Gradient Descent Optimization of Fermion Nodes in Diffusion Monte Carlo

John McFarland(1,4) and Efstratios Manousakis(1,2,6)
(1) Department of Physics and National High Magnetic Field Laboratory, Florida State University, Tallahassee, FL 32306-4350, USA and (2) Department of Physics, National and Kapodistrian University of Athens, Panepistimiopolis, Zografos, 157 84 Athens, Greece

We present a method for optimizing the location of the fermion ground-state nodes using a combination of diffusion Monte Carlo (DMC) and projected gradient descent (PGD). A PGD iteration shifts the parameters of an arbitrary node-fixing trial function in the opposite direction of the DMC energy gradient, while maintaining the cusp condition for atomic electrons. The energy gradient is calculated from DMC walker distributions by one of three methods we derive from an exact analytical expression. We combine our energy gradient calculation methods with different gradient descent algorithms and a projection operator that maintains the cusp condition. We apply this stochastic PGD method to trial functions of Be, Li$_2$, and Ne, all consisting of a single Slater determinant with randomized parameters, and find that the nodes dramatically improve to the same DMC energy as nodes optimized by variational Monte Carlo. Our method, therefore, departs from the standard procedure of optimizing the nodes with a non-DMC scheme such as variational Monte Carlo, Density function theory, or configuration interaction based calculation, which do not directly minimize the DMC energy.

I. INTRODUCTION

Diffusion Monte Carlo (DMC)\cite{11,12}, also known as Green’s function Monte Carlo or Projector Monte Carlo, is a technique for projecting the many-body wavefunction to the ground state. It has been used to accurately tackle bosonic or non-frustrated quantum-spin systems (see Ref.\cite{13} for list of References). In the case of fermionic systems, while in general one has to restrict oneself to the fixed-node approximation, where it has been accurately applied to nuclear physics\cite{14,15}, when the required initial trial function was accurate enough. Node relaxation has led to accurate results for the electron gas in the continuum\cite{16} and for electrons on a lattice\cite{17,18}. There are variants of the method, such as the constraint path\cite{19,20} which have been successfully applied to condensed matter physics\cite{21,22} problems.

The DMC method samples the ground-state wavefunction $\Psi$ of $N$ particles with walkers. These are points in a $dN$-dimensional space ($d$ being the dimensionality of space) with a probability density equal to $\Psi\Psi_g$, where $\Psi_g$ is a guiding function used for importance sampling. DMC projects $\Psi$ to the ground state by propagating it along the imaginary time axis, which is done by changing the position and weight of each walker in a way that samples the Green’s function (i.e. the matrix elements of the evolution operator). To prevent large variations in weights, walkers are regularly deleted, duplicated, or combined in a way that the initial and final $\Psi\Psi_g$ are proportional.

For a fermionic $\Psi$, it is generally necessary to impose zero boundary conditions at an a priori chosen nodal surface. That is, the nodal surface of $\Psi$ is taken to be the nodal surface of a trial function $\Psi_t$, which is an approximate ground-state wavefunction produced by a non-DMC method. Typically $\Psi_t$ is a product of a symmetric Jastrow function\cite{23} factor, which describes correlations, and an anti-symmetric factor, which is a single or a combination of Slater determinants\cite{24} that describe the nodes. It is standard to set $\Psi_t = \Psi_g$, in which case nodal boundary conditions are naturally imposed by imaginary-time propagation alone. In the present paper we distinguish $\Psi_t$ from $\Psi_g$ because we often use a node-less $\Psi_g$, in which case nodal boundary conditions are imposed by deleting those walkers that attempt to cross the nodes.

DMC error is improved with a $\Psi_t$ that better approximates the exact ground state wavefunction. The only source of error that cannot be eliminated with the DMC propagation is the fixed node error, which is contained in just the anti-symmetric part of $\Psi_t$. The other sources of error, namely the finite time-step error, statistical error, and population control error, can be respectively controlled with a smaller time-step, a greater sample number, and a larger walker population.

This makes the choice of the nodes of $\Psi_t$ the fundamental approximation of DMC. The resulting fixed-node error typically ranges from about 82 to 435 meV per atom\cite{25} and is generally controlled by increasing the complexity of $\Psi_t$ and better optimizing its parameters. Because DMC energy is an upper bound to the true ground state energy\cite{26}, a parameter optimization method that minimizes DMC energy is the most accurate. However, optimization methods in use will minimize some other quantity, such as the VMC energy, the Kohn-Sham energy, or the local VMC energy variance\cite{27,28}.

The first attempt to optimize the parameters of $\Psi_t$ using DMC walker distributions alone was by Reboredo et al.\cite{29}. They iteratively generated a new $\Psi_t$ by projecting the coefficients of its determinants (or pfaffians) from the walker distribution of the previous $\Psi_t$. Mindful that it becomes expensive to evaluate a $\Psi_t$ with an increasing number of determinants, they also proposed a method to
reduce the complexity of $\Psi_t$ by minimizing a cost function between the initial and reduced $\Psi_t$ that heavily penalized changes of the nodes.

We propose to directly use the DMC energy as a “cost function” when optimizing the parameters of $\Psi_t$. This differs from using the VMC energy as a cost function. Our method iteratively performs PGD on $\Psi_t$ by shifting its parameters in a direction roughly opposite to the energy gradient, while maintaining any constraints (the electron-nuclear cusp condition in our examples) by projecting the parameters back to the surface satisfying the constraint with each iteration. Gradient descent has an advantage for large parameter number $N_p$ in that it does not require an estimate of $N_p \times N_p$ matrices, which is required by many others, e.g. the linear method.

We calculate the gradient of each parameter iteration from walker samples using one of three methods that we label A, B, and C. The walker distributions are produced by DMC imaginary time propagation with nodes fixed by the parameters. We experiment with several gradient descent algorithms commonly used for machine learning. In order to keep our method self-reliant, during PGD we do not rely on a pre-optimized Jastrow function, which does not affect the fixed-node error, although it does affect the other errors.

Our paper is organized as follows: In Sec. II we present the three methods for evaluating the derivative of the energy from the DMC walker distribution. In Sec. III we describe our implementation of gradient descent, including the DMC scheme used, practical issues, and the form of $\Psi_t$ and $\Psi_g$. In Sec. IV we present tests on the accuracy of our method, including the DMC scheme used, practical issues, and the form of $\Psi_t$ and $\Psi_g$. In Sec. V we discuss the advantages of our method and ways to improve it. We discuss parameter fluctuations in Appendix A and we describe the gradient descent algorithms used in Appendix B and we compare them in Appendix C.

II. METHOD

A PGD iteration shifts the nodes in a direction expected to lower the DMC energy $E$. The nodes are determined by the parameters $\theta_i$ of the anti-symmetric part of $\Psi_t$, and the shift of the nodes is determined by the gradient $\frac{\partial E}{\partial \theta_i}$. The gradient of a nodal surface is calculated with walker samples produced by DMC imaginary-time propagation of $\Psi \Psi_g$, with $\Psi$ the fixed-node ground state and $\Psi_g$ a guiding function. Thus PGD iterations change the nodal surface, while DMC iterations generate data for the gradient of a given nodal surface.

