol{W}^{\mathrm{edge}}$ and bias $\boldsymbol{b}^{\mathrm{edge}}$. Subsequently, the atom embeddings are updated with L dense layers

$$\boldsymbol{H}_I^{l+1} = \sigma\big(\boldsymbol{W}^l \boldsymbol{H}_I^l + \boldsymbol{b}^l\big) + \boldsymbol{H}_I^l, \tag{18}$$

to finally diffuse them to electron-atom embeddings $\boldsymbol{e}_{iI}$ of the form

$$\boldsymbol{e}_{iI} = \sigma\big(\boldsymbol{W}^{\text{out}_1}\boldsymbol{e}^0_{iI} + \boldsymbol{H}^L_I + \boldsymbol{W}^{\text{out}_2}\boldsymbol{h}^0_i + \boldsymbol{b}^{\text{out}}\big) \odot \big(\boldsymbol{W}^{\text{out}_3}\,\Gamma^{\text{out}}(x_{iI},\, \boldsymbol{x}_{iI})\big). \tag{19}$$

with trainable matrix $\boldsymbol{W}^{\text{out}_1}, \boldsymbol{W}^{\text{out}_2}, \boldsymbol{W}^{\text{out}_3}$ and trainable bias vector $\boldsymbol{b}^{\text{out}}$. For the sake of simplicity we omitted the spin dependence in this presentation of the different embedding stages. Compared to the original Moon embedding [35], we use separate filters Γ for the intermediate layers and the output layer, we include the twist as input feature, and omit the final aggregation step from electron-ion embeddings $\boldsymbol{e}_{iI}$ to electron embeddings $\boldsymbol{e}_i$.

Orbitals The orbital features $\tilde{\boldsymbol{c}}_{\mu J}$ are a concatenation of four different types of features. First, as proposed by Scherbela et al. [21], we rely on mean-field coefficients from a Hartree-Fock calculation. The mean-field orbitals ϕ_μ are localized as described in Sec. 4.6 and expanded in periodic, atom-centered, basis functions b_η

$$\phi_\mu(\boldsymbol{r}_i) = \sum_{I=1}^{N_{\text{atoms}}} \sum_{\eta=1}^{n_{\text{b}}} c_{I\mu,\eta}\; b_\eta(\boldsymbol{r}_i - \boldsymbol{R}_I), \tag{20}$$

where n_{b} represents the per-atom basis set size of the Hartree-Fock calculation. We use a periodic version of the cc-pVDZ basis set [36] and find no strong dependence of our results on the basis set used. Additionally, we include relative atom positions $\tilde{\boldsymbol{R}}_I$

$$\tilde{\boldsymbol{R}}_I = \boldsymbol{R}_I - \frac{\sum_{J=1}^{N_{\text{atoms}}} \boldsymbol{R}_J Z_J}{\sum_{K=1}^{N_{\text{atoms}}} Z_K} \tag{21}$$

and analogously relative orbital positions $\tilde{\boldsymbol{R}}^{\text{orb}}_\mu$

$$\tilde{\boldsymbol{R}}^{\text{orb}}_\mu = \boldsymbol{R}^{\text{orb}}_\mu - \frac{\sum_{J=1}^{N_{\text{atoms}}} \boldsymbol{R}_J Z_J}{\sum_{K=1}^{N_{\text{atoms}}} Z_K}, \tag{22}$$

where $\boldsymbol{R}^{\text{orb}}_\mu$ is the position of the localized orbital μ as outlined in Sec. 4.6. This allows the network to differentiate between different atoms and orbitals within the supercell. As a final feature we include the twist of the system

$$\tilde{\boldsymbol{k}}^{\text{s}}_I = [\boldsymbol{k}_{\text{s}},\, \sin(\boldsymbol{R}_I \cdot \boldsymbol{k}_{\text{s}}), \cos(\boldsymbol{R}_I \cdot \boldsymbol{k}_{\text{s}})] \in \mathbb{R}^5. \tag{23}$$

The final orbital features $\tilde{\boldsymbol{c}}_{I\mu}$ are obtained as a concatenation

$$\tilde{\boldsymbol{c}}_{I\mu} = [\boldsymbol{c}_{I\mu},\, \tilde{\boldsymbol{R}}_I,\, \tilde{\boldsymbol{R}}^{\text{orb}}_\mu,\, \tilde{\boldsymbol{k}}^{\text{s}}_I] \in \mathbb{R}^{d_{\text{orb}}}, \tag{24}$$

where $d_{\text{orb}} = n_b + 11$.

4.4 Sampling

We use the Metropolis Hastings algorithm [37] to draw samples $\boldsymbol{r}$ from our unnormalized density $|\Psi_{\boldsymbol{\theta}}|^2$. We use Gaussian all-electron proposals $\boldsymbol{r}^{\text{prop}}$ of the form

$$\boldsymbol{r}^{\text{prop}} = \boldsymbol{r} + s\boldsymbol{\delta}, \tag{25}$$

where $\boldsymbol{\delta}$ is drawn from a $3n_{\text{el}}$-dimensional standard normal distribution. We continously adjust the stepsize s to obtain a mean acceptance probability of 50%.

