A Theoretical Framework for an Efficient Normalizing Flow-Based Solution to the Electronic Schrödinger Equation

Daniel Freedman[1], **Eyal Rozenberg**[2], **Alex Bronstein**[2]

[1]Independent Researcher
[2]Technion - Israel Institute of Technology

Abstract

A central problem in quantum mechanics involves solving the Electronic Schrödinger Equation for a molecule or material. The Variational Monte Carlo approach to this problem approximates a particular variational objective via sampling, and then optimizes this approximated objective over a chosen parameterized family of wavefunctions, known as the ansatz. Recently neural networks have been used as the ansatz, with accompanying success. However, sampling from such wavefunctions has required the use of a Markov Chain Monte Carlo approach, which is inherently inefficient. In this work, we propose a solution to this problem via an ansatz which is cheap to sample from, yet satisfies the requisite quantum mechanical properties. We prove that a normalizing flow using the following two essential ingredients satisfies our requirements: (a) a base distribution which is constructed from Determinantal Point Processes; (b) flow layers which are equivariant to a particular subgroup of the permutation group. We then show how to construct both continuous and discrete normalizing flows which satisfy the requisite equivariance. We further demonstrate the manner in which the non-smooth nature ("cusps") of the wavefunction may be captured, and how the framework may be generalized to provide induction across multiple molecules. The resulting theoretical framework entails an efficient approach to solving the Electronic Schrödinger Equation.

Extended version — https://arxiv.org/abs/2406.00047

1 Introduction

The Electronic Schrödinger Equation A central problem in quantum mechanics involves solving the Electronic Schrödinger Equation to compute the ground state energy and wavefunction of a molecule or material. This problem has manifold applications in chemistry, condensed matter physics, and materials science. However, solving the Electronic Schrödinger Equation can be tricky: analytical solutions only exist for certain very simple problems; and numerical solutions must cope with the curse of dimensionality, due to the fact that the wavefunction's dimensionality grows linearly with the number of electrons.

Variational Monte Carlo A standard computational approach to this problem is based on Variational Monte Carlo (Ceperley and Alder 1986; Austin, Zubarev, and Lester Jr 2012; Gubernatis, Kawashima, and Werner 2016; Foulkes et al. 2001; Needs et al. 2009). Solving for the ground state is equivalent to finding the eigenfunction of the Hamiltonian corresponding to the smallest eigenvalue (energy); this problem can be posed as the minimization of a Rayleigh quotient. The issue is that both the numerator and denominator of the Rayleigh quotient correspond to very high dimensional integrals. Variational Monte Carlo approximates these integrals via sampling, and the approximated objective is optimized over a family of wavefunctions, yielding an upper bound on the ground state energy. The heart of this method is therefore the wavefunction family, also known as the ansatz: the more expressive the ansatz, the tighter the upper bound will be, yielding better approximations to the ground state energy.

Neural Networks as the Ansatz Recent work has proposed using neural networks as a flexible ansatz, and has achieved very high quality results. The network must be adapted to respect the properties of electronic wavefunctions – particularly antisymmetry, as electrons are Fermions. Important examples of this type of ansatz are PauliNet (Hermann, Schätzle, and Noé 2020; Schätzle, Hermann, and Noé 2021) and FermiNet (Pfau et al. 2020; Spencer et al. 2020), both of which attain excellent ground state energies (e.g. FermiNet achieves 99.8% of the correlation energy for boron atoms).

The Problem: Sampling Inefficiency In order to be able to apply the Variational Monte Carlo formalism to the ansätze just described, such as PauliNet or FermiNet, one must be able to sample from the densities corresponding to the wavefunctions given by their neural networks. In general, this is only possibly using Markov Chain Monte Carlo (MCMC) techniques such as Langevin Monte Carlo (Umrigar, Nightingale, and Runge 1993) or any of several variations. The issue with using such MCMC approaches to sampling is that they are inherently time-consuming: each sample is itself the solution of a stochastic differential equation as time goes to infinity.

Goals and Contributions The main goal of this paper is to solve the problem of sampling inefficiency, thereby yielding faster algorithms for solving the Electronic Schrödinger Equation. We achieve this goal by specifying a wavefunction ansatz which is easy to sample from, yet satisfies the requisite quantum mechanical properties. In particular, we use an ansatz based on specially designed normalizing flows. More specifically, we provide the following contributions:

- We establish that an ansatz satisfying our desiderata can be instantiated as a normalizing flow with these characteristics: (a) its base distribution is symmetric under a particular subgroup of the permutation group, and vanishes for identical electrons; (b) the flow transformation is equivariant to the same subgroup of the permutation group.
- We show that the base distribution can be constructed using a particular combination of Determinantal Point Processes.
- We construct both continuous and discrete normalizing flows obeying the requisite equivariance.
- We provide a training regimen based on standard stochastic gradient descent.
- We show how to accommodate cusps, which encapsulate non-smooth aspects of the wavefunction.
- We generalize the framework so that induction across multiple molecules may be accommodated, while including the necessary additional invariances, in particular rigid motion invariance.

We end by noting that our contributions are of a theoretical character. Our interest is to elucidate a theoretical approach to solving the Electronic Schrödinger Equation based on neural networks, which retains both the expressivity of the neural network function class while at the same time providing considerably greater efficiency. These contributions should be of importance and interest to the growing community of researchers working on the boundary of machine learning and quantum physics; including those researchers whose work involves the practical computation of ground state energies.

2 Related Work

Neural Networks and Physics We begin by noting that neural networks have recently found broad and quite varied use in physics problems. Amongst others, examples include solution of physics-based partial differential equations (Raissi, Perdikaris, and Karniadakis 2019; Li et al. 2020; Lu et al. 2021); density functional theory (Ryczko, Strubbe, and Tamblyn 2019; Kirkpatrick et al. 2021); topological states (Deng, Li, and Das Sarma 2017a); quantum state tomography (Torlai et al. 2018); and quantum optics (Rozenberg et al. 2021, 2022). Useful surveys include (Carleo et al. 2019; Zhang et al. 2023).

Pure Spin Systems Various works have used neural networks as the ansatz in the case of pure-spin systems, sometimes also referred to as "discrete space systems", in which the spins are at fixed locations. These include (Carleo and Troyer 2017; Deng, Li, and Das Sarma 2017b; Gao and Duan 2017; Levine et al. 2019; Sharir, Shashua, and Carleo 2022; Passetti et al. 2023).

Continuous Space Systems In terms of continuous space problems of the sort that interest us, DeepWF (Han, Zhang, and Weinan 2019) bases its on ansatz on the classical Slater-Jastrow formalism, but learns both the symmetric and antisymmetric parts; the latter contains only two-electron terms, limiting the accuracy. PauliNet (Hermann, Schätzle, and Noé 2020; Schätzle, Hermann, and Noé 2021) also bases its ansatz on the Slater-Jastrow-Backflow form, but does so in a way that captures many-electron interactions, while respecting permutation-equivariance; this, as well as the inclusion of cusp terms, leads to much higher accuracy (e.g. 97.3% of the correlation energy for boron atoms). FermiNet (Pfau et al. 2020; Spencer et al. 2020) attains still higher accuracy (e.g. 99.8% of the correlation energy for boron atoms) by using an appropriately designed neural network to represent the entire wavefunction, which contains a generalization of Slater determinants to account for all-electron interactions. A hybrid solution which improves upon both PauliNet and FermiNet is presented in (Gerard et al. 2022). Techniques for learning / induction across several molecules or materials at once are presented in (Gao and Günnemann 2023; Scherbela, Gerard, and Grohs 2024; Gerard et al. 2024). We briefly mention applications to periodic systems (Wilson et al. 2022; Li, Li, and Chen 2022; Pescia et al. 2022; Cassella et al. 2023); techniques that use Diffusion Monte Carlo (Wilson et al. 2021; Ren et al. 2023); and methods that deal with excited states (Entwistle et al. 2023; Pfau et al. 2023; Naito et al. 2023).