Provided the $\theta_i$’s are not already at a local minimum, $E$ will be lowered by the following change of parameters

$$\theta_i \rightarrow \theta_i - a_i \frac{\partial E}{\partial \theta_i},$$

provided that $a_i$ is positive and sufficiently small. However, the presence of stochastic error in our gradient means that only the expectation of $E$ will be lowered with sufficiently small $a_i$. If the $\theta_i$ are constrained (e.g. from the cusp condition or symmetry) we modify Eq. [1] to

$$\theta_i \rightarrow P(\theta_i - a_i \frac{\partial E}{\partial \theta_i}),$$

where $P$ is the projection operator that moves the parameters to the nearest point on the manifold that satisfies the constraint. Some gradient descent algorithms, e.g. ADAM, simulate momentum with friction by replacing $\frac{\partial E}{\partial \theta_i}$ with an exponential trailing average of current and past iterations.

II.A. Derivative of the energy

Central to our method is calculating the DMC energy gradient in parameter space. Working in atomic units, we derive an expression for $\frac{\partial E}{\partial \theta_i}$, also derived by Berman. To simplify the derivation, let us again set the ground state $\Psi$ to zero except for one nodal pocket, as justified in Sec. II.A. Now let us examine how the energy, given as

$$E = \frac{\langle \Psi | H | \Psi \rangle}{\langle \Psi | \Psi \rangle},$$

changes due to an infinitesimal change $\delta \theta_i$ in one of the parameters, which shifts the node and, thus, changes $\Psi$ to $\Psi + \delta \Psi$. By doing that the change in energy $\delta E$ is given by

$$\delta E = \frac{\langle \delta \Psi | (H - E) | \Psi \rangle}{\langle \Psi | \Psi \rangle} + \frac{\langle \Psi | (H - E) | \delta \Psi \rangle}{\langle \Psi | \Psi \rangle} + \frac{\langle \delta \Psi | (H - E) | \delta \Psi \rangle}{\langle \Psi | \Psi \rangle}.$$  

We convert the above expression to an integral over the 3N dimensional position vector $R$ and integrate by parts to obtain

$$\delta E = \int dR \frac{2 \delta \Psi(H - E) \Psi + \delta \Psi(H - E) \delta \Psi}{\langle \Psi | \Psi \rangle}.$$  

The $\delta \Psi(H - E) \Psi$ term of Eq. [5] is zero except at the original nodes, where the action of the Laplacian on a discontinuity of $\nabla \Psi$ produces a delta function. The term $\delta \Psi(H - E) \delta \Psi$ of the above equation is second order in $\delta \theta_i$ except at the original and shifted nodes, where the action of the Laplacian on the discontinuities of $\nabla \delta \Psi$ produce zeroth order delta functions. Thus, we only need to consider the operation of the Laplacian on these discontinuities at the original and shifted nodes to evaluate Eq. [5] up to first order in $\delta \theta_i$.

Let us now choose a coordinate system with the shape of the nodes. Let $z$ be the coordinate perpendicular to the node, zero at the initial node, and pointing towards the direction where $\Psi \neq 0$ (see Fig. 1 for an illustration). Let $A$ be the remaining $3N-1$ dimensional coordinates that parameterize the nodal surface at $z = 0$. Then $\Psi$ to first order is

$$\Psi(A, z) = g(A) R(z),$$

Fig. 1 Nodal-pocket.
where \( g(A) \equiv |\nabla \Psi(A, z = 0^+)| \), and \( R \) is the ramp function:

\[
R(z) = \begin{cases} 
  z & \text{if } 0 \leq z \\
  0 & \text{if } z < 0 
\end{cases}.
\]

To express \( \delta \Psi \) near the node, let us define \( \eta(A) \) as the displacement of the node in the \( z \) direction that results from \( \delta \theta_i \). Then, to first order in \( \Psi \), \( \delta \Psi \) is the difference between the displaced and original wave function, that is

\[
\delta \Psi(A, z) = g(A)R(z - \eta(A)) - g(A)R(z).
\]

The delta functions resulting from the Laplacian are then

\[
\nabla^2 \Psi(A, z) = g(A)\delta(z),
\]

\[
\nabla^2 \delta \Psi(A, z) = g(A)\delta(z - \eta) - g(A)\delta(z).
\]

Since only the delta functions contribute to first order, we leave only their contribution to Eq. \( 5 \) yielding

\[
\delta E = \frac{-\int dR \ \delta \Psi(A, z)g(A)\delta(z) + \delta(z - \eta)}{2 \langle \Psi | \Psi \rangle}. \quad (10)
\]

We reduce this to an integral over the nodal surface by integrating over \( z \). Absorbing the Jacobian determinant into \( dA \) we obtain:

\[
\delta E = \frac{-\int_{\text{node}} dA \ g(A)\delta(\Psi(A, 0)) + \delta(\Psi(A, \eta))}{2 \langle \Psi | \Psi \rangle}. \quad (11)
\]

From Eq. \( 7 \) we find that \( \delta \Psi(A, 0) + \delta \Psi(A, \eta) = g(A)\eta(A) \) both when \( \eta(A) > 0 \) and when \( \eta(A) < 0 \). This yields

\[
\delta E = \frac{\int_{\text{node}} dA \ \eta(A)g^2(A)}{2 \langle \Psi | \Psi \rangle}. \quad (12)
\]

By taking a derivative with respect to \( \theta_i \) we derive our analytical expression

\[
\frac{\partial \delta E}{\partial \theta_i} = \int_{\text{node}} dA \left| \nabla \Psi(A) \right|^2 \frac{\partial \theta_i \eta(A)}{2 \langle \Psi | \Psi \rangle}. \quad (13)
\]

**II.B. Practical methods to estimate the energy gradient**

Starting from Eq. \( 13 \) we have derived three different equations for calculating the energy gradient from walker distributions, which we refer to as method A, B, and C. Since DMC usually samples terms linear in \( \Psi \), but Eq. \( 13 \) contains two terms bi-linear in \( \Psi \), an approximation used by methods A and B replaces the bi-linear terms with a mixed estimate of the true ground-state wavefunction and the trial function, i.e.,

\[
\langle \Psi | \Psi \rangle \rightarrow \langle \Psi_t | \Psi \rangle, \quad |\nabla \Psi|^2 \rightarrow |\nabla \Psi_t| \cdot |\nabla \Psi|.
\]

Then, using

\[
\frac{\partial \eta}{\partial \theta_i} = -\frac{1}{|\nabla \Psi_t|} \frac{\partial \Psi_t}{\partial \theta_i},
\]

we approximate Eq. \( 13 \) with

\[
\frac{\partial \delta E}{\partial \theta_i} \approx \frac{-\int_{\text{node}} dA \left| \nabla \Psi(A) \right| \partial \theta_i \Psi_t(A)}{2 \langle \Psi_t | \Psi \rangle}. \quad (16)
\]

Unfortunately, this relies on the quality of \( \Psi_t \), although this quality is improved with PGD. Method C does not make this approximation, thus in principle requires no Jastrow optimization.