When calculating properties of the hydrogen chain for different lattice constants R, special care must be given to the treatment of spins. The hydrogen chain has two phases with different arrangements of spins: In the insulating phase at large lattice constant, the ground state is anti-ferromagnetic, i.e. neighbouring spins prefer to be aligned antiparallel. In the metallic phase at small lattice constant, this antiferromagnetic ordering decreases and the system may even show ferromagnetic domains [3]. Moving between these two configurations is difficult using local Monte

Carlo updates as given by Eq. (25), so we modify our Metropolis Hastings proposal function. In addition to moving electrons in real space, we occasionally propose moves that swap the positions of two electrons with opposite spin. To avoid biasing our sampling towards either spin configuration, we initialize half our Monte Carlo walkers in the antiferromagnetic configuration (neighbouring electrons having opposite spin) and half our Monte Carlo walkers in a ferromagnetic configuration (all spin-up electrons in one half of the chain and all spin-down electrons in the other half). We found that on the contrary initializing all walkers in the antiferromagnetic configuration (as might be indicated, for example, by a mean-field calculation) can cause the optimization to fall into local energy minima during wavefunction optimization.

4.5 Complex KFAC

We use the Kronecker Factored Approximate Curvature (KFAC) method [38] to optimize the trainable parameters of our ansatz. KFAC uses the Fisher information matrix as a metric in the space of wavefunction parameters. For real wavefunctions, the Fisher matrix is equivalent to the preconditioner used in the stochastic reconfiguration method [6], but this is not the case for complex wavefunctions. Instead, the Fubini-Study metric should be used, given by

$$F_{ij} = \mathrm{Re}\left\{ \left\langle \frac{\partial \ln \psi^*}{\partial \theta_i} \frac{\partial \ln \psi}{\partial \theta_j} \right\rangle \right\} \tag{26}$$

Writing the complex wavefunction in polar form, $\psi = \rho e^{i\phi}$, this becomes

$$F = \left\langle \frac{\partial \ln \rho}{\partial \theta_i} \frac{\partial \ln \rho}{\partial \theta_j} + \frac{\partial \phi}{\partial \theta_i} \frac{\partial \phi}{\partial \theta_j} \right\rangle, \tag{27}$$

where the first term is the Fisher information matrix and the second term is the new contribution due to the phase of the wavefunction. The second term is zero if the phase is a global constant, as is the case when the phase arises from the twist of the wavefunction only.

4.6 Orbital localization

To obtain orbital features that generalize well across system sizes, we do not use the canonical mean-field coefficients $\boldsymbol{c}$ as network inputs. Rather, we use the coefficients $\boldsymbol{c}^{\mathrm{loc}}$ of maximally localized Wannier orbitals computed from $\boldsymbol{c}$. We follow the procedure of [39] to find a unitary rotation $\boldsymbol{U}$ within the subspace spanned by the occupied orbitals. Given a set of mean-field orbitals $\phi_\mu(\boldsymbol{r}), \mu = 1, \ldots, n_{\mathrm{el}}$, expanded in periodic, atom-centered basis functions $b_{I\eta}(\boldsymbol{r}), I = 1, \ldots, N_{\mathrm{atoms}}, \eta = 1, \ldots, n_{\mathrm{b}}$, as described in Sec. 4.3, we compute the complex polarization matrix

$$\chi_{\alpha,\nu\mu} = \int \phi^*_\nu(\boldsymbol{r}) e^{i\boldsymbol{r}^T \boldsymbol{G}_\alpha} \phi_\mu(\boldsymbol{r}) \mathrm{d}\boldsymbol{r}, \quad \chi \in \mathbb{C}^{3\times n_{\mathrm{orb}} \times n_{\mathrm{orb}}} \tag{28}$$

where $\boldsymbol{G} = 2\pi \boldsymbol{L}^{-T}$ is the matrix of reciprocal lattice vectors. Given a unitary transformation $\boldsymbol{U} \in \mathbb{C}^{n_{\mathrm{orb}} \times n_{\mathrm{orb}}}$, the transformed polarization matrix $\hat{\boldsymbol{\chi}}$ and the corresponding localization loss $\mathcal{L}$ are given by

$$\boldsymbol{\Omega}_{\alpha\mu} = \hat{\boldsymbol{\chi}}_{\alpha,\mu\mu} = \left(\boldsymbol{U}^\dagger \chi_\alpha \boldsymbol{U}\right)_{\mu\mu} \tag{29}$$

$$\mathcal{L}(\boldsymbol{U}) = -||\boldsymbol{\Omega}(\boldsymbol{U})||_2^2, \tag{30}$$

where $||\cdot||_2$ denotes the L_2-norm. To facilitate unconstrained optimization, we parameterize the unitary matrix $\boldsymbol{U}$ as the complex matrix exponential of a symmetrized, unconstrained complex matrix $\boldsymbol{A}$:

$$\boldsymbol{U} = e^{\frac{i}{2}(\boldsymbol{A}+\boldsymbol{A}^\dagger)}. \tag{31}$$

