Dynamic Learning of Correlation Potentials for a Time-Dependent Kohn-Sham System

Harish S. Bhat HBHAT@UCMERCED.EDU
Department of Applied Mathematics, University of California Merced

Kevin Collins KCOLLINS9@UCMERCED.EDU
Department of Physics, University of California Merced

Prachi Gupta PGUPTA11@UCMERCED.EDU
Department of Applied Mathematics, Department of Chemistry and Biochemistry, University of California Merced

Christine M. Isborn CISBORN@UCMERCED.EDU
Department of Chemistry and Biochemistry, University of California Merced

Editors: R. Firoozi, N. Mehr, E. Yel, R. Antonova, J. Bohg, M. Schwager, M. Kochenderfer

Abstract

We develop methods to learn the correlation potential for a time-dependent Kohn-Sham (TDKS) system in one spatial dimension. We start from a low-dimensional two-electron system for which we can numerically solve the time-dependent Schrödinger equation; this yields electron densities suitable for training models of the correlation potential. We frame the learning problem as one of optimizing a least-squares objective subject to the constraint that the dynamics obey the TDKS equation. Applying adjoints, we develop efficient methods to compute gradients and thereby learn models of the correlation potential. Our results show that it is possible to learn values of the correlation potential such that the resulting electron densities match ground truth densities. We also show how to learn correlation potential functionals with memory, demonstrating one such model that yields reasonable results for trajectories outside the training set.

Keywords: Physics-constrained learning, adjoint methods, quantum dynamics, TDDFT.

1. Introduction

The time-dependent Schrödinger equation (TDSE) governs the behavior of N quantum particles,

$$i\partial_t \Psi(r_1, r_2, ..., r_N, t) = \hat{H}(r_1, r_2, ..., r_N, t)\Psi(r_1, r_2, ..., r_N, t), \tag{1}$$

where $\hat{H}$ is the Hamiltonian and Ψ is the many-body wave function. In d-dimensional space, the many-body Coulomb interaction in the potential term of $\hat{H}$ leads to a coupled system of partial differential equations (PDE) in $dN + 1$ variables. Hence (1) can only be solved for simple model problems, such as for one electron in three dimensions or two electrons in one dimension. To simulate electron dynamics in molecules and materials, a widely used approach is time-dependent density functional theory (TDDFT), in which the many-body wave function Ψ is replaced with the Kohn-Sham wave function $\Phi(r)$ to give the time-dependent Kohn-Sham (TDKS) equation (Maitra, 2016; Ullrich, 2011):

$$i\partial_t \Phi(r,t) = \sum_{i=1}^{N} [-(1/2)\nabla_i^2 + v^{\text{ext}}(r_i,t) + v^H[n](r_i,t) + v^{XC}[n,\Psi_0,\Phi_0](r_i,t)]\Phi(r,t). \tag{2}$$

Because $\Phi(r)$ is constructed as a product of non-interacting single-particle orbitals $\phi(r_i)$, (2) decouples into N separate evolution equations in $3+1$ variables. Assuming all terms in (2) are specified, one can use (2) to simulate molecular systems for which numerical simulation of (1) is intractable.

In (2), the many-body Coulomb interaction between electrons is replaced by known classical Hartree v^H and unknown exchange-correlation v^{XC} single-particle potentials, with the latter incorporating many-body effects. TDDFT is formally an exact theory, as the Runge-Gross and Van Leeuwen theorems proved the existence of a time-dependent electronic potential and the unique mapping to the time-dependent electron density, which is generated from the KS orbitals of the TDKS equation (Runge and Gross, 1984; van Leeuwen, 1999).

The challenge in TDDFT is to construct v^{XC} potentials that yield an electron density n that is identical to the exact time-dependent many-body electron density generated from the TDSE. Previous work has shown that the unknown v^{XC} formally depends on the initial many-body wave function Ψ_0, the initial KS state Φ_0, and the electron density at all points in time $n(r, s < t)$ (Maitra et al., 2002). Although the development of v^{XC} for electrons is a very active area of research, almost all v^{XC} make use of the so-called "adiabatic approximation" that only takes into account the instantaneous electron density, leading to significant inaccuracies in electron dynamics due to the lack of memory in v^{XC}. *The desire for more accurate electron dynamics leads to a natural question: can we learn v^{XC} from time series data?* Note that this is an entirely different problem than the problem of learning static, ground state potentials from the exact ground state electron density in *time-independent* density functional theory (DFT) (Nagai et al., 2018; Kalita et al., 2021).