**II.C. Method A: the energy gradient from a standard walker distribution**

Method A calculates Eq. \( 16 \) with a standard walker distribution equal to \( \Psi_t \Psi \), which we think will make it easiest to implement into existing DMC code. It requires the evaluation of \( \Psi_t \), its parameter derivative, and the parameter derivative of its 3\( N \) dimensional Laplacian. To get Eq. \( 16 \) in a form where this is possible, we change the nodal integral to a volume integral. Using the \( z \) coordinate as defined in Sec. II and Gauss’ theorem, we transform Eq. \( 16 \) to

\[
\frac{\partial \delta E}{\partial \theta_i} \approx -\frac{1}{2 \langle \Psi_t | \Psi \rangle} \int_{\text{vol}} dA \lim_{\epsilon \rightarrow 0} \int_{-\epsilon}^{\epsilon} dz \frac{\partial \Psi_t}{\partial \theta_i} \nabla^2 \Psi \quad (17)
\]

where we used the fact that \( |\nabla \Psi| = \frac{\partial \Psi}{\partial z} \) for \( z > 0 \) and \( \Psi = 0 \) for \( z < 0 \).

We replace \( \nabla^2 \) with \( 2(E - H) \) in Eq. \( 17 \) in order to increase the bounds of the \( z \) integral to all space, which is possible since \( (E - H) \Psi = 0 \) everywhere except the nodes. We then integrate by parts to obtain

\[
\frac{\partial \delta E}{\partial \theta_i} \approx \frac{\int_{\text{vol}} dR \langle H - E \rangle \partial \theta_i \Psi_t}{\int_{\text{vol}} dR \Psi_t \Psi} \quad (18)
\]

Interestingly, one can also arrive at Eq. \( 18 \) by taking the derivative of the mixed estimator \( \langle \Psi_t | \Psi \rangle \) with \( \Psi \) fixed.

Method A calculates the integrals of Eq. \( 18 \) from positions \( R_k \) and weights \( w_k \) of a walker distribution equal to \( \Psi_t \Psi \) with the equation

\[
\frac{\partial \delta E}{\partial \theta_i} \approx \sum_k w_k \Psi_t^{-1}(R_k) \langle (H - E) \partial \theta_i | \Psi_t(R_k) \rangle \sum_k w_k. \quad (19)
\]

**II.D. Method B: the energy gradient from a nodal walker distribution**

Method B requires the evaluation of \( \Psi_t \) and its parameter derivative. It samples \( |\nabla \Psi| \) of Eq. \( 16 \) directly from a walker distribution on the nodal surface, which we generate by recording walkers that cross the nodal-surface, and are thus deleted. We note that shifting crossed walker positions closer to the nodes using Newton’s method noticeably improves performance. Unlike those of method A, the pre-crossed walkers cannot be guided by \( \Psi_t \). This is partly because no walkers will cross the nodes of \( \Psi_t \) in the zero time step limit. Instead we guide them with a node-less function \( \Psi_g \).

We wish to relate \( |\nabla \Psi| \) at the nodal-surface to the number of walkers that cross the nodal-surface per area \( dF/dA \) during time \( \Delta \tau \). Let us assume a constant fixed-node walker distribution equal to \( \Psi_g \Psi \). If we suddenly remove the nodal-surface, there will be an initial increase of walker population resulting from walkers that would have crossed the nodal-surface where it present. Therefore the resulting population increase per nodal area to first order in time is \( dF/dA \). Let us make \( \Delta \tau \) small enough that \( \Psi \) does not change at a distance \( \epsilon \) from the node; then, the population increase per nodal area from the released
method B, method C requires the evaluation of \( \Psi_t \) walker distribution used by method B, and samples the right with a factor of II. Method C: the exact energy gradient

\[
\frac{dF(A)}{dA} = \Psi_g(R_A) \int_{-\epsilon}^{\epsilon} \int_0^{\Delta \tau} dz d\tau \frac{d\Psi(A, z)}{d\tau}, \tag{20}
\]

where \( R_A = (A, 0) \). We can rewrite the above as

\[
\frac{dF(A)}{dA} = \Psi_g(R_A) \int_{-\epsilon}^{\epsilon} \int_0^{\Delta \tau} dz d\tau (-H) \Psi(A, z). \tag{21}
\]

We take the \( \epsilon \rightarrow 0 \) limit of Eq. 21 which leaves only a kinetic contribution from the discontinuity of \( \nabla \Psi \):

\[
\frac{dF(A)}{dA} = \Psi_g(R_A) \int_{-\epsilon}^{\epsilon} \int_0^{\Delta \tau} dz d\tau \frac{1}{2} \Delta \tau \nabla^2 \Psi(A, z). \tag{22}
\]

Then, using

\[
\nabla^2 \Psi(A, \epsilon) = \partial^2 \Psi(A, \epsilon)
\]

\[
|\nabla \Psi(A, \epsilon)|^2 = \partial \Psi(A, \epsilon)
\]

\[
|\nabla \Psi(A, -\epsilon)| = 0,
\]

we write

\[
\frac{dF(A)}{dA} = \frac{1}{2} \Delta \tau \Psi_g(R_A) |\nabla \Psi(R_A)|. \tag{24}
\]

We can now use Eq. 24 to sample the \( |\nabla \Psi(R_A)| \) term of Eq. 20 which yields the equation

\[
\frac{\partial E}{\partial \theta_i} \approx -\sum_j w_j \Psi^{-1}(R_j) \partial_{\theta_i} \Psi_i(R_j) \frac{\Delta \tau |\Psi_i(\epsilon)|}{\Delta \tau |\Psi_i(\epsilon)|}, \tag{25}
\]

where \( R_j \) and \( w_j \) are, respectively, the positions and weights of walkers that crossed the nodes during a period \( \Delta \tau \). To sample the denominator, we simply use

\[
\Delta \tau |\Psi_i(\epsilon)| = \int_0^{\Delta \tau} d\tau \sum_k w_k \Psi^{-1}_g(R_k, \tau) w_k(\tau), \tag{26}
\]

with \( R_k \) and \( w_k \) the positions and weights of all walkers as a function of time \( \tau \).

Method B therefore calculates the energy gradient with

\[
\frac{\partial E}{\partial \theta_i} \approx -\sum_j w_j \Psi^{-1}_g(R_j) \partial_{\theta_i} \Psi_i(R_j) \int_0^{\Delta \tau} d\tau \sum_k w_k \Psi^{-1}_g(R_k) \Psi_i(R_k). \tag{27}
\]

II.E. Method C: the exact energy gradient

Method C does not use the approximation of Eq. 20. It samples one factor of \( |\nabla \Psi| \) in Eq. 14 with the same nodal walker distribution used by method B, and samples the other factor of \( |\nabla \Psi| \) using forward walking. Just as with method B, method C requires the evaluation of \( \Psi_t \) and its parameter derivative.