Normalizing Flows and Quantum Problems There is a body of work which has applied normalizing flows to problems in lattice field theories, e.g. (Kanwar et al. 2020; Boyda et al. 2021; Abbott et al. 2023b,a); due to the different physical scenario, the setup in these papers is naturally quite different from ours. The work by (Xie, Zhang, and Wang 2022) introduced the use of normalizing flows in the context of Fermionic problems; follow on papers for specific applications include (Xie, Zhang, and Wang 2023; Saleh et al. 2023). Our paper considerably broadens this approach, by (a) establishing all results rigorously, as in Theorems 1 - 6; (b) showing how to practically implement the flows with discrete layers, as in Theorem 7; (c) providing important extensions to deal with phase computation (see Theorem 9), the crucial issue of cusps (see Theorem 10), and induction across multiple molecules (see Theorems 11 - 12). Finally, we mention two further papers: (Stokes, Chen, and Veerapaneni 2023) focuses on geometric aspects of the problem; while WaveFlow (Thiede, Sun, and Aspuru-Guzik 2022), develops a specialized normalizing flow architecture for one-dimensional systems, which is interesting, albeit of limited applicability.

3 Problem Setup

3.1 Goals

The Setting Our overall goal is to compute the ground state wavefunction and energy of a molecule given its molecular parameters and spin multiplicity. Denote $x_i = (r_i, s_i)$ to be the pair consisting of the position and spin for the i^{th} electron; x will denote the entire ordered list $(x_1, \ldots, x_n)$, with corresponding definitions for r and s. We specify wavefunctions as $\psi(x)$; due to the fact that electrons are Fermions, valid wavefunctions must be *antisymmetric*, that is if $\pi \in \mathbb{S}_n$ is a permutation, then

$$\psi(\pi x) = (-1)^{\pi}\psi(x) \tag{1}$$

where as usual, $(-1)^{\pi}$ is shorthand for $(-1)^{N(\pi)}$ where $N(\pi)$ is the minimal number of flips to produce π.

Let R_I and Z_I denote the position and atomic number of the I^{th} nucleus, and let the Laplacian for the i^{th} electron

be $\Delta_i = \frac{\partial^2}{\partial r_{i1}^2} + \frac{\partial^2}{\partial r_{i2}^2} + \frac{\partial^2}{\partial r_{i3}^2}$, and denote the sum of the Laplacians by $\Delta = \sum_i \Delta_i$; then the Hamiltonian is given by

$$H = -\tfrac{1}{2}\Delta + V(x) \tag{2}$$

where the potential $V(x)$ is given by

$$V(x) = \sum_{i>j} \frac{1}{\|r_i - r_j\|} - \sum_{iI} \frac{Z_I}{\|r_i - R_I\|} + \sum_{I>J} \frac{Z_I Z_J}{\|R_I - R_J\|} \tag{3}$$

Our goal is to compute the ground state wavefunction, which we denote as $\psi_0(x)$ and corresponding ground state energy E_0. They may be computed using the variational principle, i.e. by minimizing the Rayleigh quotient:

$$\psi_0 = \operatorname*{argmin}_{\psi \in \Psi} \frac{\langle \psi | H | \psi \rangle}{\langle \psi | \psi \rangle} \quad \text{and} \quad E_0 = \frac{\langle \psi_0 | H | \psi_0 \rangle}{\langle \psi_0 | \psi_0 \rangle} \tag{4}$$

where Ψ is the set of all possible valid wavefunctions, and H is the Hamiltonian. If we specify the wavefunction ansatz as a neural network with parameters θ, this becomes

$$\theta^* = \operatorname*{argmin}_{\theta} \frac{\langle \psi(\cdot;\theta) | H | \psi(\cdot;\theta) \rangle}{\langle \psi(\cdot;\theta) | \psi(\cdot;\theta) \rangle} \tag{5}$$

and

$$E^* = \frac{\langle \psi(\cdot;\theta^*) | H | \psi(\cdot;\theta^*) \rangle}{\langle \psi(\cdot;\theta^*) | \psi(\cdot;\theta^*) \rangle} \geq E_0 \tag{6}$$

That is, we compute an upper bound E^* to the ground state energy E_0. The more expressive the ansatz, the tighter the bound will be.

Variational Monte Carlo The issue with the formulation to this point is the need to compute the inner products in Equations (4) and (6), which correspond to very high-dimensional integrals. A standard solution to this problem is based on a Monte Carlo scheme. To begin with, let us define the local energy $\mathcal{E}(x)$ and its real part $\mathcal{E}_r(x)$ as

$$\mathcal{E}(x) \equiv \frac{H\psi(x)}{\psi(x)} = -\frac{\Delta\psi(x)}{2\psi(x)} + V(x), \quad \mathcal{E}_r(x) = \mathfrak{Re}\{\mathcal{E}(x)\} \tag{7}$$

In this case, one can simplify the minimand in Equation (4) (see the Extended Version) as

$$\frac{\langle \psi | H | \psi \rangle}{\langle \psi | \psi \rangle} = \mathbb{E}_{x \sim \rho(\cdot)} \left[\mathcal{E}_r(x) \right] \approx \frac{1}{K} \sum_{k=1}^{K} \mathcal{E}_r \left(x^{(k)} \right) \tag{8}$$

where the $x^{(k)}$ are sampled from

$$\rho(x) = \frac{|\psi(x)|^2}{\langle \psi | \psi \rangle}. \tag{9}$$

3.2 The General Approach

As detailed in Sections 1 and 2, a number of recent works have followed the above approach using a variety of neural networks as the ansatz for the wavefunction $\psi(\cdot;\theta)$. In order to do so, one must be able to sample from $\rho(x;\theta) = |\psi(x;\theta)|^2 / \langle \psi | \psi \rangle$; and as the networks are quite general, the only feasible method for sampling is a Markov Chain Monte Carlo technique such as Langevin Monte Carlo (Umrigar, Nightingale, and Runge 1993) or any of several variations. This can be time-consuming, as each sample is the solution of a stochastic differential equation as time goes to infinity.

A solution to this problem presents itself if we can somehow specify a wavefunction $\psi(x)$ which is easy to sample from. We are interested in wavefunctions which satisfy the following three properties:

(W1) There is an explicit functional form for the wavefunction $\psi(x)$.

(W2) ψ is antisymmetric.

(W3) We can sample non-iteratively (in constant time) from $|\psi(\cdot)|^2$.

The first two properties are necessary for any form of Variational Monte Carlo: (W1) allows us to evaluate the local energy $\mathcal{E}_r$ in (7) for use in (8); and (W2) is required for valid electronic (Fermionic) wavefunctions. But (W3) is the new ingredient: if we have a family of wavefunctions ψ satisfying (W1)-(W3), then solving the minimization in (6) via the Monte Carlo approach in (8) will be considerably accelerated, as each sample will only require constant time to generate. We add a fourth property, which is not strictly necessary but is both desirable and will prove useful:

(W4) ψ is normalized, that is $\int |\psi(x)|^2 dx = 1$.

It turns out that generating such wavefunctions is possible using the following procedure:

Theorem 1. *Let $\rho(\cdot)$ be a probability density function which we can sample from in constant time. Let $\rho(\cdot)$ satisfy two additional properties:*

(D1) $\rho(x)$ is symmetric: $\rho(\pi x) = \rho(x)$ for all permutations $\pi \in \mathbb{S}_n$.

(D2) $\rho(x) = 0$ if $x_i = x_j$ for any i, j.