For machine learning of v^{XC} to proceed in the *time-dependent* context (TDDFT), a first obstacle is formulating a tractable learning problem. In recent work, Suzuki et al. (2020) works with a *spatially one-dimensional* electron-hydrogen scattering problem. For this model problem, one can solve (1) numerically; from the solution, one can compute the electron density $n(x,t)$ on spatial/temporal grids. In this problem, we know both the functional form of v^X and that $v^{XC} = v^X + v^C$. Furthermore, the one-dimensionality enables one to solve for exact values of v^C (Elliott et al., 2012), again on spatial/temporal grids. With grid-based values of both v^C and n, the task of learning $v^C[n]$ becomes a static, supervised learning problem, which Suzuki et al. (2020) solves using neural network models. To our knowledge, this is the only prior work on learning v^{XC} for TDDFT.

We revisit the electron-hydrogen scattering model problem and develop methods to learn $v^C[n]$ that do not require us to solve for grid-based values of v^C beforehand. *In short, we view the v^C functional as a control that guides TDKS propagation.* We formulate the learning problem as an optimal control problem: find v^C that minimizes the squared error between TDKS electron densities n and reference electron densities $\tilde{n}$. Implicit in this formulation is the dynamical constraint that electron densities n evolve forward in time via the TDKS equation with the model v^C. The adjoint or costate method is often used to handle constraints of this kind (Bryson and Ho, 1975; Hasdorff, 1976). To our knowledge, the derivations and applications of the adjoint method, *to learn v^C models with memory for the TDKS equation*, are considered here for the first time[1]. We derive adjoint systems for two settings: (i) to learn pointwise values of v^C on a grid, and (ii) to learn the functional dependence of v^C on the electron density at two points in time. We apply our methods to train both types of models, and study their training and test performance. In particular, we train a neural network model of $v^C[n]$ with memory that, when used to solve the TDKS equations for initial conditions outside the training set, yields qualitatively accurate predictions of electron density.

1. See Section 5.7 in the Appendix of the preprint https://arxiv.org/pdf/2112.07067 for further context.

2. Methods

To formulate the problem of learning v^C from time series, we first work in continuous space and time. Later, to derive numerical algorithms to solve this problem, we discretize.

Continuous Problem. Define the 1D electron density created from KS orbitals

$$n(x,t) = 2|\phi(x,t)|^2, \tag{3}$$

and the soft-Coulomb external v^{ext} and interaction W^{ee} potentials

$$v^{\text{ext}}(x) = -((x+10)^2+1)^{-1/2}, \tag{4a}$$
$$W^{ee}(x',x) = ((x'-x)^2+1)^{-1/2}. \tag{4b}$$

The potentials (4a) and (4b) specify that we are working with the spatially one-dimensional electron-hydrogen scattering problem considered by several previous authors. For this problem, we know that $v^{XC} = v^X + v^C$. Let ϕ and n stand for $\phi(x,t)$ and $n(x,t)$. Then in one spatial dimension and expressed in atomic units (a.u.), the TDKS system (2) becomes:

$$i\partial_t\phi = -\frac{1}{2}\partial_{xx}\phi + v^{\text{ext}}(x,t)\phi + v^H[n](x,t)\phi + v^X[n](x,t)\phi + v^C[\phi](x,t)\phi, \tag{5a}$$
$$v^H[n](x,t) = \int_{x'} W^{ee}(x',x)n(x',t)\,dx', \qquad v^X[n](x,t) = -\frac{1}{2}v^H[n](x,t). \tag{5b}$$

In (5), the term that we are trying to learn (e.g., the control) is $v^C[\phi]$. Prior first principles work has shown that at time t, v^C should depend functionally on the electron density $n(x,s)$ for $s \le t$, the initial Kohn-Sham state $\phi(x,0)$ and the initial Schrödinger wave function $\Psi(x,0)$ (Maitra et al., 2002; Wagner et al., 2012). In this work, we ignore the dependence of v^C on the initial states $\phi(x,0)$ and $\Psi(x,0)$, and focus on modeling the dependence on present and past electron densities. By (3), dependence on n is equivalent to a particular type of dependence on ϕ; we use the notation $v^C[\phi]$ to refer to models that depend on ϕ either directly or through n.