We start by writing the \( \Psi \) projector as a time evolution operator

\[
\frac{\langle \Psi | \Psi \rangle}{|\Psi \rangle} = \lim_{\tau \to \infty} \exp \left[ \tau (E - H) \right] \tag{28}
\]

and contract the left side of Eq. 28 with \( |\Psi_t \rangle \) and the right with \( \langle J | R \rangle \) where

\[
|J | R \rangle = \frac{1}{\Psi_i(R)} |R \rangle. \tag{29}
\]

resulting in

\[
\frac{\langle \Psi_t | \Psi \rangle}{|\Psi \rangle} = \lim_{\tau \to \infty} \langle J | \exp \left[ \tau (E - H) \right] |R \rangle. \tag{30}
\]

We recognize that

\[
\langle J | \exp \left[ \tau (E - H) \right] |R \rangle = \int dR' \tilde{g}(R', R, \tau), \tag{31}
\]

where \( \tilde{g} \), a Green’s function, is the evolution operator of \( \Psi_t \). Because \( \Psi(R)/\Psi_t(R) \) approaches \( |\nabla \Psi(R_A)|/|\nabla \Psi_t(R_A)| \) as \( R \) approaches the nodal surface (since \( \nabla = \partial_z \) at \( z \to 0^+ \)), at the nodal-surface we have

\[
|\nabla \Psi(R_A)| = |\nabla \Psi_t(R_A)| \frac{|\Psi|}{|\Psi_t|} \lim_{\tau \to \infty} \int dR' \tilde{g}(R', R_A, \tau). \tag{32}
\]

We now substitute one \( |\nabla \Psi| \) term of Eq. 13 with Eq. 32 and again use \( \frac{\partial \psi}{\partial \theta_i} = -\frac{1}{|\nabla \Psi|} \frac{\partial |\Psi|}{\partial \theta_i} \) to get

\[
\frac{\partial E}{\partial \theta_i} = \frac{1}{2} \left( |\Psi_i(\epsilon)|^2 \right) \frac{\partial}{\partial \theta_i} \frac{\partial}{\partial \theta_i} |\nabla \Psi(R_A)| \tag{33}
\]

\[
\times \lim_{\tau \to \infty} \int dR' \tilde{g}(R', R_A, \tau). \]

Eq. 33 is identical to Eq. 16 except the added factor \( \lim_{\tau \to \infty} \int dR' \tilde{g}(R', R_A, \tau) \). We sample this factor with the value \( \xi \) we define as the final weight of a walker that had an initial weight of one and an initial position of \( R_A \), and was then propagated for time \( \tau \) with \( \Psi_t \) as the guiding function, that is

\[
\langle \xi(R_A, \tau) \rangle = \int dR' \tilde{g}(R', R_A, \tau). \tag{34}
\]

Instead of taking the \( \tau \to \infty \) limit, we cut the propagation time short at \( \tau_c \). The value we choose should be long enough for \( \langle \xi(R_A, \tau) \rangle \) to become roughly constant.

Placing a walker guided by \( \Psi_t \) at the nodes can be problematic since the local energy and raw drift velocity \( V = \nabla \ln \Psi_t \) are divergent at the nodes when \( \Psi_t \) is used as the guiding function. When calculating \( \xi \), we suppress the effects of both divergences in the standard way, where we modify the drift velocity and weight increase by multiplying them with \( \frac{1 + \sqrt{1 + 4V^2 \delta \tau}}{2} \), where \( \delta \tau \) is the time step. To insure walkers move towards the positive side of the node, we also multiply the drift velocity by \( \text{sign}(\Psi_t) \).

Method C calculates Eq. 33 with node crossing walkers following the same logic we used for method B and Eq. 16 but with the extra factor \( \xi \). We multiply Eq. 27 with \( \xi(R_j, \tau_c) \) to obtain our equation for method C,

\[
\frac{\partial E}{\partial \theta_i} \approx -\sum_j w_j \xi(R_j, \tau_c) \Psi^{-1}_g(R_j) \partial_{\theta_i} \Psi_i(R_j) \int_0^{\Delta \tau} d\tau \sum_k w_k \Psi^{-1}(R_j, \tau_c). \tag{35}
\]

III. IMPLEMENTATION

To test our method, we developed a DMC code with walker propagation that for the most part follows the prescription of Umrigar et al.\(^\text{23}\). In addition, we introduced our PGD iteration method in the code. The main steps are outlined next. We will apply our technique on three systems, atomic Be, Li\(_2\), and atomic Ne.
III.A. Trial and guiding function

The \( \Psi_t \) used for all of our calculations were of the well-known Slater-Jastrow form

\[
\Psi_t = J \Psi_S, \quad \Psi_S = D^+ D^t,
\]

where \( D^+ \) and \( D^t \) are spin-up and spin-down Slater determinants of single-particle orbitals \( \phi_a(r) \) described in the next paragraph. \( J \) is a Jastrow function

\[
J = \prod_{i<j} f_{ij}, \quad f_{ij} = e^{-\alpha_{ij} \tau_{ij}}
\]

where \( r_{ij} \) is the electron-electron distance and \( a_{ij} \) is equal to 1/4 (or 1/2) when both \( i \) and \( j \) correspond to electrons of parallel (or anti-parallel) spin projection.\(^{23}\)

The single-particle orbitals \( \phi \) of the Slater determinants are made of a basis of Slater functions, taking the form

\[
\phi_a(r) = \sum_{b=1}^{N_{basis}} C_{ab} r_b^{n_b-1} e^{-\xi_b r_b} Y_{l_b m_b}(\hat{r}_b),
\]

where \( r_b = |r - R_b| \), and \( n_b, l_b, \) and \( m_b \) are quantum numbers characterizing the \( b \) basis state which is centered at the atomic nuclear position \( R_b \). The parameters we optimize are the \( C_{ab} \) and \( \xi_b \), with \( \xi_b \) being shared by all orbitals.

Since \( J \) is node-less, and, thus, does not affect the energy, our method cannot improve it. We therefore do not include it during PGD since we have no a priori knowledge of its parameters. Although there is a reason to expect that a quality \( J \) will improve the methods A and B, since they rely on a \( \Psi \rightarrow \Psi_t \) approximation, we do not notice such an improvement. Method C on the other hand makes no such approximation, so in principle, it does not need an optimized \( J \). We only include \( J \) when evaluating the energy after optimization.

Our walker distributions for methods B and C use an electron-nuclear Jastrow function as a guiding function \( \Psi_g \). It has the form

\[
\Psi_g = \prod_{i,k} \exp \left( -\frac{Z_k |R_k - r_i|}{1 + |R_k - r_i|} \right),
\]

with \( R_k \) the nuclear coordinate, \( Z_k \) the nuclear charge, and \( r_i \) the electron coordinate. We choose this \( \Psi_g \) instead of \( \Psi_t = 1 \) to reduce the statistical, time-step, and population growth errors. This \( \Psi_g \) also results in many more walkers crossing the node per time, and the electrons are far less likely to ionize. Ionization can still be a problem with poor nodes, when this is the case, we surround the system with a potential barrier.