Finally, let $\kappa(x)$ be a complex function which satisfies $|\kappa(x)| = 1 \ \forall x$, and is nearly antisymmetric:

$$\kappa(\pi x) = \begin{cases} (-1)^{\pi} \kappa(x) & \text{if } x_i \neq x_j \text{ for all } i, j \\ \bar{\kappa} & \text{otherwise} \end{cases} \tag{10}$$

where $\bar{\kappa} \in \mathbb{C}$ is an arbitrary value with $|\bar{\kappa}| = 1$. Then ψ satisfies (W1)-(W4) if and only if ψ can be written as $\psi(x) = \kappa(x)\sqrt{\rho(x)}$ with κ and ρ satisfying the above-stated properties.

Proof: See the Extended Version.

The general idea expressed in Theorem 1 is that we can build the wavefunction ψ out of an easy-to-sample-from density function satisfying additional properties (D1)-(D2); and a nearly antisymmetric phase function κ. In what follows, we will show how to construct both of these ingredients. But before doing so, we take a short detour to address the most important practical scenario, that of fixed spin multiplicity.

3.3 Fixed Spin Multiplicity

Notation As in most approaches to this problem, we assume that the spin multiplicity of the molecule is specified, which is equivalent to fixing the number of spin up and spin down electrons, denoted n_u and n_d respectively, with $n_u + n_d = n$. Define the *canonical spin vector* to be given by $\bar{s} = [\uparrow, \ldots, \uparrow$

$,\downarrow,\ldots,\downarrow]$, i.e. the first n_u are $\uparrow$, the last n_d are $\downarrow$. We let the sets of indices of up and down spin electrons for the canonical spin vector be denoted by $\mathcal{N}_u = \{1,\ldots,n_u\}$ and $\mathcal{N}_d = \{n_u+1,\ldots,n\}$. Finally, we will be interested in the subgroup of permutations in which a permutation is applied separately to spin-up and spin-down electrons. We denote this subgroup by

$$\mathbb{G} \equiv \mathbb{S}_{\mathcal{N}_u} \times \mathbb{S}_{\mathcal{N}_d} \qquad (\mathbb{G} \text{ is a subgroup of } \mathbb{S}_n) \tag{11}$$

Specification of the Density In the case of fixed spin multiplicity, the specification of the density $\rho(x)$ is simplified:

Theorem 2. *Given a configuration $x = (r, s)$, let a permutation which maps the spin vector s to the canonical spin vector $\bar{s}$ be given by $\bar{\pi}_s$, i.e. $\bar{s} = \bar{\pi}_s s$. Let $\bar{\rho}(r)$ be a density function on electron positions (i.e. no spins) satisfying*

(R1) $\bar{\rho}$ is $\mathbb{G}$-invariant: $\bar{\rho}(\pi r) = \bar{\rho}(r)$ for all $\pi \in \mathbb{G}$
(R2) $\bar{\rho}(r) = 0$ if $r_i = r_j$, for $i, j \in \mathcal{N}_u$ or $i, j \in \mathcal{N}_d$

A density $\rho(x) = \rho(r, s)$ satisfies conditions (D1)-(D2) in Theorem 1 if and only if it may be written as $\rho(r, s) = \bar{\rho}(\bar{\pi}_s r)$ for a density $\bar{\rho}(r)$ satisfying conditions (R1) and (R2).

Proof: See the Extended Version.

To summarize: in the case of fixed spin multiplicity, specifying a wavefunction ψ satisfying our desired conditions (W1)-(W4) is equivalent to specifying a density $\bar{\rho}(r)$ satisfying conditions (R1)-(R2); and then applying the transformations given in Theorems 1 and 2 to map from $\bar{\rho}$ to ψ.[1] Therefore, henceforth we will focus exclusively on specifying densities $\bar{\rho}(r)$ satisfying conditions (R1)-(R2). To avoid unnecessary notational complexity, we will drop the bars and simply write $\rho(r)$.

4 Using Normalizing Flows to Construct the Wavefunction Ansatz

4.1 Sufficient Properties of the Normalizing Flow's Base Density and Transformation

Our goal is to use a normalizing flow to construct the density $\rho(r)$. Let D be the ambient dimension (i.e. $D = 3$) and n be the number of electrons. The relevant vectors will live in the space $\mathbb{R}^{Dn}$ construed as the Cartesian product $\mathbb{R}^D \times \cdots \times \mathbb{R}^D$ (which is of course isomorphic to $\mathbb{R}^{Dn}$). A normalizing flow will consist of two ingredients: (1) a base random variable z, which lives in $\mathbb{R}^{Dn}$, and is described by the density $\rho_z(z)$; (2) an invertible transformation $T : \mathbb{R}^{Dn} \to \mathbb{R}^{Dn}$, such that $r = T(z)$. In this case, the density $\rho(r)$ is the push-forward of ρ_z along T, and is given by the change of variables formula

$$\rho(r) = \rho_z(T^{-1}(r))|\det J_{T^{-1}}(r)| \tag{12}$$

Recall that we would like our density $\rho(r)$ to satisfy conditions (R1)-(R2) laid out in Theorem 2. The following theorem establishes conditions for this to occur:

Theorem 3. *Suppose that we have a normalizing flow, whose base density ρ_z satisfies properties (R1) and (R2) from Theorem 2, and whose transformation T is $\mathbb{G}$-equivariant. Then the density resulting from the normalizing flow will satisfy properties (R1) and (R2).*

[1] We have for the moment ignored the issue of the phase κ, which we return to in Sections 4.6 and 5.1.

Proof: See the Extended Version.

Armed with this key result, we now set out to design the base density ρ_z and transformation T which satisfy the conditions of Theorem 3.

4.2 Base Density: Determinantal Point Processes

In most cases in machine learning, the base density for a normalizing flow is taken to be a standard distribution, most often a Gaussian. In our case, we require that the base density have certain special properties, namely (R1) and (R2) from Theorem 2. It turns out that Determinantal Point Processes (DPPs) have just the properties we require. In particular, we are interested in the class of DPPs known as Projection DPPs (Gautier, Bardenet, and Valko 2019; Lavancier, Møller, and Rubak 2015), which can be specified as follows. We will let y specify a generic point in $\mathbb{R}^D$. Let $h_k : \mathbb{R}^D \to \mathbb{R}$ for $k = 1,\ldots,n$ be a set of n functions which are orthogonal, that is $\langle h_i, h_j \rangle = \int_{\mathbb{R}^D} h_i(y) h_j(y) dy = \delta_{ij}$. Let $H(y)$ be the column vector composed by stacking the individual functions $h_i(y)$ and define the kernel function as $K(y, y') = H(y)^T H(y')$. Then for a given collection of n points in $\mathbb{R}^D$, that is $r = (r_1,\ldots,r_n)$, we define the $n \times n$ kernel matrix $\mathbf{K}_n(r)$:

$$\mathbf{K}_n(r) = \begin{bmatrix} K(r_1, r_1) & \ldots & K(r_1, r_n) \\ \vdots & \ddots & \vdots \\ K(r_n, r_1) & \ldots & K(r_n, r_n) \end{bmatrix} \tag{13}$$

Using this kernel matrix, we specify the density of the Projection DPP as follows:

$$\rho_{dpp}(r; n) = \frac{1}{n!} \det \mathbf{K}_n(r) \tag{14}$$

Since $\mathbf{K}_n(r)$ is positive semi-definite, it follows that its determinant is non-negative so that $\rho_{dpp}(r; n)$ is non-negative, as desired. A proof that $\rho_{dpp}(r; n)$ is properly normalized (i.e. integrates to 1) can be found, for example, in Proposition 2.10 of (Johansson 2006).