For the sake of intuition, let us formulate the control problem in continuous time and space. Assume that for $t \in [0,T]$, we have access to a reference electron density trajectory $\tilde{n}(x,t)$. Suppose that our model $v^C[\phi;\boldsymbol{\theta}]$ is parameterized by $\boldsymbol{\theta}$. Then we seek to minimize the squared loss

$$\mathcal{J}(\boldsymbol{\theta}) = \frac{1}{2}\int_{x=-\infty}^{\infty}\int_{t=0}^{T}(n(x,t)-\tilde{n}(x,t))^2\,dt\,dx, \tag{6}$$

subject to the constraint that $n(x,t)$ is computed via (3), with $\phi(x,t)$ evolving on the interval $0 \le t \le T$ according to the TDKS system (5). In this TDKS system, we identify v^C with our model $v^C[\phi;\boldsymbol{\theta}]$. *In short, we seek $\boldsymbol{\theta}$ such that the resulting $v^C[\phi;\boldsymbol{\theta}]$ functional guides the TDKS system to yield a solution $\phi(x,t)$ such that $n = 2|\phi|^2$ matches the reference trajectory $\tilde{n}(x,t)$.*

Direct and Adjoint Methods. In a *direct method* to minimize the loss (6), we compute gradients by applying $\nabla_{\boldsymbol{\theta}}$ to both sides of (6). This will yield an expression for $\nabla_{\boldsymbol{\theta}}\mathcal{J}$ that involves $\nabla_{\boldsymbol{\theta}}\phi$. To compute this latter quantity, we numerically solve an evolution equation derived by taking $\nabla_{\boldsymbol{\theta}}$ of both sides of (5a). At each iteration of our gradient-based optimizer, we would carry out this procedure to compute $\nabla_{\boldsymbol{\theta}}\mathcal{J}$, which is then used to update $\boldsymbol{\theta}$. In practice, this direct method suffers

from one major problem: if we discretize ϕ in space using $J+1$ grid points, and if $\boldsymbol{\theta}$ has dimension B, then at each point in time, $\nabla_{\boldsymbol{\theta}}\phi$ will have dimension $(J+1)B$. In our work, B can exceed 10^7, while $J \geq 600$ is required for sufficient spatial accuracy. Solving the evolution equation for $\nabla_{\boldsymbol{\theta}}\phi$ in $(J+1)B$-dimensional space thus incurs huge computational expense at each optimization step.

In this paper, we pursue the *adjoint method*, which enables us to compute all required gradients without computing or even storing any $(J+1)B$-dimensional objects, thus dramatically reducing computational costs relative to the direct method. Within the space of adjoint methods, there are two broad approaches: (i) to use the continuous-time loss and constraints to derive differential equations for continuous-time adjoint variables, and (ii) to first discretize the loss and constraints, and then derive numerical schemes for discrete-time adjoint variables. In approach (i), we must still discretize the adjoint differential equations in order to solve them; the choice of discretization can lead to subtle issues (Sanz-Serna, 2016). We choose approach (ii) for its relative simplicity.

In the discrete adjoint method, we incorporate a discretized version of the dynamical system (5) as a constraint using time-dependent Lagrange multipliers $\boldsymbol{\lambda}(t)$. In this approach, we derive and numerically solve a backward-in-time evolution equation for $\boldsymbol{\lambda}(t)$, from which we compute required gradients. Importantly, $\boldsymbol{\lambda}(t)$ has the same dimension as the state variables $\boldsymbol{\phi}(t)$ defined below; in our implementation, both quantities are $(J+1)$-dimensional. We obtain the gradients of the discretized loss at a computational cost that is proportional to that of computing the loss itself.