III.B. Projection on the cusp-condition satisfying parameter space

The electron-nuclear cusp-condition\(^ {24,25} \) is given by

\[
S_k = -\frac{d}{dr} \ln \Psi_t \bigg|_{r=0}, \quad S_k = Z_k,
\]

where \( r_k \) is the radial coordinate from nucleus \( k \), \( Z_k \) is the nuclear charge, and \( \Psi_t^2 \) is defined as the angular average of \( \Psi_t^2 \) about \( r_k = 0 \). It is a necessary requirement to avoid a large divergence of local energy near the nucleus, which gives rise to a large increase of all errors besides the fixed node error. Since the cusps \( S_k \) of our \( \Psi_t \) depend on \( \Psi_S \) and its varied parameters \( \theta_i \), we project \( \theta_i \) back to the manifold of the cusp condition after each iteration of gradient descent.

The cusp condition of \( \Psi_S \) is satisfied when the cusps \( S_{ka} \) of all single-particle orbitals \( \phi_a \) satisfy \( S_{ka} = Z_k \). We assume the shift in parameters is small enough (due to small \( a_i \)) that a linear approximation of the cusp \( \tilde{S}_{ka} \) can be made at the pre-projected parameters \( \theta_{0a} \):

\[
\tilde{S}_{ka}(\theta_i) \equiv S_{ka}(\theta_{0a}) + \sum_i (\theta_i - \theta_{0i}) \frac{\partial S_{ka}(\theta_{0a})}{\partial \theta_i}.
\]

\( \theta_i \) is then shifted to the closest point satisfying

\[
\tilde{S}_{ka}(\theta_i) = \tilde{S}_{ka}(\theta_{0a}) + c(Z_k - \tilde{S}_{ka}(\theta_{0a})),
\]

where \( c \) is added for stability and is between zero and one (we used 0.5). We repeat this until \( S_{ka} - Z_k \) is within a threshold.

III.C. Details of the PGD iterations

After each PGD iteration, we update the energy and effective time-step\(^ {26} \) with data of that iteration. We use the mixed estimate

\[
E = \frac{\langle \Psi_t^* | H | \Psi_t \rangle}{\langle \Psi_t | \Psi_t \rangle},
\]

for method A, and for methods B and C we use the growth estimate

\[
E = \Delta \tau^{-1} \ln \frac{\sum_{\tau=0}^{N_{samples}}}{\sum_{\tau=0}^{N_{samples}}},
\]

where \( \Sigma(\tau) \) is the sum of the weights of all the walkers at imaginary time \( \tau \), ignoring weight normalization.

To calculate the gradient, we propagate the walker distribution \( f \) until the number of samples (walker positions and weights) reaches a threshold. Samples for method A are taken every time-step, and samples for methods B-C are taken from all node-crossings walkers. After an iteration of projected gradient descent, we propagate \( f \) for an extra time (around 0.1 \( \text{Ha}^{-1} \)) to adjust it to the new \( \Psi_t \) before taking samples for the next gradient. This is necessary to do for method A because after a shift of the node, \( f \) does not go to zero at the nodes fast enough for the expectation of the sum of Eq. 17 to be finite, due to the \( \Psi_t^{-1} \) factor. For method A, requiring \( f \) to go to zero sufficiently fast also requires us to use an accept-reject step.

The gradient has stochastic error that prevents PGD of the parameters \( \theta_i \) from fully settling to an accurate minimum, where the signal to noise ratio of the gradient diverge. Instead, \( \theta_i \) continues to fluctuate around the
minimum after some PGD iterations. We argue in Appendix A that to first order in the $a_i$ of Eq. (1), the variance of fluctuations is proportional to both the variance of the gradient error and $a_i$. Thus we can suppress the fluctuations either through smaller $a_i$ and/or by reducing the statistical error of energy derivative with more samples.

An important choice to make is the value of $a_i$. The optimal value depends on many factors, such as the parameters $\theta_i$, the form of $\Psi_t$, the number of PGD iterations, and the stochastic gradient error. Generally, decreasing $a_i$ requires more PGD iterations to reach a minimum, and may exacerbate the problem of getting stuck in local minima. While increasing $a_i$ increases fluctuations from the gradient error and may cause overshooting of $\theta_i$ from its optimum value.

The most basic gradient descent scheme uses a common value of $a_i$ for all parameters and iterations. We experiment with this and with the two adaptive gradient descent algorithms ADAM$^{27}$ and RMSprop$^{28}$. The adaptive algorithms make $a_i$ inversely proportional to a trailing root square mean of past gradient components, and are described more in Appendix B. The $a_i$ of all our algorithms have a common factor which we label the learning rate $\alpha$, which we determine empirically. We sometimes decrease this value with PGD iterations to suppress fluctuations of $\theta_i$. In Appendix C we examine the effective range of $\alpha$ for different gradient descent algorithms and systems.

IV. RESULTS

We tested our methods for three systems, atomic Be, Li$_2$, and atomic Ne, and we compared our results to those obtained by the DMC calculation of Umrigar et al.$^{23}$ where the fermion nodes were optimized using VMC.

IV.A. Derivative of the energy calculation

We present tests of the energy gradient calculation methods A, B, and C in this subsection. They examine one parameter of a basis we added to the simple Be trial-function that is fully described in reference$^{23}$. The basis has quantum numbers $n = 2$, $l = 1$, and $m = 0$, and the parameter is the non-normalized coefficient of the basis. We varied this parameter and obtained its partial first and second energy derivatives by means of a fixed node DMC calculation, which is shown in the upper panel of Fig. 2. As can be noticed, the energy obtained from the fixed node DMC is well approximated with a parabola. We wish to compare the derivative of the parabola with the energy derivatives obtained by our three methods which are shown in the center panel of Fig. 2 and are well approximated with lines intersecting the origin.

The energy derivative obtained by method C agrees with that from the parabola to within 1%, while the energy derivative obtained using methods A and B is proportional to that from the parabola but off by a factor of 2, which we attribute to the approximation of Eq. (14).

We expect that when each of the components of the energy derivative has the correct sign it results in shifting the parameters towards the direction where the energy is lower. In addition, many gradient descent algorithms divide each gradient component by a trailing root-mean-square of past gradient components, thus, the exact value of the slope is not needed by such algorithms. As a result methods A and B may still be useful.

The bottom panel of Fig. 2 shows how the energy derivative obtained with method C depends on the cut off time $\tau_c$. Our choice of $\tau_c$ when optimizing with method C is based on this dependence.

We compare the efficiency of the three methods in Fig. 3 by plotting the error of the energy derivative versus wall-time, which is proportional to the sample size. In this Figure we show the best-fit power laws which, as
Fig. 3 We compare the efficiency of methods A, B, and C by plotting their energy derivative error for the parameter of Fig. 2 as wall-time (sample size) is increased, using a common time step. Best-fit (dotted-lines) power laws are roughly proportional to the inverse square root of sample size. Methods B and C require similar wall-time to achieve the same error, while method A requires on the order of fifty times more.

expected, are approximately proportional to the inverse square root of sample size. Method B and C seem to require similar wall-time to reach a given error level when using $\tau_c = 0.25$ Ha$^{-1}$. Method A on the other hand is considerably more expensive, though a fair comparison is difficult partly because method A may not require the same time step, and the error depends both on the associate parameter and on $\Psi_t$.