We obtain the optimal $\boldsymbol{U}^{\mathrm{loc}}$, and corresponding orbital coefficients $\boldsymbol{c}^{\mathrm{loc}}$ via gradient-based optimization

$$\boldsymbol{U}^{\mathrm{loc}} = \underset{\boldsymbol{U}}{\mathrm{argmin}}\, \mathcal{L}(\boldsymbol{U}), \qquad c^{\mathrm{loc}}_{I\eta,\mu} = \sum_m c_{I\eta,\nu} U^{\mathrm{loc}}_{\nu\mu}, \tag{32}$$

using the Adam [40] optimizer. For orthorombic supercells, the position of the Wannier center $\boldsymbol{R}_\mu^{\rm orb}$ of the localized orbital μ can be inferred from the localized polarization matrix $\hat{\boldsymbol{\chi}}$ as

$$R_{l\alpha}^{\rm orb} = -\frac{\boldsymbol{L}_{\alpha\alpha}}{2\pi}\mathrm{Im}\log\hat{\boldsymbol{\chi}}_{\mu\mu}^{\alpha}, \qquad \alpha = 1\dots 3, \quad \mu = 1\dots n_{\rm orb}. \tag{33}$$

For other supercells we follow the generalization given in [39].

4.7 Observables and post-processing

Twist-averaged boundary conditions In a finite system, there are finite-size errors related to both the artificial constraint of periodicity in the supercell and the lack of correlations of longer range than the supercell. The effects of the former on the single-particle contributions to the Hamiltonian, namely the kinetic energy, the Hartree-energy and the electron-ion interaction, can be reduced by using twist-averaged boundary conditions (TABC)[20, 25]. This means that the wavefunction obeys

$$\Psi(\boldsymbol{r}_1,\dots,\boldsymbol{r}_i+\boldsymbol{L}_\alpha,\dots,\boldsymbol{r}_N) = e^{i\boldsymbol{k}\cdot\boldsymbol{L}_\alpha}\Psi(\boldsymbol{r}_1,\dots,\boldsymbol{r}_i,\dots,\boldsymbol{r}_N), \tag{34}$$

where $\boldsymbol{L}_\alpha$ is the α'th supercell lattice vector. TABC are enforced by adding a position-dependent phase $e^{i\boldsymbol{k}_s\cdot\boldsymbol{r}_i}$ for each electron in the transferable atomic orbitals, as seen in Eq. (5) and averaging observables across a grid of twists $\boldsymbol{k}_s$ spanning first Brillouin zone.

Structure factor correction In order to handle finite-size errors in the Ewald energy, we use the finite-size corrections proposed by [25]. Writing the Ewald energy in terms of Fourier series, we get

$$\left\langle \hat{V}_E \right\rangle = \frac{N}{2}\left\{ v_M + \frac{1}{\Omega}\sum_{\boldsymbol{G}_s\neq 0} v_E(G_s)[S(\boldsymbol{G}_s)-1] \right\} + \frac{1}{2\Omega}\sum_{\boldsymbol{G}_p\neq 0} v_E(G_p)\rho(\boldsymbol{G}_p)\rho^*(\boldsymbol{G}_p). \tag{35}$$

Here v_M is the Madelung energy, Ω is the supercell volume, $v_E(\boldsymbol{k}) = 4\pi/k^2$ is the Fourier transform of the Coulomb interaction, and $\boldsymbol{G}_s$ ($\boldsymbol{G}_p$) is a simulation (primitive) cell reciprocal lattice vector. The translationally-averaged structure factor $S(\boldsymbol{G}_p)$ is defined by

$$S(\boldsymbol{G}_s) = \frac{1}{N}\left[\langle\hat{\rho}(\boldsymbol{G}_s)\hat{\rho}^*(\boldsymbol{G}_s)\rangle - \langle\hat{\rho}(\boldsymbol{G}_s)\rangle\langle\hat{\rho}^*(\boldsymbol{G}_s)\rangle\right], \tag{36}$$

where $\hat{\rho}(\boldsymbol{G}_s) = \sum_i \exp\left(-i\boldsymbol{G}_s\cdot\boldsymbol{r}_i\right)$ is the Fourier representation of the operator for the electron density. The structure factor converges fairly rapidly with supercell size, so we can assume that $S_\Omega(\boldsymbol{k}) \approx S_\infty(\boldsymbol{k})$. In this limit, the largest contribution to the error is the omission of the $G_s = 0$ term in the first sum. In cubic systems, we have $S(\boldsymbol{k}) \propto \eta k^2 + O(k^4)$, with odd terms missing due to inversion symmetry, and the $\boldsymbol{k}\to 0$ limit of $S(\boldsymbol{k})v_E(k)$ is well defined. As such, to a first approximation, the Ewald finite-size error is given by

$$\Delta V_E \approx \frac{N}{2\Omega}\lim_{k\to 0} v_E(k)S(\boldsymbol{k}) = \frac{4\pi N}{2\Omega}\lim_{k\to 0}\frac{S(\boldsymbol{k})}{k^2}. \tag{37}$$

Sampling $S(\boldsymbol{G}_s)$ at supercell reciprocal lattice vectors $\boldsymbol{G}_s$, we approximate the limit $k\to 0$ by fitting the function

$$S(k) \approx f(k) = 1 - e^{-a_0k^2 - a_1k^4}, \tag{38}$$

with a_0 and a_1 greater than zero. The form of the fit ensures that $S(k)$ has the correct k^2 behavior at small k and that $\lim_{k\to\infty} S(\boldsymbol{k}) = 1$. The finite-size correction $\Delta V_E \approx 4\pi N a_0/2\Omega$.