Given the notion of a Projection DPP, we may define the base density as follows. As above, let the base random variable be z, where z can be broken into spin-up and spin-down pieces, denoted z_u and z_d. (Specifically, z_u and z_d are the parts of z corresponding to electrons in $\mathcal{N}_u$ and $\mathcal{N}_d$, respectively.) The base density can then be constructed by taking

$$\rho_z(z) = \rho_{dpp}(z_u; n_u)\rho_{dpp}(z_d; n_d) \tag{15}$$

That is, z_u and z_d are chosen from two independent Projection DPPs. We then have the following theorem:

Theorem 4. *Let ρ_z be the density specified in Equation (15). Then ρ_z satisfies conditions (R1) and (R2) from Theorem 2.*

Proof: See the Extended Version.

We therefore have an explicit form for the base density from Equations (14) and (15). Furthermore, sampling from the base density amounts to sampling from two independent Projection DPPs. A sampling procedure for Projection DPPs is specified in the Extended Version.

We note that it is possible to make the base density itself learnable. This is achieved by making each Projection DPP

learnable, through the introduction of learnable orthogonal functions $h_k : \mathbb{R}^D \to \mathbb{R}$, that is $h_k(y;\theta)$, from which the kernel function $K(y, y';\theta) = H(y;\theta)^T H(y;\theta)$ becomes learnable. In order to retain orthogonality, one may proceed as follows. Let $B_\theta : \mathbb{R}^D \to \mathbb{R}^n$ be a specified by a network with parameters θ, for example a Multilayer Perceptron (in which case B_θ is a type of neural field), and let $H(y;\theta) = AB_\theta(y)$ for a square $n \times n$ matrix A. Let the Gram matrix of the network B_θ be given by Ξ_θ, that is $(\Xi_\theta)_{ij} = \langle (B_\theta)_i, (B_\theta)_j \rangle$. If the eigendecomposition of Ξ_θ is given by $\Xi_\theta = U_\theta \Lambda_\theta U_\theta^T$, then orthogonality of $H(\cdot;\theta)$ is achieved by choosing $A = A_\theta = \Lambda_\theta^{-1/2} U_\theta^T$.

4.3 $\mathbb{G}$-Equivariant Layers

As noted in Section 4, we require the normalizing flow transformation to be $\mathbb{G}$-equivariant. Of course, chaining together many layers which are each $\mathbb{G}$-equivariant results in an overall transformation which is also $\mathbb{G}$-equivariant. Now, suppose that a particular layer ℓ can be written as

$$r^{\ell+1} = T^\ell(r^\ell) \tag{16}$$

where $r^\ell = (r_1^\ell, \dots, r_n^\ell)$ and likewise for $r^{\ell+1}$. We will need to see the action on the spin-up and spin-down electrons separately, so we denote $r_u^\ell = (r_i^\ell)_{i \in \mathcal{N}_u}$ and $r_d^\ell = (r_i^\ell)_{i \in \mathcal{N}_d}$; and we may write

$$r_u^{\ell+1} = T_u^\ell(r_u^\ell, r_d^\ell) \quad \text{and} \quad r_d^{\ell+1} = T_d^\ell(r_u^\ell, r_d^\ell) \tag{17}$$

For notational convenience, we use $\alpha \in \{u, d\}$ to denote the spin, and the complement of the spin is given by $\hat{\alpha}$ (i.e. if $\alpha = u$ then $\hat{\alpha} = d$ and vice-versa). Then we have the following theorem:

Theorem 5. *The transformation T^ℓ is $\mathbb{G}$-equivariant if and only if*

$$T_\alpha^\ell(\pi_\alpha r_\alpha^\ell, \pi_{\hat{\alpha}} r_{\hat{\alpha}}^\ell) = \pi_\alpha T_\alpha^\ell(r_\alpha^\ell, r_{\hat{\alpha}}^\ell) \qquad \alpha \in \{u, d\} \tag{18}$$

That is, T_α^ℓ is equivariant with respect to r_α^ℓ, and invariant with respect to $r_{\hat{\alpha}}^\ell$.

Proof: See the Extended Version.

We now show how to specify continuous and discrete normalizing flows satisfying Theorem 5.

4.4 Continuous Normalizing Flows

According to Theorem 3, we are required a find a transformation which is $\mathbb{G}$-equivariant. We now show this can be achieved via a continuous normalizing flow. We specify this flow via the ordinary differential equation (ODE)

$$\frac{dv}{dt} = \Gamma_t(v), \quad \text{with} \quad v(0) = z \sim \rho_z(\cdot) \quad \text{and} \quad r = v(1) \tag{19}$$

That is, the transformation $r = T(z)$ is derived as follows: the initial condition is sampled from the base density; and r is gotten by integrating the ODE forward to time $t = 1$. Γ's t-dependence is indicated via a subscript for notational convenience. We then have the following theorem:

Theorem 6. *Let the transformation $r = T(z)$ be specified as in Equation (19). Then T is $\mathbb{G}$-equivariant if Γ_t is $\mathbb{G}$-equivariant for all t.*

Proof: See the Extended Version.

It therefore suffices to design a $\mathbb{G}$-equivariant function Γ_t. Let us break this down by spin: from Theorem 5, we know that this implies that for all t, we have that $\Gamma_t(\pi_\alpha r_\alpha, \pi_{\hat{\alpha}} r_{\hat{\alpha}}) = \pi_\alpha \Gamma_t(r_\alpha, r_{\hat{\alpha}})$ for $\alpha \in \{u, d\}$. We show in the Extended Version how to implement a layer of Γ with a combination of multihead attention, fully connected layers, and linear projections (Γ can be composed of many such layers).

Continuous normalizing flows are elegant; however, they can present some numerical difficulties. In particular, the issue of ODE stiffness frequently arises in deep learning pipelines involving continuous normalizing flows. Thus, we now present an alternative method, based on discrete normalizing flows.

4.5 Discrete Normalizing Flows

Our goal is now to design such functions T_u^ℓ and T_d^ℓ which satisfy Equation (18), and for which the overall transformation $T^\ell = (T_u^\ell, T_d^\ell)$ is invertible. The goal of the layer we propose here is to *not sacrifice on expressivity*, especially when compared to many layers which are designed for discrete normalizing flows. In particular, the main issue will be to show that the expressivity can be retained even with the joint requirements of invertibility and $\mathbb{G}$-equivariance. We note that the kind of transformation we propose below is not generally used for normalizing flows, as the determinant of its Jacobian is not fast to compute; however, this is not an issue in our case, as the dimension of the spaces we are dealing with is moderate in size. For a more detailed discussion, see the Extended Version.

To solve this problem, we introduce the Split Subspace Layer; we note that this layer may be of broader interest in machine learning, independent of the current setting. As before, we take D to represent the ambient spatial dimension; in our case, $D = 3$. A key parameter for the ℓ^{th} layer will be the orthogonal matrix $\Lambda_\alpha^\ell \in O(D)$; in particular, we divide this matrix into 2 pieces

$$\Lambda_\alpha^\ell = [\beta_\alpha^\ell, \xi_\alpha^\ell] \text{ with } \beta_\alpha^\ell \in \mathbb{R}^{D \times D_\beta} \text{ and } \xi_\alpha^\ell \in \mathbb{R}^{D \times (D - D_\beta)} \tag{20}$$

That is, β_α^ℓ represents the first D_β columns of Λ_α^ℓ, and ξ_α^ℓ represents the final $D - D_\beta$ columns. For each electron i, we compute the inner product of its coordinates with β_α^ℓ, i.e.

$$\gamma_{\alpha,i}^\ell = (\beta_\alpha^\ell)^T r_{\alpha,i}^\ell \quad \text{so that } \gamma_{\alpha,i}^\ell \in \mathbb{R}^{D_\beta} \tag{21}$$