IV.B. Projected gradient descent evolution

In order to demonstrate the utility of the method, we start from random values for the parameters $C$ and $\xi$ of all-electron single-determinant wavefunctions of Be, Li$_2$, Ne, and we perform PGD iterations with our cusp condition projection. We present a history of the energy of their nodes in Fig. 4. One can see the DMC energies rapidly decrease with iteration to the energy of the VMC optimized nodal surface obtained by Umrigar et. al. (red dotted lines). Ne continues to fluctuate after initial decrease, which we attribute to statistical error of the energy gradient and an excessively large decent rate ($a_i$ term of Eq. 1). ADAM was used for all examples, and we decreased the descent rate ($\alpha$ parameter of Eq. B3) with iteration number. Method C was used for all except Be, where method A was used, demonstrating the utility of the approximation of Eq. 16.

We chose these three systems because a full description of their VMC optimized wavefunctions was present in the literature, allowing us to directly compare our PGD optimization using wavefunctions of the same form. There are DMC calculations for larger atoms and many atom systems; however, many of the atomic electrons (the inner) are absent in most of these calculations because they are using pseudopotentials and, therefore, do have to implement the cusp condition. Nevertheless, we have applied PGD to a somewhat larger system of an all electron single determinant of F$_2$, shown in Fig. 5. Unfortunately we do not have a wavefunction of the same form optimized by other means for comparison. However, a DMC calculation of F$_2$ was done by Giner et al., who used a configuration interaction optimized wavefunction consisting of $10^5$ determinants. They achieved an energy of -199.2977(1) Ha, while our lowest PGD iteration was -196.7(3) Ha. This discrepancy can partly be attributed to the fact that our wavefunction had just one determinant (a current limitation of our code), and a single particle basis of just 20 Slater type orbitals. The decrease in error of the F$_2$ energy with iteration number suggests that PGD also indirectly reduces local energy variance.
IV.C. Energy upper-bound estimation

The set \( \{E_1, E_2, ..., E_n, \ldots\} \) formed by the values of the energy \( E_i \) at \( i^{th} \) step of the PGD method shown in Fig. 4 is to be taken as a set of energy upper bounds. As already discussed the variations from step to step are due to the fact that the sign of the energy derivative with respect to the PGD parameter \( \alpha \) is sampled and, thus, its choice has some random element. For every such choice, the DMC evolution can be carried out with controlled level of error because the nodal surface is fixed. Therefore, the best energy upper-bound \( \mathcal{E} \) corresponds to the lowest value of the energy in the set, i.e., \( \mathcal{E} = \min\{E_1, E_2, ..., E_n, \ldots\} \). The last iteration energy obtained for the three systems studied in this work are compared with the best DMC energy obtained with nodes determined by a VMC optimization\(^{5}\) in Table I.

The present calculation suggests that the nodes obtained with VMC optimization are not far from the optimum at least within the error of the present calculation. Therefore, the small discrepancy with the exact nonrelativistic values could be attributed to the fact that the form of the one-body factor is limited with the space of a single Slater determinant and the lack of spin-spin correlations because the one-body factor is taken to be a product of the form \( D^\uparrow D^\downarrow \).

| System | Last PGD iteration | VMC optimized nodes | Exact energy |
|--------|-------------------|--------------------|-------------|
| Be     | -14.6567(5)       | -14.6571(1)        | -14.66736   |
| Li\(\text{\textit{II}}\) | -14.989(1)       | -14.9898 (1)       | -14.9954    |
| Ne     | -128.92(1)        | -128.919 (3)       | -128.939    |

Table I DMC energy comparison between the last PGD iteration nodes with VMC optimized nodes. Exact nonrelativistic values are present.

V. DISCUSSION

We are not yet comfortable claiming one of methods A, B, or C is ideal, and the best choice may depend on circumstance. For example, although method A was least efficient in our test, it may not require as small of a time step, and we think it would be the simplest to implement into existing code due to its use of a standard walker distribution. Although the efficiency of method C was comparable to that of method B in our test, it requires additional wall-time to calculate \( \xi \), which also introduces the additional parameter \( \tau_c \) which we do not yet have a good procedure for determining.

Nevertheless, method C performed the best in our tests and also has the advantage of not relying on the quality of \( \Psi_0 \). We recommend pairing method C with ADAM. In our tests we were able to optimize the location of the nodes to roughly the same energy as with the VMC optimized nodes using any of the methods and without using Jastrow factors in our guiding function.

Without the requirement of an optimized Jastrow function, our method is independent of prerequisite optimization by VMC. Nevertheless, simultaneously optimizing the Jastrow and Slater parameters may be an fruitful avenue, as a quality Jastrow function reduces many sources of error (except for the fixed node error) and should improve the approximation of methods A and B. This could be done with the standard VMC method, or possibly with DMC and gradient descent, which require a gradient different than one of the DMC energy, perhaps one of the local energy variance.

An important advantage of PGD optimization is that the use of DMC propagation produces correct correlations in the fixed-node wavefunction, whereas VMC optimizes the nodal surface within a limited Jastrow form, which captures only pairwise correlations. Thus, the exact path to optimization due to the interplay between the adjustment of these correlations as the position of the nodal surface changes and vice versa was not fully accounted in previous DMC studies.

We would like to give an example of a system which demonstrates the significance of the present work. Variational calculations of liquid \( ^3\text{He} \), where just a Jastrow-Slater wavefunction is used, yield\(^ {32}\) an un-physical result that the fully polarized liquid is of lower energy at its equilibrium density than the unpolarized liquid at its equilibrium density. Adding three-body correlation factors in the wave-function does not change the above conclusion. It is when one includes backflow correlations, which are state-dependent and modify the nodal surface of the variational wavefunction, that one finds\(^ {32}\) that the unpolarized state of liquid \( ^3\text{He} \) is energetically favorable.

Therefore, let us pretend that we do not know the fact that the naturally occurring liquid \( ^3\text{He} \) is spin-unpolarized; we might then be naive and use a Slater-Jastrow variational calculation to determine the optimum polarization. The nodes that would be determined by such a VMC optimization would be those that correspond to a polarized liquid. A subsequent fixed node DMC calculation with these nodes is not going to change this result, and we would reach an unphysical conclusion. The PGD method described in the present paper, however, should find that the unpolarized state is the ground
state.

Another advantage of PGD is that it requires only one gradient calculation to shift all the parameters in the correct direction. Whereas if one were to vary the parameters and select values with lower calculated DMC energy, the number of required energy calculations would scale with the number of parameters used. We should point out that in our PGD evolution examples, far more CPU time was spent calculating the energy of a single nodal surface than the entire PGD process.