Zero-point vibrational energy for graphene To estimate the zero-point vibrational energy (ZPVE) contribution for graphene, we obtained the phonon density of states $D(\omega)$ calculated within DFT at the PBE [1] level from [41]. The ZPVE energy per primitive cell, $E_{\rm ZPVE}$, is then given as

$$E_{\rm ZPVE} = \frac{3N_{\rm atoms}^{\rm prim}}{\int D(\omega)\mathrm{d}\omega}\int D(\omega)\frac{1}{2}\hbar\omega\mathrm{d}\omega, \tag{39}$$

where $N_{\text{atoms}}^{\text{prim}} = 2$ is the number of atoms per primitive unit cell of graphene. This yields a ZPVE of 12.8 mHa / primitive cell.

5 Code availability

All code is available on our github repository `https://github.com/mdsunivie/deeperwin`.

6 Data availability

All data, including geometries, configurations, and the figure source data is available on our github repository `https://github.com/mdsunivie/deeperwin`.

7 Acknowledgements

We acknowledge helpful discussions with Ji Chen, Xiang Li, Tony Lou, and Gunnar Arctaedius. We gratefully acknowledge financial support from the following grants: Austrian Science Fund FWF Project I 3403 (P.G.), WWTF-ICT19-041 (L.G.) and Aker Scholarship (H.S.). The computational results have been partially achieved using the Vienna Scientific Cluster and Leonardo (Project L-AUT 005). HS gratefully acknowledges the Gauss Centre for Supercomputing e.V. (www.gauss-centre.eu) for providing computing time through the John von Neumann Institute for Computing (NIC) on the GCS Supercomputer JUWELS at Jülich Supercomputing Centre (JSC); the HPC RIVR consortium and EuroHPC JU for resources on the Vega high performance computing system at IZUM, the Institute of Information Science in Maribor; and the UK Engineering and Physical Sciences Research Council for resources on the Baskerville Tier 2 HPC service. Baskerville was funded by the EPSRC and UKRI through the World Class Labs scheme (EP/T022221/1) and the Digital Research Infrastructure programme (EP/W032244/1) and is operated by Advanced Research Computing at the University of Birmingham. The funders had no role in study design, data collection and analysis, decision to publish or preparation of the manuscript.

8 Author Contributions Statement

HS proposed the idea of shared twist averaging; MS and LG proposed concrete architecture and approach. LG, MS and HS jointly worked on implementation: LG worked on the periodic Hamiltonian and contributed to the model. MS implemented/extended the model, orbital localization and contributed to the mean-field orbitals. HS extended KFAC to complex wavefunctions, implemented the evaluation of the mean-field orbitals and contributed to the periodic Hamiltonian. MS designed and ran experiments on the H chains. LG designed and ran experiments for LiH and graphene with support from HS and MS. LG, MS, HS and WMCF jointly wrote the paper with supervision and funding from PG and WMCF.

9 Competing Interests Statement

The authors declare no competing interests.

References

[1] John P. Perdew, Kieron Burke, and Matthias Ernzerhof. "Generalized Gradient Approximation Made Simple". In: *Phys. Rev. Lett.* 77.18 (Oct. 1996), pp. 3865–3868. DOI: 10.1103/PhysRevLett.77.3865.

[2] C. David Sherrill. "Frontiers in Electronic Structure Theory". In: *The Journal of Chemical Physics* 132.11 (Mar. 2010), p. 110902. DOI: 10.1063/1.3369628.

[3] Simons Collaboration on the Many-Electron Problem et al. "Ground-State Properties of the Hydrogen Chain: Dimerization, Insulator-to-Metal Transition, and Magnetic Phases". In: *Physical Review X* 10.3 (Sept. 2020), p. 031058. DOI: 10.1103/PhysRevX.10.031058.

[4] Kieron Burke. "Perspective on Density Functional Theory". In: *The Journal of Chemical Physics* 136.15 (Apr. 2012), p. 150901. ISSN: 0021-9606. DOI: 10.1063/1.4704546.

[5] W. M. C. Foulkes et al. "Quantum Monte Carlo Simulations of Solids". In: *Rev. Mod. Phys.* 73.1 (Jan. 2001), pp. 33–83. ISSN: 0034-6861, 1539-0756. DOI: 10.1103/RevModPhys.73.33.

[6] David Pfau et al. "Ab Initio Solution of the Many-Electron Schrödinger Equation with Deep Neural Networks". In: *Phys. Rev. Res.* 2.3 (2020), p. 033429. DOI: 10.1103/PhysRevResearch.2.033429.

[7] Giuseppe Carleo and Matthias Troyer. "Solving the Quantum Many-Body Problem with Artificial Neural Networks". In: *Science* 355.6325 (2017), pp. 602–606. DOI: 10.1126/science.aag2302.

[8] Jan Hermann, Zeno Schätzle, and Frank Noé. "Deep-Neural-Network Solution of the Electronic Schrödinger Equation". In: *Nature Chemistry* 12.10 (2020), pp. 891–897. DOI: 10.1038/s41557-020-0544-y.

[9] Leon Gerard et al. "Gold-Standard Solutions to the Schrödinger Equation Using Deep Learning: How Much Physics Do We Need?" In: *Advances in Neural Information Processing Systems*. Vol. 35. Curran Associates, Inc., 2022, pp. 10282–10294.