Discretized Problem. To keep this paper focused on the learning/control problem, we have moved details of the numerical solution of the TDKS system (5) to Section 5.1 of the Appendix[2]. Here we include only the most important concepts. First, we discretize the Kohn-Sham state by introducing $\boldsymbol{\phi}(t_k) = [\phi(x_0, t_k), \ldots, \phi(x_J, t_k)]^T$. The spatial domain is $x \in [L_{\text{min}}, L_{\text{max}}]$. With $\Delta x = (L_{\text{max}} - L_{\text{min}})/J > 0$, our spatial grid is $x_j = L_{\text{min}} + j\Delta x$. Our temporal grid is $t_k = k\Delta t$, with $\Delta t = T/K$. The positive integers J and K are user-defined parameters that control the accuracy of the discretization. Second, by applying finite differences, Simpson's quadrature rule, and operator splitting, we can derive the following evolution equation for the discretized state $\boldsymbol{\phi}$ defined above:

$$\boldsymbol{\phi}(t_{k+1}) = \exp(-i\mathcal{K}\Delta t/2)\exp(-iV(\boldsymbol{\phi}(t_k), \boldsymbol{v}_k^C)\Delta t)\exp(-i\mathcal{K}\Delta t/2)\boldsymbol{\phi}(t_k). \tag{7}$$

Here $\mathcal{K}$ is a constant $(J+1)\times(J+1)$ matrix, while V is a diagonal $(J+1)\times(J+1)$ matrix that depends functionally on both the state $\boldsymbol{\phi}$ and on $\boldsymbol{v}^C$, our spatially discretized model of the correlation potential v^C from (5). Detailed descriptions of $\mathcal{K}$ and V are provided in Section 5.1.

Evolving $\boldsymbol{\phi}$ according to (7) generates a numerical approximation to the solution $\phi(x,t)$ of (5). This approximation has a truncation error of $O(\Delta t^2)$ in time and $O(\Delta x^4)$ in space.

First Adjoint Method: Learning v^C Pointwise. Assume we have access to observed values of electron density on the grid—we denote these observed or reference values by $\tilde{n}(x_j, t_k)$. The first problem we consider is to learn $v^C(x_j, t_k)$ on the same grid. Suppose we start from an initial condition $\boldsymbol{\phi}(0)$ and an estimate $\boldsymbol{v}^C$. We iterate (7) forward in time and obtain a trajectory $\boldsymbol{\phi}(t_k)$ for $0 \leq k \leq K$. We then form $n(x_j, t_k) = |\phi(x_j, t_k)|^2$. In this subsection, ϕ and n are the predicted wave function and density when we use the estimated correlation potential $\boldsymbol{v}^C$. Let $\mathcal{P}_{\mathcal{K}} = \exp(-i\mathcal{K}\Delta t/2)$ and abbreviate $\boldsymbol{\phi}_k = \boldsymbol{\phi}(t_k)$, $\boldsymbol{v}_k^C = \boldsymbol{v}^C(t_k)$. Define the discrete-time propagator

$$\boldsymbol{F}_{\Delta t}(\boldsymbol{\phi}, \boldsymbol{v}^C) = \mathcal{P}_{\mathcal{K}}\exp(-iV(\boldsymbol{\phi}, \boldsymbol{v}^C)\Delta t)\mathcal{P}_{\mathcal{K}}\boldsymbol{\phi}, \tag{8}$$

2. Henceforth, for Section 5.x or the Appendix, see https://arxiv.org/pdf/2112.07067.

so that (7) can be written as the discrete-time system $\boldsymbol{\phi}_{k+1} = \boldsymbol{F}_{\Delta t}(\boldsymbol{\phi}_k, \boldsymbol{v}_k^C)$. Both sides of this system are complex-valued. In order to form a real-valued Lagrangian and take real variations, we split both $\boldsymbol{\phi}$ and $\boldsymbol{F}$ into real and imaginary parts: $\boldsymbol{\phi} = \boldsymbol{\phi}^R + i\boldsymbol{\phi}^I$ and $\boldsymbol{F}_{\Delta t} = \boldsymbol{F}_{\Delta t}^R + i\boldsymbol{F}_{\Delta t}^I$. Superscript R and I denote, respectively, the real and imaginary parts of a complex quantity. Let the uppercase $\boldsymbol{\Phi}$, $\boldsymbol{\Lambda}$, and $\boldsymbol{V}^C$ denote the collections of all corresponding lowercase $\boldsymbol{\phi}_k$, $\boldsymbol{\lambda}_k$, and $\boldsymbol{v}_k^C$ for all k. Then we form a real-variable Lagrangian that consists of the discretized squared loss with the constraint that $\boldsymbol{\phi}$ evolves via (7).