We can collect the individual vectors $\gamma_{\alpha,i}^\ell$ into a list $\gamma_\alpha^\ell = (\gamma_{\alpha,i}^\ell)_{i \in \mathcal{N}_\alpha}$. Given this, we define the Split Subspace Layer T_α^ℓ on a per-electron basis by

$$r_{\alpha,i}^{\ell+1} = T_{\alpha,i}^\ell(r_\alpha^\ell, r_{\hat{\alpha}}^\ell) = r_{\alpha,i}^\ell + \xi_\alpha^\ell \varphi_{\alpha,i}^\ell(\gamma_\alpha^\ell, \gamma_{\hat{\alpha}}^\ell) \tag{22}$$

where φ_α^ℓ is a network, $\varphi_{\alpha,i}^\ell$ is the part of (the output of) φ_α^ℓ corresponding to the i^{th} electron, and $\varphi_{\alpha,i}^\ell(\gamma_\alpha^\ell, \gamma_{\hat{\alpha}}^\ell) \in \mathbb{R}^{D - D_\beta}$. A crucial aspect of this φ_α^ℓ network is that it captures dependencies between all electrons. This can been seen

by examining Equation (22): $\varphi^\ell_{\alpha,i}$ depends on both γ^ℓ_α and $\gamma^\ell_{\hat\alpha}$. But recall that γ^ℓ_α is a list of the vectors $\gamma^\ell_{\alpha,i}$ for each electron i with spin α, and $\gamma^\ell_{\hat\alpha}$ is the corresponding list for electrons with the complement spin $\hat\alpha$. As a result, $\varphi^\ell_{\alpha,i}$ depends on all electrons, as desired.

The layer is referred to as the Split Subspace Layer due to the fact that its input is one subspace of $\mathbb{R}^D$, given by β^ℓ_α; whereas its output is in the orthogonal complement of this subspace, given by ξ^ℓ_α. Note that there are multiple possible versions of this layer, as any choice D_β in the set $\{1,\dots,D-1\}$ is valid. Since in our case $D=3$, this gives us exactly two choices: $D_\beta = 1$ or $D_\beta = 2$.

The main ingredient of the layer is the network φ^ℓ_α. We now show two things: (1) the layer is invertible for any choice of φ^ℓ_α (2) we derive conditions on φ^ℓ_α to achieve $\mathbb{G}$-equivariance of T^ℓ_α.

Theorem 7. *Let T^ℓ be a Split Subspace Layer, as given in Equation (22). Then T^ℓ is invertible. In particular, let $\underline{\gamma}^{\ell+1}_{\alpha,i} = (\beta^\ell_\alpha)^T r^{\ell+1}_{\alpha,i}$; then the inverse of the layer is given by*

$$r^\ell_{\alpha,i} = r^{\ell+1}_{\alpha,i} - \xi^\ell_\alpha \varphi^\ell_{\alpha,i}(\underline{\gamma}^{\ell+1}_\alpha, \underline{\gamma}^{\ell+1}_{\hat\alpha}) \tag{23}$$

Furthermore, the layer T^ℓ is $\mathbb{G}$-equivariant if

$$\varphi^\ell_\alpha(\pi_\alpha\gamma^\ell_\alpha, \pi_{\hat\alpha}\gamma^\ell_{\hat\alpha}) = \pi_\alpha \varphi^\ell_\alpha(\gamma^\ell_\alpha, \gamma^\ell_{\hat\alpha}) \tag{24}$$

i.e. if $\varphi^\ell_\alpha(\gamma^\ell_\alpha, \gamma^\ell_{\hat\alpha})$ is equivariant with respect to permutations on γ^ℓ_α and invariant with respect to permutations on $\gamma^\ell_{\hat\alpha}$.

Proof: See the Extended Version.

The Split Subspace Layer therefore depends on implementation of the network φ^ℓ_α so that it satisfies Equation (24). We show in the Extended Version how φ^ℓ_α can be implemented with a combination of multihead attention, fully connected layers, and linear projections. We specify a more general version of the Split Subspace Layer in the Extended Version.

We now comment on the complexity of this method vs. that of standard MCMC approaches to sampling. In our case, a single sample may be drawn by passing a sample from the base density through the flow network; in other words, only a single call to the flow network is required. By contrast, when sampling using MCMC techniques, each sample requires many calls to the network representing the wavefunction or density. For example, in using Langevin Monte Carlo techniques, a network representing the (unnormalized) density $\rho(r)$ may be sampled by the iterations

$$r_{k+1} = r_k + \tau \nabla \log \rho(r_k) + \sqrt{2\tau}\xi_k \tag{25}$$

where ξ_k is Gaussian noise. In the limit of $\tau \to 0$, as $k \to \infty$, the samples thus generated will be distributed according to (the normalized version of) the density $\rho(r)$. We note that in the case of finite τ, one often adds treats each new iterate as a Metropolis-Hastings style proposal, which can be correspondingly accepted or rejected. The key point, however, is that a single sample requires many calls to the network representing ρ, whereas in our case only a single network call is required.

4.6 Training via SGD

Log Domain: Density In order to avoid numerical issues, it is best to operate in the log domain. Suppose that

$$\psi(r) = e^{q(r)+iw(r)} \tag{26}$$

so that

$$q(r) = \tfrac{1}{2}\log\rho(r) \;\text{ and }\; w(r) = \text{atan2}\left(\kappa_i(r), \kappa_r(r)\right) \tag{27}$$

where $\kappa_r(r)$ and $\kappa_i(r)$ are the real and imaginary parts of the phase $\kappa(r)$, respectively; and atan2 is the "full" arctangent.

The log-density $q(r;\theta)$ may be computed for both continuous and discrete normalizing flows, where we now introduce the parameters θ of the network explicitly. Consider a sample z chosen from the base density $\rho_z(z)$, and in analogy to $q(r)$, define $q_z(z) = \frac{1}{2}\log\rho_z(z)$. Now, in the case of a continuous normalizing flow, let $v(t)$ satisfy Equation (19); then $q(r;\theta)$ can be by computed (Chen et al. 2018) by solving the ODE

$$\frac{da}{dt} = -\text{Trace}\left(\frac{\partial \Gamma_t}{\partial v}(v(t);\theta)\right) \tag{28}$$

with $a(0) = q_z(z)$ and $q(r;\theta) = a(1)$. This is the continuous analogue of the change of variables formula. In the case of a discrete normalizing flow, fix the following notation: $r^0 = z$, $r = r^{L+1}$, and $T = T^L \circ \cdots \circ T^0$. Then we may use a logarithmic version of the standard change of variables formula (12):

$$q(r;\theta) = q_z\left(T^{-1}(r;\theta)\right) + \frac{1}{2}\sum_{\ell=0}^{L} \log\left|\det J_{(T^\ell)^{-1}}(r^{\ell+1};\theta)\right| \tag{29}$$

Log Domain: Gradient of the Objective Recall that our goal in finding an approximation to the ground state wavefunction is to solve the optimization problem in Equation (6). Using Equation (8) and noting that $\langle\psi(\cdot;\theta)|\psi(\cdot;\theta)\rangle = 1$ since $\rho(\cdot;\theta)$ is normalized, we may write the objective function to be minimized as

$$\begin{aligned}\mathcal{L}(\theta) &= \langle\psi(\cdot;\theta)|H|\psi(\cdot;\theta)\rangle \\ &= \mathbb{E}_{r\sim\rho(\cdot;\theta)}\left[\mathcal{E}_r(r;\theta)\right] \approx \frac{1}{K}\sum_{k=1}^{K}\mathcal{E}_r\left(r^{(k)};\theta\right)\end{aligned} \tag{30}$$

with samples $r^{(k)} \sim \rho(\cdot;\theta)$. Then we have the following theorem, which shows that the local energy can be written entirely as a function of $q(r;\theta)$ and the potential $V(r)$, so that the phase $w(r;\theta)$ does not appear; and furthermore gives the gradient of the objective function $\mathcal{L}(\theta)$.