[10] Ingrid von Glehn, James S Spencer, and David Pfau. "A Self-Attention Ansatz for Ab-Initio Quantum Chemistry". In: *The Eleventh International Conference on Learning Representations*. 2023.

[11] Kenny Choo et al. "Symmetries and Many-Body Excitations with Neural-Network Quantum States". In: *Physical Review Letters* 121.16 (Oct. 2018), p. 167204. DOI: 10.1103/PhysRevLett.121.167204.

[12] Or Sharir et al. "Deep Autoregressive Models for the Efficient Variational Simulation of Many-Body Quantum Systems". In: *Physical Review Letters* 124.2 (Jan. 2020), p. 020503. DOI: 10.1103/PhysRevLett.124.020503.

[13] Di Luo and Bryan K. Clark. "Backflow Transformations via Neural Networks for Quantum Many-Body Wave Functions". In: *Physical Review Letters* 122.22 (June 2019), p. 226401. DOI: 10.1103/PhysRevLett.122.226401.

[14] Gino Cassella et al. "Discovering Quantum Phase Transitions with Fermionic Neural Networks". In: *Phys. Rev. Lett.* 130.3 (2023), p. 036401. DOI: 10.1103/PhysRevLett.130.036401.

[15] Gabriel Pescia et al. "Message-Passing Neural Quantum States for the Homogeneous Electron Gas". In: *arXiv preprint* (2023). arXiv: 2305.07240.

[16] Jane Kim et al. "Neural-Network Quantum States for Ultra-Cold Fermi Gases". In: *arXiv preprint* (2023). arXiv: 2305.08831.

[17] Wan Tong Lou et al. "Neural Wave Functions for Superfluids". In: *arXiv preprint* (2024). arXiv: 2305.06989.

[18] Xiang Li, Zhe Li, and Ji Chen. "Ab Initio Calculation of Real Solids via Neural Network Ansatz". In: *Nature Communications* 13.1 (2022), p. 7895. DOI: 10.1038/s41467-022-35627-1.

[19] Xiang Li, Yubing Qian, and Ji Chen. "Electric Polarization from Many-Body Neural Network Ansatz". In: *arXiv preprint* (Aug. 2023). arXiv: 2307.02212.

[20] C. Lin, F.-H. Zong, and D. M. Ceperley. "Twist-Averaged Boundary Conditions in Continuum Quantum Monte Carlo". In: *Physical Review E* 64.1 (June 2001), p. 016702. ISSN: 1063-651X, 1095-3787. DOI: 10.1103/PhysRevE.64.016702.

[21] Michael Scherbela, Leon Gerard, and Philipp Grohs. "Towards a Transferable Fermionic Neural Wavefunction for Molecules". In: *Nature Communications* 15.1 (2024), p. 120. DOI: 10.1038/s41467-023-44216-9.

[22] Simons Collaboration on the Many-Electron Problem et al. "Towards the Solution of the Many-Electron Problem in Real Materials: Equation of State of the Hydrogen Chain with State-of-the-Art Many-Body Methods". In: *Physical Review X* 7.3 (Sept. 2017), p. 031059. DOI: 10.1103/PhysRevX.7.031059.

[23] Hendrik J. Monkhorst and James D. Pack. "Special Points for Brillouin-zone Integrations". In: *Physical Review B* 13.12 (June 1976), pp. 5188–5192. DOI: 10.1103/PhysRevB.13.5188.

[24] Hao Shi and Shiwei Zhang. "Some recent developments in auxiliary-field quantum Monte Carlo for real materials". In: *The Journal of Chemical Physics* 154.2 (Jan. 2021), p. 024107. DOI: 10.1063/5.0031024.

[25] Simone Chiesa et al. "Finite-Size Error in Many-Body Simulations with Long-Range Interactions". In: *Physical Review Letters* 97.7 (Aug. 2006), p. 076404. DOI: 10.1103/PhysRevLett.97.076404.

[26] Leo Brewer. *LBL3720: The Cohesive Energies of the Elements*. Tech. rep. Lawrence Berkeley Laboratory, 1977.

[27] S. Lebègue et al. "Cohesive Properties and Asymptotics of the Dispersion Interaction in Graphite by the Random Phase Approximation". In: *Physical Review Letters* 105.19 (Nov. 2010), p. 196401. DOI: 10.1103/PhysRevLett.105.196401.

[28] S. J. Nolan et al. "Calculation of Properties of Crystalline Lithium Hydride Using Correlated Wave Function Theory". In: *Phys. Rev. B* 80.16 (Oct. 2009), p. 165109. DOI: 10.1103/PhysRevB.80.165109.

[29] Michael Scherbela, Leon Gerard, and Philipp Grohs. "Variational Monte Carlo on a Budget - Fine-Tuning Pre-Trained Neural Wavefunctions". In: *Thirty-Seventh Conference on Neural Information Processing Systems*. 2023.

[30] Riccardo Rende et al. "Fine-Tuning Neural Network Quantum States". In: *arXiv preprint* (2024). arXiv: 2403.07795.