PGD optimization opens up a possible way to optimize atomic positions to minimize DMC energy. If one were to combine the nuclei and electrons into one walker and try to minimize DMC energy, it would be possible to combine the nuclei and electrons into one walker to optimize atomic positions to minimize DMC energy. If one were to vary the parameters, requiring the atoms be fixed.

VI. SUMMARY

We presented a general and self-reliant method using DMC and projected gradient descent that optimizes the nodal surface of a trial function to minimize the fixed-node ground state, i.e., the DMC energy. We derived three methods for calculating the DMC energy gradient from walker distribution (methods A, B, and C). Methods B and C required comparable CPU time in our test, while method A was more expensive.

We combined the three methods with several gradient descent algorithms and a projection operation that maintains the cusp condition. We benchmarked this projected gradient descent method to trial functions with randomized parameters of Be, Li$_2$, and Ne. Their energies were lowered to the same level as VMC optimized parameters, without the use of a Jastrow factor.

Our work is a proof of concept using simple systems, but we see no reason it cannot be applied to larger and periodic systems, or be implemented into existing DMC code for various types of node determining wavefunctions. However, before attempting these goals, our work may be further improved with better and more adaptive choices for various parameters, by simultaneously optimizing Jastrow parameters during PGD, by better suppressing PGD fluctuations, and by establishing criteria for convergence. In addition, we see no reason why the method could not be extended to DMC on a discrete lattice or to finite temperature path integral Monte Carlo.

VII. ACKNOWLEDGMENTS

We would like to thank the Florida State University high performance computing center for the allocation of CPU time to carry out the calculations presented in this work.

Appendix A: Parameter fluctuations

After a number of PGD iterations, the parameters will continue to fluctuate around a local minimum due to stochastic error of the energy gradient. Here, we investigate the variance of these fluctuations. First let us perform a coordinate shift so that the parameters are zero at the local minima, then let us rotate them so that the Hessian $\frac{\partial^2 E}{\partial \theta_i \partial \theta_j}$ is diagonal. Then, if we expand $E$ to second order in $\theta_i$, one iteration of gradient descent using Eq. 1 will result in new parameters $\theta'_i$ given by

$$\theta'_i = \theta_i - a_i (K_i \theta_i + \sigma_i \eta),$$

where $K_i = \frac{\partial^2 E}{\partial \theta_i^2}$, $\sigma_i$ is the standard deviation of stochastic error, and $\eta$ is a random number with $\langle \eta \rangle = 0$ and $\langle \eta^2 \rangle = 1$.

Let us take the variance of both sides of Eq. A1

$$\langle \theta_i'^2 \rangle = \langle \theta_i^2 \rangle + a_i^2 \left( K_i^2 \langle \theta_i^2 \rangle + \sigma_i^2 \right) - 2a_i K_i \langle \theta_i^2 \rangle.$$

After enough PGD iterations, the variance of fluctuations will become constant, i.e.,

$$\langle \theta_i'^2 \rangle = \langle \theta_i^2 \rangle.$$

If this is the case, we can use Eq. A1 to evaluate the variance of latter PGD iterations, namely,

$$\langle \theta_i'^2 \rangle = \frac{a_i \sigma_i^2}{2K_i - a_i K_i^2}.$$

We notice that the variance blows up if any $a_i$ approach $2/K_i$, which is the result of overshooting the minimum by more than double the distance. For $a_i << 2/K_i$ we see that the variance has the property

$$\langle \theta_i'^2 \rangle \propto a_i \sigma_i^2.$$

Appendix B: Gradient descent algorithms

We list the gradient descent algorithms we tested. We use $g_i \equiv \frac{\partial E}{\partial \theta_i}$ as the gradient, and determine $\alpha$ empirically. We sometimes decrease $\alpha$ with iteration number to suppress parameter fluctuations around the minimum.

Basic gradient descent:

$$\theta_i \rightarrow \theta_i - \alpha g_i,$$

RMSprop:

$$v_i \rightarrow \beta v_i + (\beta - 1)g_i^2,$$

$$\theta_i \rightarrow \theta_i - \frac{\alpha g_i}{\sqrt{v_i}},$$

ADAM:

$$t \rightarrow t + 1,$$

$$m_i \rightarrow \beta_1 m_i + (\beta_1 - 1)g_i,$$

$$v_i \rightarrow \beta_2 v_i + (\beta_2 - 1)g_i^2,$$

$$\theta_i \rightarrow \theta_i - \frac{\alpha m_i g_i}{(1 - \beta_1^t) \sqrt{v_i} / (1 - \beta_2^t)}.$$
Appendix C: Effective $\alpha$

In Table II we show the effective ranges of the learning rate $\alpha$, which is a hyperparameter proportional to a shift in parameters, as shown in Appendix B. We define this effective range as the values that bring the energy of a randomized $\Psi_S$ down to the optimum level, and show an example in Fig. 6. We empirically determine this range for 100 PGD iterations for different combinations of gradient descent algorithms, atomic systems, and methods A, B or C. We display the end-points of the effective range along with the logarithmic distance between them.

Compared to basic gradient descent, the adaptive algorithms RMSprop and ADAM have more consistent endpoints for methods A, B and C and for different atomic systems, making the choice of $\alpha$ easier. We attribute the more consistent endpoints to the division of each gradient component by a trailing root square mean of its gradients $\alpha$.

Table I. Learning rate $\alpha$ for different combinations of gradient-descent algorithms, atomic systems, and methods A, B, or C. We display the end-points of the effective range along the logarithmic distance between them.

Table I

| Algorithm       | Gradient method | $\varphi_S$       | Minimum effective $\alpha$ | Maximum effective $\alpha$ | $\log \max \frac{\alpha_{\text{max}}}{\alpha_{\text{min}}}$ |
|-----------------|-----------------|-------------------|-----------------------------|-----------------------------|-------------------------------------------------|
| Basic gradient  | Basic           | Be                | $1.2 \times 10^{-4}$       | $1.3 \times 10^{-1}$        | 1.0                                             |
| descent         | B               | Be                | $2.4 \times 10^{-3}$       | $6.4 \times 10^{-1}$        | 1.4                                             |
|                 | A               | Li$_2$            | $1.8 \times 10^{-3}$       | $2.0 \times 10^{-1}$        | 1.6                                             |
|                 | B               | B                 | $4.5 \times 10^{-2}$       | $1.8 \times 10^{-1}$        | 1.6                                             |
| RMSprop         | B               | Be                | $5.5 \times 10^{-2}$       | $1.0 \times 10^{-1}$        | 1.3                                             |
|                 | C               | Be                | $2.0 \times 10^{-2}$       | $1.0 \times 10^{-1}$        | 1.7                                             |
|                 | A               | Li$_2$            | $1.8 \times 10^{-3}$       | $2.1 \times 10^{-1}$        | 5.1                                             |
|                 | B               | B                 | $5.6 \times 10^{-2}$       | $1.5 \times 10^{-1}$        | 1.4                                             |
| ADAM            | B               | Be                | $3.8 \times 10^{-2}$       | $1.0 \times 10^{-1}$        | 1.4                                             |
|                 | C               | Be                | $1.1 \times 10^{-2}$       | $1.0 \times 10^{-1}$        | 2.0                                             |
|                 | A               | Li$_2$            | $1.6 \times 10^{-1}$       | $7.0 \times 10^{-2}$        | 3.6                                             |
|                 | B               | B                 | $2.4 \times 10^{-1}$       | $9.2 \times 10^{-1}$        | 1.6                                             |

Table II. The range of the effective learning rate $\alpha$ for 100 PGD iterations for three gradient descent algorithms: basic gradient descent (common $\alpha_1$), RMSprop, and ADAM (see Fig. 6, for example in the tenth row). RMSprop used $\beta = 0.99$. ADAM used $\beta_1 = 0.9$, $\beta_2 = 0.99$. Method C used $\tau_c = 0.3$ Ha$^{-1}$.

\[E \tau\]

The difference between estimated and correct energy.