Theorem 8. *The local energy can be written as*

$$\mathcal{E}_r(r;\theta) = -\tfrac{1}{2}\Delta_r q(r;\theta) - \tfrac{1}{2}\|\nabla_r q(r;\theta)\|^2 + V(r) \tag{31}$$

In particular, the local energy is independent of the phase $w(r;\theta)$. Furthermore, let

$$\Omega(r;\theta) = 2\left(\mathcal{E}_r(r;\theta) - \mathbb{E}_{r\sim\rho(\cdot;\theta)}\left[\mathcal{E}_r(r;\theta)\right]\right)\nabla_\theta q(r;\theta) \tag{32}$$

Then the gradient of the loss function may be written as

$$\nabla_\theta\mathcal{L}(\theta) = \mathbb{E}_{r\sim\rho(\cdot;\theta)}\left[\Omega(r;\theta)\right] \approx \frac{1}{K}\sum_{k=1}^{K}\Omega\left(r^{(k)};\theta\right) \tag{33}$$

with samples $r^{(k)} \sim \rho(\cdot;\theta)$.

Proof: See the Extended Version.

Thus, in order to optimize the objective in Equation (30), we may use gradient descent using the estimate for the gradient in Equation (33). A detailed version of the optimization routine is given in the Extended Version.

5 Further Details: Phase, Cusps, and Induction

5.1 The Phase

Since the Hamiltonian is time-reversal invariant and Hermitian, both its eigenvalues and its eigenfunctions are real. Since the ground-state wavefunction we are looking for is real, the phase can be taken to belong to the two element set $\{0, \pi\}$. Given that we now know how to solve for an approximation to the density $\rho_0(r)$ corresponding to the ground state wavefunction, we now show one way of assigning the phase so that the resulting ground state wavefunction $\psi_0(r)$ is appropriately antisymmetric.

Theorem 9. *Let $\rho_0(r)$ be the the density for the ground state wavefunction. Let $\prec$ be a strict total order on $\mathbb{R}^D$, and define the set*

$$\mathcal{R} = \{r = (r_1, \dots r_n) : r_1 \prec r_2 \prec \cdots \prec r_{n_u} \text{ and } r_{n_u+1} \prec r_{n_u+2} \prec \cdots \prec r_n\} \quad (34)$$

For any r without $r_i = r_j$, define the permutation $\pi_\prec(r) \in \mathbb{G}$ by $\pi_\prec(r)r \in \mathcal{R}$. Then a valid antisymmetric ground state wavefunction is given by

$$\psi_0(r) = \begin{cases} (-1)^{\pi_\prec(r)}\sqrt{\rho_0(r)} & \text{if } r_i \neq r_j \ \forall i, j \\ 0 & \text{otherwise} \end{cases} \quad (35)$$

Proof: See the Extended Version.

Thus, given the density ρ_0, we can use Theorem 9 to easily compute the ground state wavefunction ψ_0. A question remains: what is the strict total order $\prec$? Any choice is valid, but the simplest thing to do is to use lexicographic ordering on the coordinates of the two points in $\mathbb{R}^D$ that are being compared.

5.2 Incorporating Cusps

Electron-Electron Cusps Wavefunctions are known to have certain non-smooth properties, known as cusps. In particular, the gradient of the wavefunction should exhibit a discontinuity when two electrons coincide. One way to incorporate such gradient discontinuities is via the introduction of terms which depend on the distance between electrons (Pfau et al. 2020); as the distance is itself a continuous but non-smooth function of the electron positions, using distances can allow us to model such cusps. In the case of the discrete normalizing flow, our goal will be to design a layer which incorporates the inter-electron distances directly. Given the requirements of a normalizing flow, the challenge is to enforce invertibility for such a layer. We have the following result:

Theorem 10. *Let the set of distances be given by $\delta^\ell = \left\{\delta^\ell_{ij}\right\}_{i<j}$ where $\delta^\ell_{ij} = \|r^\ell_i - r^\ell_j\|$. Given a layer of the form*

$$r^{\ell+1}_i = \Theta^\ell(\delta^\ell; \theta)\, r^\ell_i + t^\ell(\delta^\ell; \theta) \quad (36)$$

with $\Theta^\ell(\delta^\ell; \theta) \in O(D)$ and $t^\ell(\delta^\ell; \theta) \in \mathbb{R}^D$. Then the layer is both $\mathbb{G}$-equivariant as well as invertible.

Proof: See the Extended Version.

The essence of this layer to rotate all electrons in a given configuration $r = (r_1, \dots, r_n)$ by the same rotation matrix Θ^ℓ and translation vector t^ℓ; and the rotation matrix and translation vector are both functions the configuration r entirely through the distances δ^ℓ. The latter fact is crucial, as it means that different configurations r are treated differently, which gives the layer expressivity. An implementation of this layer based on a Deep Set architecture (Zaheer et al. 2017) is given in the Extended Version.

It is also known that the gradient of the wavefunction should exhibit a discontinuity when an electron and nucleus coincide. The treatment is similar, and is given in the Extended Version.

5.3 Induction Across Multiple Molecules

In an effort to accelerate the ground state computation, we may try to learn the ground state wavefunctions and energies for an entire class of molecules simultaneously, as in (Gao and Günnemann 2023; Scherbela, Gerard, and Grohs 2024; Gerard et al. 2024). In particular, the molecular parameters are given by $R = (R_1, \dots, R_N)$, the nuclear positions; and $Z = (Z_1, \dots, Z_N)$, the atomic numbers of each nucleus. Then our goal is to learn a function of the form $\psi_0(x; R, Z)$, i.e. a ground state wavefunction which is explicitly parameterized by the molecular parameters. This entails computing the density $\rho(r; R, Z)$. However, this latter task is made more complicated by the fact that two new invariances are required. The first is nuclear permutation invariance:

$$\rho(r; \pi R, \pi Z) = \rho(r; R, Z) \quad \text{for } \pi \in \mathbb{S}_N \quad (37)$$

and the second is joint rigid motion invariance:

$$\rho(\tau r; \tau R, Z) = \rho(r; R, Z) \quad \text{for } \tau \in E(D) \quad (38)$$

We henceforth assume that the nuclei have their center of mass at the origin, i.e. $\bar{R} = \frac{1}{N}\sum_{I=1}^N R_I = 0$; this removes the need to deal with translations, which generally require special (and uninteresting) treatment, e.g. see (Satorras, Hoogeboom, and Welling 2021). Thus, Equation (38) becomes joint rotation invariance:

$$\rho(\Theta r; \Theta R, Z) = \rho(r; R, Z) \quad \text{for } \Theta \in O(D) \quad (39)$$

We now show that densities satisfying Equations (37) and (39) can be realized via a variation of the continuous normalizing flow we have introduced in Section 4.4:

Theorem 11. *Let $\bar{R} = \frac{1}{N}\sum_{I=1}^N R_I = 0$. Given a continuous normalizing flow of the form $dv/dt = \Gamma_t(v; R, Z)$ with $v(0) = z \sim \rho_z(\cdot)$ and $r = v(1)$. Let the function Γ_t be invariant with respect to nuclear permutations and equivariant with respect to joint rotations, i.e. for all t*

$$\begin{aligned} &\Gamma_t(v; \pi R, \pi Z) = \Gamma_t(v; R, Z) \ \ \forall \pi \in \mathbb{S}_N \\ &\Gamma_t(\Theta v; \Theta R, Z) = \Theta \Gamma_t(v; R, Z) \ \ \forall \Theta \in O(D) \end{aligned} \quad (40)$$

Furthermore, suppose that the base density is invariant with respect to rotations, $\rho_z(\Theta z) = \rho_z(z)$ for $\Theta \in O(D)$. Then the resulting density $\rho(r; R, Z)$ satisfies Equations (37) and (39).

Proof: See the Extended Version.