[31] Sam Azadi and W. M. C. Foulkes. "Efficient Method for Grand-Canonical Twist Averaging in Quantum Monte Carlo Calculations". In: *Phys. Rev. B* 100.24 (Dec. 2019), p. 245142. DOI: 10.1103/PhysRevB.100.245142.

[32] Ruichen Li et al. "Forward Laplacian: A New Computational Framework for Neural Network-Based Variational Monte Carlo". In: *arXiv preprint* (2023). arXiv: 2307.08214.

[33] Xiang Li et al. "Fermionic Neural Network with Effective Core Potential". In: *Phys. Rev. Res.* 4.1 (Jan. 2022), p. 013021. DOI: 10.1103/PhysRevResearch.4.013021. URL: https://link.aps.org/doi/10.1103/PhysRevResearch.4.013021.

[34] P. P. Ewald. "Die Berechnung Optischer Und Elektrostatischer Gitterpotentiale". In: *Annalen der Physik* 369.3 (1921), pp. 253–287. DOI: 10.1002/andp.19213690304.

[35] Nicholas Gao and Stephan Günnemann. "Generalizing Neural Wave Functions". In: *Proceedings of the 40th International Conference on Machine Learning*. Vol. 202. Proceedings of Machine Learning Research. PMLR, 2023, pp. 10708–10726.

[36] Thom H. Dunning Jr. "Gaussian Basis Sets for Use in Correlated Molecular Calculations. I. The Atoms Boron through Neon and Hydrogen". In: *The Journal of Chemical Physics* 90.2 (Jan. 1989), pp. 1007–1023. ISSN: 0021-9606. DOI: 10.1063/1.456153.

[37] Nicholas Metropolis et al. "Equation of State Calculations by Fast Computing Machines". In: *The Journal of Chemical Physics* 21.6 (June 1953), pp. 1087–1092. ISSN: 0021-9606. DOI: 10.1063/1.1699114.

[38] James Martens and Roger Grosse. *Optimizing Neural Networks with Kronecker-Factored Approximate Curvature*. 2020. arXiv: 1503.05671 [cs.LG].

[39] Pier Luigi Silvestrelli. "Maximally Localized Wannier Functions for Simulations with Supercells of General Symmetry". In: *Physical Review B* 59.15 (Apr. 1999), pp. 9703–9706. ISSN: 0163-1829, 1095-3795. DOI: 10.1103/PhysRevB.59.9703.

[40] Diederik P. Kingma and Jimmy Ba. "Adam: A Method for Stochastic Optimization". In: *arXiv preprint* (2017). arXiv: 1412.6980.

[41] W. A. Diery, Elie A. Moujaes, and R. W. Nunes. “Nature of Localized Phonon Modes of Tilt Grain Boundaries in Graphene”. In: *Carbon* 140 (Dec. 2018), pp. 250–258. ISSN: 0008-6223. DOI: 10.1016/j.carbon.2018.08.045.

[42] Aleksandar Botev and James Martens. *KFAC-JAX*. 2022. URL: https://github.com/google-deepmind/kfac-jax.

Supplementary Information for Transferable Neural Wavefunctions for Solids

S1 Scaling of compute cost with system size

Computing the orbital matrix Φ_{ik} in our ansatz for each electron i and orbital k requires a sum over all nuclei J. Since the number of orbitals and electrons are equal to n_{el} and the number of nuclei N_{atoms} is in a worst-case also equal to n_{el}, materializing this matrix has a worst-case scaling of $\mathcal{O}({n_{\text{el}}}^3)$. This is in contrast to other approaches such as FermiNet, where this matrix is not given as a sum over nuclei and thus only scales as $\mathcal{O}({n_{\text{el}}}^2)$. Because scaling in the limit of $n_{\text{el}} \to \infty$ is in either case dominated by the evaluation of the determinant, which scales as $\mathcal{O}({n_{\text{el}}}^3)$, this does not impact the overall scaling of the method, but can lead to different empirical scaling. Fig. S1 depicts median run-time per optimization step for FermiNet (our implementation) and our approach. We compare timings on chains of Hydrogen atoms of increasing length and dimers of increasing nuclear charge. The former depicts the worst case for our method, the latter is close to the best case. All timings are obtained on 2 A100-GPUs using a batch-size of 512 and 8 determinants.

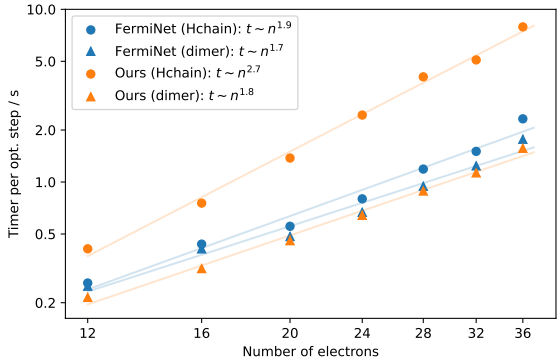

Figure S1: **Scaling of computational cost**: Markers correspond to measured timings, lines correspond to least-square fits of power-laws, with the exponents denoted in the legend.