$$\begin{aligned}\mathscr{L}(\boldsymbol{\Phi}^R, \boldsymbol{\Phi}^I, \boldsymbol{\Lambda}^R, \boldsymbol{\Lambda}^I, \boldsymbol{v}^C) = \frac{1}{2}\sum_{k=0}^{K}\sum_{j=0}^{J}(2\phi^R(x_j,t_k)^2 + 2\phi^I(x_j,t_k)^2 - \tilde{n}(x_j,t_k))^2 \\ - \sum_{k=0}^{K-1}[\boldsymbol{\lambda}_{k+1}^R]^T(\boldsymbol{\phi}_{k+1}^R - \boldsymbol{F}_{\Delta t}^R(\boldsymbol{\phi}_k^R, \boldsymbol{\phi}_k^I, \boldsymbol{v}_k^C)) + [\boldsymbol{\lambda}_{k+1}^I]^T(\boldsymbol{\phi}_{k+1}^I - \boldsymbol{F}_{\Delta t}^I(\boldsymbol{\phi}_k^R, \boldsymbol{\phi}_k^I, \boldsymbol{v}_k^C)). \quad (9)\end{aligned}$$

Setting $\delta\mathscr{L} = 0$ for all variations $\delta\boldsymbol{\phi}_k^R$ and $\delta\boldsymbol{\phi}_k^I$ for $k \geq 1$, we obtain

$$\boldsymbol{\lambda}_K = 4\left[(2|\boldsymbol{\phi}_K|^2 - \tilde{\boldsymbol{n}}_K) \circ \boldsymbol{\phi}_K\right] \quad (10a)$$

$$\begin{bmatrix}\boldsymbol{\lambda}_k^R \\ \boldsymbol{\lambda}_k^I\end{bmatrix}^T = 4(2|\boldsymbol{\phi}_k|^2 - \tilde{\boldsymbol{n}}_k) \circ \begin{bmatrix}\boldsymbol{\phi}_k^R \\ \boldsymbol{\phi}_k^I\end{bmatrix}^T + \begin{bmatrix}\boldsymbol{\lambda}_{k+1}^R \\ \boldsymbol{\lambda}_{k+1}^I\end{bmatrix}^T \boldsymbol{J}_{\boldsymbol{\phi}}\boldsymbol{F}_{\Delta t}(\boldsymbol{\phi}_k^R, \boldsymbol{\phi}_k^I, \boldsymbol{v}_k^C). \quad (10b)$$

Here $\boldsymbol{J}_{\boldsymbol{\phi}}\boldsymbol{F}_{\Delta t}$ denotes the Jacobian of $\boldsymbol{F}$ with respect to $\boldsymbol{\phi}$. We use (10a) as a final condition and iterate (10b) backward in time for $k = K-1, \ldots, 1$. Having computed $\boldsymbol{\Lambda}$ from (10), we return to (9) and compute the gradient with respect to $\boldsymbol{v}_\ell^C$:

$$\nabla_{\boldsymbol{v}_\ell^C}\mathscr{L} = \begin{bmatrix}\boldsymbol{\lambda}_{\ell+1}^R \\ \boldsymbol{\lambda}_{\ell+1}^I\end{bmatrix}^T \nabla_{\boldsymbol{v}_\ell^C}\begin{bmatrix}\boldsymbol{F}_{\Delta t}^R \\ \boldsymbol{F}_{\Delta t}^I\end{bmatrix}(\boldsymbol{\phi}_\ell^R, \boldsymbol{\phi}_\ell^I, \boldsymbol{v}_\ell^C). \quad (11)$$

Given a candidate $\boldsymbol{v}^C$, we solve the forward problem to obtain $\boldsymbol{\Phi}$. We then solve the adjoint system to obtain $\boldsymbol{\Lambda}$. This provides everything required to evaluate (11) for each ℓ. The variations, the block matrix form of the Jacobian $\boldsymbol{J}_{\boldsymbol{\phi}}\boldsymbol{F}_{\Delta t}$, and the gradients of the discrete-time propagator $\boldsymbol{F}$ can be found in Sections 5.3 and 5.4 of the preprint Appendix.