First, we note that the base density in Equation (15) can be made invariant to rotations by constructing the relevant Projection DPP from a kernel function $K(y, y') = H(y)^T H(y)$, where the functions $h_i(y)$ are derived from taking arbitrary rotationally-invariant functions $\tilde{h}_i(y)$, and orthogonalizing them with Gram-Schmidt; e.g. one may use Gaussians of varying bandwidths, $\tilde{h}_i(y) = e^{-\|y\|^2/\sigma_i^2}$.

Now, we turn to the construction of Γ_t. Recall from Theorem 6 that $\Gamma_t(\cdot; R, Z)$ must be $\mathbb{G}$-equivariant for all t. Furthermore, we have already noted that $\mathbb{G}$-equivariant functions may be constructed using a combination of standard pieces: multihead attention, fully connected layers, and linear projections. It would be nice if we were able to use this result while also incorporating the extra conditions in Equation (38). We now show that this is possible:

Theorem 12. *Let $\phi_t(v; R, Z)$ be a function which is $\mathbb{G}$-equivariant with respect to v i.e. $\phi_t(gv; R, Z) = g\phi_t(v; R, Z)$ for $g \in \mathbb{G}$. Let $\omega_t(v; R, z)$ be a function whose output is itself a rotation, i.e. $\omega_t(v; R, z) \in O(D)$. Let ω_t be $\mathbb{G}$-invariant with respect to v, and $O(D)$-equivariant jointly with respect to v and R i.e. $\omega_t(\Theta v; \Theta R, Z) = \Theta\omega_t(v; R, Z)$. Finally, let both ϕ_t and ω_t be permutation-invariant jointly with respect to R and Z i.e. $\phi_t(v; \pi R, \pi Z) = \phi_t(v; R, Z)$ and likewise for ω_t. Then the function*

$$\Gamma_t(v; R, Z) = \zeta\phi_t(\zeta^{-1}v; \zeta^{-1}R, Z) \tag{41}$$

where $\zeta = \omega_t(v; R, Z)$ satisfies the properties in Equation (40) and is $\mathbb{G}$-equivariant with respect to v.

Proof: See the Extended Version.

We can use the previously mentioned recipe in the Extended Version in order to construct a $\mathbb{G}$-equivariant ϕ_t, with an extra path in the network for the R, Z dependence, based on either Deep Set or a Transformer architecture with pooling to gain the requisite invariance. The function ω_t can be constructed by using an $E(D)$ Equivariant Graph Neural Network (Satorras, Hoogeboom, and Welling 2021) whose output is a rotation matrix, similar to what is done in (Kaba et al. 2023). More detailed information is contained in the Extended Version.

6 Conclusions

We have demonstrated a theoretical framework for efficiently solving the Electronic Schrödinger Equation using normalizing flows. Using these flows allows us to sample efficiently from the wavefunction, thereby side-stepping the need for time-consuming MCMC approaches to sampling. Future work will focus on using diffusion techniques (Yang et al. 2023) to model wavefunctions, which is not straightforward due to the need to maintain the requisite quantum mechanical properties; and on adapting flow-matching (Lipman et al. 2022) and related techniques for the optimization of the variational objective.

References

Abbott, R.; Albergo, M. S.; Botev, A.; Boyda, D.; Cranmer, K.; Hackett, D. C.; Kanwar, G.; Matthews, A. G.; Racanière, S.; Razavi, A.; et al. 2023a. Normalizing flows for lattice gauge theory in arbitrary space-time dimension. *arXiv preprint arXiv:2305.02402*.

Abbott, R.; Albergo, M. S.; Botev, A.; Boyda, D.; Cranmer, K.; Hackett, D. C.; Matthews, A. G.; Racanière, S.; Razavi, A.; Rezende, D. J.; et al. 2023b. Aspects of scaling and scalability for flow-based sampling of lattice QCD. *The European Physical Journal A*, 59(11): 257.

Austin, B. M.; Zubarev, D. Y.; and Lester Jr, W. A. 2012. Quantum Monte Carlo and related approaches. *Chemical reviews*, 112(1): 263–288.

Boyda, D.; Kanwar, G.; Racanière, S.; Rezende, D. J.; Albergo, M. S.; Cranmer, K.; Hackett, D. C.; and Shanahan, P. E. 2021. Sampling using SU (N) gauge equivariant flows. *Physical Review D*, 103(7): 074504.

Carleo, G.; Cirac, I.; Cranmer, K.; Daudet, L.; Schuld, M.; Tishby, N.; Vogt-Maranto, L.; and Zdeborová, L. 2019. Machine learning and the physical sciences. *Reviews of Modern Physics*, 91(4): 045002.

Carleo, G.; and Troyer, M. 2017. Solving the quantum many-body problem with artificial neural networks. *Science*, 355(6325): 602–606.

Cassella, G.; Sutterud, H.; Azadi, S.; Drummond, N.; Pfau, D.; Spencer, J. S.; and Foulkes, W. M. C. 2023. Discovering quantum phase transitions with fermionic neural networks. *Physical Review Letters*, 130(3): 036401.

Ceperley, D.; and Alder, B. 1986. Quantum monte carlo. *Science*, 231(4738): 555–560.

Chen, R. T.; Rubanova, Y.; Bettencourt, J.; and Duvenaud, D. K. 2018. Neural ordinary differential equations. *Advances in neural information processing systems*, 31.

Deng, D.-L.; Li, X.; and Das Sarma, S. 2017a. Machine learning topological states. *Physical Review B*, 96(19): 195145.

Deng, D.-L.; Li, X.; and Das Sarma, S. 2017b. Quantum entanglement in neural network states. *Physical Review X*, 7(2): 021021.

Entwistle, M. T.; Schätzle, Z.; Erdman, P. A.; Hermann, J.; and Noé, F. 2023. Electronic excited states in deep variational Monte Carlo. *Nature Communications*, 14(1): 274.

Foulkes, W. M.; Mitas, L.; Needs, R.; and Rajagopal, G. 2001. Quantum Monte Carlo simulations of solids. *Reviews of Modern Physics*, 73(1): 33.

Gao, N.; and Günnemann, S. 2023. Generalizing neural wave functions. In *International Conference on Machine Learning*, 10708–10726. PMLR.

Gao, X.; and Duan, L.-M. 2017. Efficient representation of quantum many-body states with deep neural networks. *Nature communications*, 8(1): 662.

Gautier, G.; Bardenet, R.; and Valko, M. 2019. On two ways to use determinantal point processes for Monte Carlo integration. *Advances in Neural Information Processing Systems*, 32.

Gerard, L.; Scherbela, M.; Marquetand, P.; and Grohs, P. 2022. Gold-standard solutions to the Schrödinger equation using deep learning: How much physics do we need? *Advances in Neural Information Processing Systems*, 35: 10282–10294.

Gerard, L.; Scherbela, M.; Sutterud, H.; Foulkes, M.; and Grohs, P. 2024. Transferable Neural Wavefunctions for Solids. *arXiv preprint arXiv:2405.07599*.

Gubernatis, J.; Kawashima, N.; and Werner, P. 2016. *Quantum Monte Carlo Methods*. Cambridge University Press.

Han, J.; Zhang, L.; and Weinan, E. 2019. Solving many-electron Schrödinger equation using deep neural networks. *Journal of Computational Physics*, 399: 108929.

Hermann, J.; Schätzle, Z.; and Noé, F. 2020. Deep-neural-network solution of the electronic Schrödinger equation. *Nature Chemistry*, 12(10): 891–897.

Johansson, K. 2006. Random matrices and determinantal processes. In *Les Houches*, volume 83, 1–56. Elsevier.

Kaba, S.-O.; Mondal, A. K.; Zhang, Y.; Bengio, Y.; and Ravanbakhsh, S. 2023. Equivariance with learned canonicalization functions. In *International Conference on Machine Learning*, 15546–15566. PMLR.