Suppresses large variations of $\xi$ magnitudes for different atomic systems. It also

Component by a trailing root square mean of its
gradient $RMSprop$ and $ADAM$ have more consistent end-

range along with the logarithmic distance between them.

For 100 PGD iterations for different combinations of gra-

example in Fig. 6. We empirically determine this range

randomized $\Psi$ in parameters, as shown in Appendix B. We define this

range of the learning $\alpha$, which is a hyper parameter proportional to a shift in parameters, as shown in Appendix B.

We empirically determine this range for

in Table II we show the effective ranges of the learning rate $\alpha$, which is a hyperparameter proportional to a shift in parameters, as shown in Appendix B. We define this effective range as the values that bring the energy of a randomized $\Psi_S$ down to the optimum level, and show an example in Fig. 6. We empirically determine this range for 100 PGD iterations for different combinations of gradient descent algorithms, atomic systems, and methods A, B or C. We display the end-points of the effective range along with the logarithmic distance between them.

Compared to basic gradient descent, the adaptive algorithms RMSprop and ADAM have more consistent endpoints for methods A, B and C and for different atomic systems, making the choice of $\alpha$ easier. We attribute the more consistent endpoints to the division of each gradient component by a trailing root square mean of its gradients $\alpha$.

Table I

| Algorithm       | Gradient method | $\varphi_S$       | Minimum effective $\alpha$ | Maximum effective $\alpha$ | $\log \max \frac{\alpha_{\text{max}}}{\alpha_{\text{min}}}$ |
|-----------------|-----------------|-------------------|-----------------------------|-----------------------------|-------------------------------------------------|
| Basic gradient  | Basic           | Be                | $1.2 \times 10^{-4}$       | $1.3 \times 10^{-1}$        | 1.0                                             |
| descent         | B               | Be                | $2.4 \times 10^{-3}$       | $6.4 \times 10^{-1}$        | 1.4                                             |
|                 | A               | Li$_2$            | $1.8 \times 10^{-3}$       | $2.0 \times 10^{-1}$        | 1.6                                             |
|                 | B               | B                 | $4.5 \times 10^{-2}$       | $1.8 \times 10^{-1}$        | 1.6                                             |
| RMSprop         | B               | Be                | $5.5 \times 10^{-2}$       | $1.0 \times 10^{-1}$        | 1.3                                             |
|                 | C               | Be                | $2.0 \times 10^{-2}$       | $1.0 \times 10^{-1}$        | 1.7                                             |
|                 | A               | Li$_2$            | $1.8 \times 10^{-3}$       | $2.1 \times 10^{-1}$        | 5.1                                             |
|                 | B               | B                 | $5.6 \times 10^{-2}$       | $1.5 \times 10^{-1}$        | 1.4                                             |
| ADAM            | B               | Be                | $3.8 \times 10^{-2}$       | $1.0 \times 10^{-1}$        | 1.4                                             |
|                 | C               | Be                | $1.1 \times 10^{-2}$       | $1.0 \times 10^{-1}$        | 2.0                                             |
|                 | A               | Li$_2$            | $1.6 \times 10^{-1}$       | $7.0 \times 10^{-2}$        | 3.6                                             |
|                 | B               | B                 | $2.4 \times 10^{-1}$       | $9.2 \times 10^{-1}$        | 1.6                                             |

Table II. The range of the effective learning rate $\alpha$ for 100 PGD iterations for three gradient descent algorithms: basic gradient descent (common $\alpha_1$), RMSprop, and ADAM (see Fig. 6, for example in the tenth row). RMSprop used $\beta = 0.99$. ADAM used $\beta_1 = 0.9$, $\beta_2 = 0.99$. Method C used $\tau_c = 0.3$ Ha$^{-1}$.

\[E \tau\]

The difference between estimated and correct energy.

Suppresses large variations of $\xi$ magnitudes for different atomic systems. It also

Component by a trailing root square mean of its

easier. We attribute the more consistent endpoints to the division of each gradient component by a trailing root square mean of its prior values. This suppresses the large variations of gradient magnitudes for different atomic systems. It also suppresses large variations of $\xi(R, \tau_c)$ of Eq. ?? used for method C, which has a factor of $\exp(\Delta E \tau_c)$, with $\Delta E$ the difference between estimated and correct energy.

![Fig. 6](image-url) An example of how the energy, calculated after a 100 PGD iterations, depends on different learning rates $\alpha$. This data was used for the 6th row of Table II.
24 T. Kato, “On the eigenfunctions of many-particle systems in quantum mechanics,” Communications on Pure and Applied Mathematics 10, 151–177 (1957), https://onlinelibrary.wiley.com/doi/pdf/10.1002/cpa.3160100201
25 E. Steiner, “Charge densities in atoms,” The Journal of Chemical Physics 39, 2365–2366 (1963), https://doi.org/10.1063/1.1701443
26 P. J. Reynolds, D. M. Ceperley, B. J. Alder, and W. A. Lester, “Fixed-node quantum monte carlo for molecules,” The Journal of Chemical Physics 77, 5593–5603 (1982)
27 D. P. Kingma and J. Ba, “Adam: A method for stochastic optimization,” (2017), arXiv:1412.6980 [cs.LG]
28 G. Hinton, “Neural networks for machine learning,” University Lecture.
29 E. Giner, A. Scemama, and M. Caffarel, “Fixed-node diffusion monte carlo potential energy curve of the fluorine molecule F2 using selected configuration interaction trial wavefunctions,” The Journal of Chemical Physics 142, 044115 (2015), https://doi.org/10.1063/1.4905528
30 D. Bressanini, G. Morosi, and S. Tarasco, “An investigation of nodal structures and the construction of trial wave functions,” The Journal of Chemical Physics 123, 204109 (2005), https://doi.org/10.1063/1.2128672
31 E. R. Davidson, S. A. Hagestrom, S. J. Chakravorty, V. M. Umar, and C. F. Fischer, “Ground-state correlation energies for two- to ten-electron atomic ions,” Phys. Rev. A 44, 7071–7083 (1991)
32 E. Manousakis, S. Fantoni, V. R. Pandharipande, and Q. N. Usmani, “Microscopic calculations for normal and polarized liquid 3He,” Phys. Rev. B 28, 3770–3781 (1983)