Second Adjoint Method: Learning v^C Functionals. Here we rederive the adjoint method to enable learning the *functional dependence* of $v^C[\phi](x,t)$ on $\phi(x,t)$ and $\phi(x,t-\Delta t)$. We take as our model $v^C[\phi] = v^C(\boldsymbol{\phi}, \boldsymbol{\phi}'; \boldsymbol{\theta})$. The parameters $\boldsymbol{\theta}$ determine a particular functional dependence of v^C on the present and previous Kohn-Sham states $\boldsymbol{\phi}$ and $\boldsymbol{\phi}'$. At spatial grid location x_j and time t_k, the model v^C is

$$v^C[\phi](x_j,t_k) = [\boldsymbol{v}^C(\boldsymbol{\phi}_k, \boldsymbol{\phi}_{k-1}; \boldsymbol{\theta})]_j. \quad (12)$$

In short, we intend $\boldsymbol{\phi}'$ to be the Kohn-Sham state at the time step *prior* to the time step that corresponds to $\boldsymbol{\phi}$. Our goal is to learn $\boldsymbol{\theta}$. This requires redefining the following quantities:

$$\begin{aligned}V(\boldsymbol{\phi}; \boldsymbol{\theta}) &= \mathrm{diag}(\mathbf{v}(\boldsymbol{\phi}, \boldsymbol{\phi}'; \boldsymbol{\theta})) \\ \boldsymbol{v}(\boldsymbol{\phi}, \boldsymbol{\phi}'; \boldsymbol{\theta}) &= -((\boldsymbol{x}+10)^2+1)^{-1/2} + W(|\boldsymbol{\phi}|^2 \circ \boldsymbol{w}) + \boldsymbol{v}^C(\boldsymbol{\phi}, \boldsymbol{\phi}'; \boldsymbol{\theta}) \\ \boldsymbol{F}_{\Delta t}(\boldsymbol{\phi}, \boldsymbol{\phi}'; \boldsymbol{\theta}) &= \mathcal{P}_{\mathcal{K}} \exp(-iV(\boldsymbol{\phi}, \boldsymbol{\phi}'; \boldsymbol{\theta})\Delta t)\mathcal{P}_{\mathcal{K}}\boldsymbol{\phi}.\end{aligned}$$

The Lagrangian still has the form of an objective function together with a dynamical constraint:

$$\mathscr{L}(\boldsymbol{\Phi}^R, \boldsymbol{\Phi}^I, \boldsymbol{\Lambda}^R, \boldsymbol{\Lambda}^I, \boldsymbol{\theta}) = \frac{1}{2}\sum_{k=0}^{K}\sum_{j=0}^{J}(2\phi^R(x_j,t_k)^2 + 2\phi^I(x_j,t_k)^2 - \tilde{n}(x_j,t_k))^2 - \sum_{k=1}^{K-1}[\boldsymbol{\lambda}_{k+1}^R]^T$$
$$(\boldsymbol{\phi}_{k+1}^R - \boldsymbol{F}_{\Delta t}^R(\boldsymbol{\phi}_k^R, \boldsymbol{\phi}_{k-1}^R, \boldsymbol{\phi}_k^I, \boldsymbol{\phi}_{k-1}^I; \boldsymbol{\theta})) + [\boldsymbol{\lambda}_{k+1}^I]^T(\boldsymbol{\phi}_{k+1}^I - \boldsymbol{F}_{\Delta t}^I(\boldsymbol{\phi}_k^R, \boldsymbol{\phi}_{k-1}^R, \boldsymbol{\phi}_k^I, \boldsymbol{\phi}_{k-1}^I; \boldsymbol{\theta})) \quad (13)$$

Setting $\delta\mathscr{L} = 0$ for all variations $\boldsymbol{\phi}_k^R$ and $\boldsymbol{\phi}_k^I$ for $k \geq 1$, we obtain the following adjoint system:

$$\boldsymbol{\lambda}_K = 4\left[(2|\boldsymbol{\phi}_K|^2 - \tilde{\boldsymbol{n}}_K) \circ \boldsymbol{\phi}_K\right] \quad (14a)$$

$$\boldsymbol{\lambda}_{K-1} = 4\left[(2|\boldsymbol{\phi}_{K-1}|^2 - \tilde{\boldsymbol{n}}_{K-1}) \circ \boldsymbol{\phi}_{K-1}\right] + [\boldsymbol{\lambda}_K^R]^T\nabla_{\boldsymbol{\phi}}\boldsymbol{F}_{\Delta t}^R(\boldsymbol{\phi}_K, \boldsymbol{\phi}_{K-1}; \boldsymbol{\theta}) + [\boldsymbol{\lambda}_K^I]^T\nabla_{\boldsymbol{\phi}}\boldsymbol{F}_{\Delta t}^I(\boldsymbol{\phi}_K, \boldsymbol{\phi}_{K-1}; \boldsymbol{\theta}) \quad (14b)$$