Kanwar, G.; Albergo, M. S.; Boyda, D.; Cranmer, K.; Hackett, D. C.; Racaniere, S.; Rezende, D. J.; and Shanahan, P. E. 2020. Equivariant flow-based sampling for lattice gauge theory. *Physical Review Letters*, 125(12): 121601.

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

Lavancier, F.; Møller, J.; and Rubak, E. 2015. Determinantal point process models and statistical inference. *Journal of the Royal Statistical Society Series B: Statistical Methodology*, 77(4): 853–877.

Levine, Y.; Sharir, O.; Cohen, N.; and Shashua, A. 2019. Quantum entanglement in deep learning architectures. *Physical review letters*, 122(6): 065301.

Li, X.; Li, Z.; and Chen, J. 2022. Ab initio calculation of real solids via neural network ansatz. *Nature Communications*, 13(1): 7895.

Li, Z.; Kovachki, N.; Azizzadenesheli, K.; Liu, B.; Bhattacharya, K.; Stuart, A.; and Anandkumar, A. 2020. Fourier neural operator for parametric partial differential equations. *arXiv preprint arXiv:2010.08895*.

Lipman, Y.; Chen, R. T.; Ben-Hamu, H.; Nickel, M.; and Le, M. 2022. Flow Matching for Generative Modeling. In *The Eleventh International Conference on Learning Representations*.

Lu, L.; Jin, P.; Pang, G.; Zhang, Z.; and Karniadakis, G. E. 2021. Learning nonlinear operators via DeepONet based on the universal approximation theorem of operators. *Nature machine intelligence*, 3(3): 218–229.

Naito, T.; Naito, H.; Hashimoto, K.; et al. 2023. Multi-body wave function of ground and low-lying excited states using unornamented deep neural networks. *Physical Review Research*, 5(3): 033189.

Needs, R. J.; Towler, M. D.; Drummond, N. D.; and Ríos, P. L. 2009. Continuum variational and diffusion quantum Monte Carlo calculations. *Journal of Physics: Condensed Matter*, 22(2): 023201.

Passetti, G.; Hofmann, D.; Neitemeier, P.; Grunwald, L.; Sentef, M. A.; and Kennes, D. M. 2023. Can neural quantum states learn volume-law ground states? *Physical Review Letters*, 131(3): 036502.

Pescia, G.; Han, J.; Lovato, A.; Lu, J.; and Carleo, G. 2022. Neural-network quantum states for periodic systems in continuous space. *Physical Review Research*, 4(2): 023138.

Pfau, D.; Axelrod, S.; Sutterud, H.; von Glehn, I.; and Spencer, J. S. 2023. Natural Quantum Monte Carlo Computation of Excited States. *arXiv preprint arXiv:2308.16848*.

Pfau, D.; Spencer, J. S.; Matthews, A. G.; and Foulkes, W. M. C. 2020. Ab initio solution of the many-electron Schrödinger equation with deep neural networks. *Physical Review Research*, 2(3): 033429.

Raissi, M.; Perdikaris, P.; and Karniadakis, G. E. 2019. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational physics*, 378: 686–707.

Ren, W.; Fu, W.; Wu, X.; and Chen, J. 2023. Towards the ground state of molecules via diffusion Monte Carlo on neural networks. *Nature Communications*, 14(1): 1860.

Rozenberg, E.; Karnieli, A.; Yesharim, O.; Foley-Comer, J.; Trajtenberg-Mills, S.; Freedman, D.; Bronstein, A. M.; and Arie, A. 2022. Inverse design of spontaneous parametric downconversion for generation of high-dimensional qudits. *Optica*, 9(6): 602–615.

Rozenberg, E.; Karnieli, A.; Yesharim, O.; Trajtenberg-Mills, S.; Freedman, D.; Bronstein, A. M.; and Arie, A. 2021. Inverse design of quantum holograms in three-dimensional nonlinear photonic crystals. In *CLEO: QELS_Fundamental Science*, FM1N–7. Optica Publishing Group.

Ryczko, K.; Strubbe, D. A.; and Tamblyn, I. 2019. Deep learning and density-functional theory. *Physical Review A*, 100(2): 022512.

Saleh, Y.; Corral, Á. F.; Iske, A.; Küpper, J.; and Yachmenev, A. 2023. Computing excited states of molecules using normalizing flows. *arXiv preprint arXiv:2308.16468*.

Satorras, V. G.; Hoogeboom, E.; and Welling, M. 2021. E(n) equivariant graph neural networks. In *International conference on machine learning*, 9323–9332. PMLR.

Schätzle, Z.; Hermann, J.; and Noé, F. 2021. Convergence to the fixed-node limit in deep variational Monte Carlo. *The Journal of Chemical Physics*, 154(12).

Scherbela, M.; Gerard, L.; and Grohs, P. 2024. Towards a transferable fermionic neural wavefunction for molecules. *Nature Communications*, 15(1): 120.

Sharir, O.; Shashua, A.; and Carleo, G. 2022. Neural tensor contractions and the expressive power of deep neural quantum states. *Physical Review B*, 106(20): 205136.

Spencer, J. S.; Pfau, D.; Botev, A.; and Foulkes, W. M. C. 2020. Better, faster fermionic neural networks. *arXiv preprint arXiv:2011.07125*.

Stokes, J.; Chen, B.; and Veerapaneni, S. 2023. Numerical and geometrical aspects of flow-based variational quantum

Monte Carlo. *Machine Learning: Science and Technology*, 4(2): 021001.

Thiede, L.; Sun, C.; and Aspuru-Guzik, A. 2022. Waveflow: Enforcing boundary conditions in smooth normalizing flows with application to fermionic wave functions. *arXiv preprint arXiv:2211.14839*.

Torlai, G.; Mazzola, G.; Carrasquilla, J.; Troyer, M.; Melko, R.; and Carleo, G. 2018. Neural-network quantum state tomography. *Nature physics*, 14(5): 447–450.

Umrigar, C.; Nightingale, M.; and Runge, K. 1993. A diffusion Monte Carlo algorithm with very small time-step errors. *The Journal of chemical physics*, 99(4): 2865–2890.

Wilson, M.; Gao, N.; Wudarski, F.; Rieffel, E.; and Tubman, N. M. 2021. Simulations of state-of-the-art fermionic neural network wave functions with diffusion Monte Carlo. *arXiv preprint arXiv:2103.12570*.

Wilson, M.; Moroni, S.; Holzmann, M.; Gao, N.; Wudarski, F.; Vegge, T.; and Bhowmik, A. 2022. Wave function ansatz (but periodic) networks and the homogeneous electron gas. *arXiv preprint arXiv:2202.04622*.

Xie, H.; Zhang, L.; and Wang, L. 2022. Ab-Initio Study of Interacting Fermions at Finite Temperature with Neural Canonical Transformation. *Journal of Machine Learning*, 1(1): 38–59.

Xie, H.; Zhang, L.; and Wang, L. 2023. m^* of two-dimensional electron gas: A neural canonical transformation study. *SciPost Physics*, 14(6): 154.

Yang, L.; Zhang, Z.; Song, Y.; Hong, S.; Xu, R.; Zhao, Y.; Zhang, W.; Cui, B.; and Yang, M.-H. 2023. Diffusion models: A comprehensive survey of methods and applications. *ACM Computing Surveys*, 56(4): 1–39.

Zaheer, M.; Kottur, S.; Ravanbakhsh, S.; Poczos, B.; Salakhutdinov, R. R.; and Smola, A. J. 2017. Deep sets. *Advances in neural information processing systems*, 30.

Zhang, X.; Wang, L.; Helwig, J.; Luo, Y.; Fu, C.; Xie, Y.; Liu, M.; Lin, Y.; Xu, Z.; Yan, K.; et al. 2023. Artificial intelligence for science in quantum, atomistic, and continuum systems. *arXiv preprint arXiv:2307.08423*.