S2 Limited expressiveness of electron-wise embeddings

Prior work has relied on embeddings $\boldsymbol{h}_i$ for each electron i to construct correlated orbitals. We show that requiring the following three reasonable symmetries already overly constrains which embeddings $\boldsymbol{h}_i$ (and consequently orbitals Φ) can be represented:

Invariance wrt. to continuous translation of *all* particles:

$$h(\boldsymbol{r}_1 + \boldsymbol{\delta}, \dots \boldsymbol{r}_{n_{\text{el}}} + \boldsymbol{\delta}, \boldsymbol{R}_1 + \boldsymbol{\delta}, \dots, \boldsymbol{R}_{N_{\text{atoms}}} + \boldsymbol{\delta}) = h(\boldsymbol{r}_1, \dots, \boldsymbol{r}_{n_{\text{el}}}, \boldsymbol{R}_1, \dots, \boldsymbol{R}_{N_{\text{atoms}}}). \tag{S1}$$

Invariance wrt. to permutation of nuclei of same charge Z:

$$h(\boldsymbol{r}_1, \dots, \boldsymbol{r}_{n_{\text{el}}}, \boldsymbol{R}_1, \dots, \boldsymbol{R}_I, \dots \boldsymbol{R}_J, \dots \boldsymbol{R}_{N_{\text{atoms}}}) = h(\boldsymbol{r}_1, \dots, \boldsymbol{r}_{n_{\text{el}}}, \boldsymbol{R}_1, \dots, \boldsymbol{R}_J, \dots \boldsymbol{R}_I, \dots \boldsymbol{R}_{N_{\text{atoms}}}). \tag{S2}$$

Invariance wrt. to translation of any particle by a supercell lattice vector $\boldsymbol{L}_{\text{sc}}$:

$$h(\boldsymbol{r}_1 + \boldsymbol{L}_{\text{sc}}, \dots, \boldsymbol{r}_{n_{\text{el}}}, \boldsymbol{R}_1, \dots \boldsymbol{R}_{N_{\text{atoms}}}) = h(\boldsymbol{r}_1, \dots, \boldsymbol{r}_{n_{\text{el}}}, \boldsymbol{R}_1, \boldsymbol{R}_{N_{\text{atoms}}}). \tag{S3}$$

To demonstrate the problem consider a simplified 1D example with a single electron and a supercell consisting of N primitive cells, with lattice consant a, each containing a single nucleus. The coordinates R_J of all nuclei in the supercell are thus given by

$$R_J = Ja. \tag{S4}$$

An embedding satisfying the invariances eq. (S1), eq. (S2), eq. (S3) is given by any permutation invariant function h

$$h = h\left((\omega(r - R_1), \ldots, \omega(r - R_{N_{\text{atoms}}})\right) \tag{S5}$$
$$= h(\omega(r - a), \omega(r - 2a), \ldots, \omega(r - Na)), \tag{S6}$$

where ω computes input features that are periodic in the supercell

$$\omega(x) = \omega(x + L_{\text{sc}}) = \omega(x + Na). \tag{S7}$$

Here using distances $r - R$ automatically enforces eq. (S1) and using periodic versions $\omega(r - R)$ of these distances automatically enforces eq. (S3).

For this system, any embedding following the structure in eq. (S5) is necessarily not only invariant under translations of electrons by a *supercell* lattice vector, but also invariant under translation of electrons by a *primitive* lattice vector.

$$h^{\text{shifted}}_{\text{prim}} = h(r + a, R_1, \ldots, R_{N_{\text{atoms}}}) \tag{S8}$$
$$= h\left((\omega(r + a - a), \omega(r + a - 2a), \ldots, \omega(r + a - Na)\right) \tag{S9}$$
$$\overset{\text{S2}}{=} h\left((\omega(r - a), \omega(r - 2a), \ldots, \omega(r)\right) \tag{S10}$$
$$\overset{\text{S3}}{=} h\left((\omega(r - a), \omega(r - 2a), \ldots, \omega(r - Na)\right) \tag{S11}$$
$$= h(r, R_1, \ldots, R_{N_{\text{atoms}}}) = h^{\text{orig}}. \tag{S12}$$

Therefore using electron embeddings with these symmetries allows only representation of orbitals that are periodic on the primitive lattice. This excludes many relevant functions such as localized orbitals and prevents the network from representing long-range correlations. To break this unwanted symmetry there are at least three options:

- Break invariance with respect to permutation of nuclei. Non-transferable ansätze such as FermiNet [6], PsiFormer [10] or DeepSolid [18] all break this permutation invariance. Since these approaches are only ever trained on a single system (and thus a single permutation of nuclei) this poses no issue there, but prevents efficient generalization to permuted, but physically identical systems.
- Break supercell lattice translational symmetry. For gas-phase calculations there is no periodicity and thus existing transferable approaches [21, 29, 35] do not face this issue. For periodic systems however periodicity is required to be able to enforce boundary conditions.
- Use permutation *equivariant* electron-ion embeddings instead of permutation *invariant* electron embeddings, as done in this work.

S3 Hyperparameters

A detailed description of the hyperparameter used in this work can be found below (cf. tab. 1). For optimization we rely on the second-order method KFAC [38] and use their Python implementation [42].

Table 1: Hyperparameter settings used in this work

HF-pre-training	Pre-training basis set	cc-pVDZ
	Pre-training steps per geometry	100-500
Embedding	Envelope power γ	2
	Uniform initialization of envelope scaling α	8-10
	El-el hidden dimension	32
	El-Ion hidden dimension	128
	Ion hidden dimension	128
	№ hidden layers of ion embedding	3
	Final el-ion embedding dimension e_{iI}	64
Transferable atomic orbitals	№ determinants n_{det}	8
	Basis set for orbital descriptor	cc-pVDZ
	№ hidden layers of f^W	2
	Hidden dimension of f^W	128
	№ hidden layers of f^a	2
	Hidden dimension of f^a	32
	Activation function	ReLU
	Residual connection	True
	Layer Norm	True
Jastrow factor	№ hidden layers of MLP	2
	Hidden dimension of MLP	40
Markov Chain Monte Carlo	№ walkers	2048
	№ decorrelation steps	20
	Target acceptance prob.	50%
Variational optimization	Optimizer	KFAC
	Damping	$1 - 3 \times 10^{-3}$
	Norm constraint	1×10^{-3}
	Batch size	2048
	Initial learning rate lr_0	$0.1 - 0.3$
	Learning rate decay	$\text{lr}(t) = \text{lr}_0(1 + t/6000)^{-1}$
	Optimization steps	100,000 - 200,000
Changes for Reuse	Initial learning rate lr_0	0.05
	Learning rate decay	$\text{lr}(t) = \text{lr}_0(1 + t/6000)^{-1}$
	Optimization steps	0 - 10,000

S4 Total energies

For better comparison we add the total energies of the results in Tab. 1 and Fig. 3. For Graphene (see Sec. 2.2) we state the total energies per primitive cell for each twist plus the combination of structure-factor-based finite size corrections and ZPVE in Tab. 2. For LiH (see Sec. 2.3) we state the twist averaged energies per primitive cell for each lattice constant, separately depicting structure-factor-based finite size correction and ZPVE in Tab. 3

Table 2: **Total energies of graphene** in Hartrees. The energies depict the total energies per primitive cell by sequentially adding structure-factor-based finite-size correction (SFC) and ZPVE. The systems represent the symmetry-inequivalent twists for the 12×12 Monkhorst-Pack grid.

Twists	**Total Energy**	**Total energy + SFC**	**Total energy + SFC + ZPVE**
$\boldsymbol{k} = (0.33, 0.33)$	-76.2572	-76.2543	-76.2415
$\boldsymbol{k} = (0, -0.17)$	-76.2002	-76.1972	-76.1844
$\boldsymbol{k} = (-0.08, -0.42)$	-76.2679	-76.2650	-76.2522
$\boldsymbol{k} = (-0.08, -0.33)$	-76.2470	-76.2441	-76.2313
$\boldsymbol{k} = (-0.08, -0.25)$	-76.2190	-76.2160	-76.2032
$\boldsymbol{k} = (-0.17, -0.42)$	-76.2615	-76.2586	-76.2458
$\boldsymbol{k} = (0.67, 0.33)$	-76.2590	-76.2560	-76.2432
$\boldsymbol{k} = (-0.17, -0.58)$	-76.2775	-76.2746	-76.2618
$\boldsymbol{k} = (-0.08, -0.17)$	-76.1914	-76.1884	-76.1756
$\boldsymbol{k} = (-0.08, -0.50)$	-76.2786	-76.2757	-76.2629
$\boldsymbol{k} = (0, 0)$	-76.1572	-76.1542	-76.1414
$\boldsymbol{k} = (0, -0.25)$	-76.2308	-76.2278	-76.2150
$\boldsymbol{k} = (-0.17, -0.33)$	-76.2407	-76.2377	-76.2249
$\boldsymbol{k} = (-0.25, -0.50)$	-76.2688	-76.2658	-76.2530
$\boldsymbol{k} = (0, -0.08)$	-76.1723	-76.1693	-76.1565
$\boldsymbol{k} = (0, -0.50)$	-76.2787	-76.2758	-76.2630
$\boldsymbol{k} = (0, -0.42)$	-76.2737	-76.2708	-76.2580
$\boldsymbol{k} = (-0.25, -0.58)$	-76.2728	-76.2699	-76.2571
$\boldsymbol{k} = (-0.17, -0.50)$	-76.2740	-76.2710	-76.2582
Averaged 3×3	-76.2465	-76.2435	-76.2307
Averaged 12×12	-76.2503	-76.2473	-76.2345

Table 3: **Total energies of LiH** in Hartrees. The energies depict the twist averaged total energies per primitive cell by sequentially adding structure-factor-based finite-size correction (SFC) and ZPVE. For the twist averaging we use a $5 \times 5 \times 5$ Monkhorst-Pack grid per lattice constant. The energies accompanies the Fig. 3.

Supercell size	**Lattice constant / a_0**	**Total Energy**	**Total energy + SFC**	**Total energy + SFC + ZPVE**
$2 \times 2 \times 2$	6.4	-8.1654	-8.1462	-8.1338
	6.8	-8.1752	-8.1574	-8.1464
	7.2	-8.1792	-8.1626	-8.1529
	7.6	-8.1789	-8.1636	-8.1554
	7.9	-8.1759	-8.1618	-8.1545
	8.3	-8.1709	-8.1580	-8.1522
	8.7	-8.1646	-8.1527	-8.1482
	9.1	-8.1574	-8.1467	-8.1436
$3 \times 3 \times 3$	7.674	-8.1644	-8.1604	-8.1524