$$\begin{bmatrix}\boldsymbol{\lambda}_k^R \\ \boldsymbol{\lambda}_k^I\end{bmatrix}^T = 4(2|\boldsymbol{\phi}_k|^2 - \tilde{\boldsymbol{n}}_k) \circ \begin{bmatrix}\boldsymbol{\phi}_k^R \\ \boldsymbol{\phi}_k^I\end{bmatrix}^T + \begin{bmatrix}\boldsymbol{\lambda}_{k+1}^R \\ \boldsymbol{\lambda}_{k+1}^I\end{bmatrix}^T \boldsymbol{J}_{\boldsymbol{\phi}}\boldsymbol{F}_{\Delta t}(\boldsymbol{\phi}_k, \boldsymbol{\phi}_{k-1}; \boldsymbol{\theta}) + \begin{bmatrix}\boldsymbol{\lambda}_{k+2}^R \\ \boldsymbol{\lambda}_{k+2}^I\end{bmatrix}^T \boldsymbol{J}_{\boldsymbol{\phi}'}\boldsymbol{F}_{\Delta t}(\boldsymbol{\phi}_{k+1}, \boldsymbol{\phi}_k; \boldsymbol{\theta}) \quad (14c)$$

The key difference between (14c) and (10b) is that the right-hand side of (14c) involves $\boldsymbol{\lambda}$ at two points in time. The adjoint system is now a linear delay difference equation with time-dependent coefficients. Additionally, the derivatives of $F_{\Delta t}$ needed to evaluate (14-15) are different—see Section 5.4 of the preprint Appendix. For a candidate value of $\boldsymbol{\theta}$, we solve the forward problem to obtain $\boldsymbol{\phi}$ on our spatial and temporal grid. Then, to compute gradients, we begin with the final conditions (14a-14b) and iterate (14c) backwards in time from $k = K-2$ to $k = 1$. Having solved the adjoint system, we compute the gradient of $\mathscr{L}$ with respect to $\boldsymbol{\theta}$ via

$$\nabla_{\boldsymbol{\theta}}\mathscr{L} = \sum_{k=1}^{K-1}\begin{bmatrix}\boldsymbol{\lambda}_{k+1}^R \\ \boldsymbol{\lambda}_{k+1}^I\end{bmatrix}^T \nabla_{\boldsymbol{\theta}}\begin{bmatrix}\boldsymbol{F}_{\Delta t}^R \\ \boldsymbol{F}_{\Delta t}^I\end{bmatrix}(\boldsymbol{\phi}_k, \boldsymbol{\phi}_{k-1}; \boldsymbol{\theta}). \quad (15)$$

3. Modeling and Implementation Details

Modeling Correlation Functionals. In this work, all models of the form (12) consist of dense, feedforward neural networks. For models of the form $\boldsymbol{v}^C(\boldsymbol{\phi}, \boldsymbol{\phi}'; \boldsymbol{\theta})$, we treat the real and imaginary parts of $\boldsymbol{\phi}$ and $\boldsymbol{\phi}'$ as real vectors each of length $J+1$. Hence for $J = 600$, we have an input layer of size $4(J+1)$. We follow this with three hidden layers each with 256 units and a scaled exponential linear unit activation function (Klambauer et al., 2017). The output layer has $J+1$ units to correspond to the vector-valued output $\boldsymbol{v}^C$. For models in which $\boldsymbol{v}^C$ depend on $\boldsymbol{n}$ and $\boldsymbol{n}'$, we take the real and imaginary parts of $\boldsymbol{\phi}$ and $\boldsymbol{\phi}'$ as inputs and use them to immediately compute $\boldsymbol{n}$ and $\boldsymbol{n}'$, which we then concatenate and feed into an input layer with $2(J+1)$ units. The remainder of the network is as above. We started with smaller networks (fewer layers, less units per layer) and increased the network size until we obtained reasonable training results; no other architecture search or hyperparameter tuning was carried out. We experimented with other activation functions and convolutional layers—none of these models produced satisfactory results during training.

Generation of Training Data. To generate training data, we solve (1) for a model system consisting of $N = 2$ electrons: a one-dimensional electron scattering off a one-dimensional hydrogen

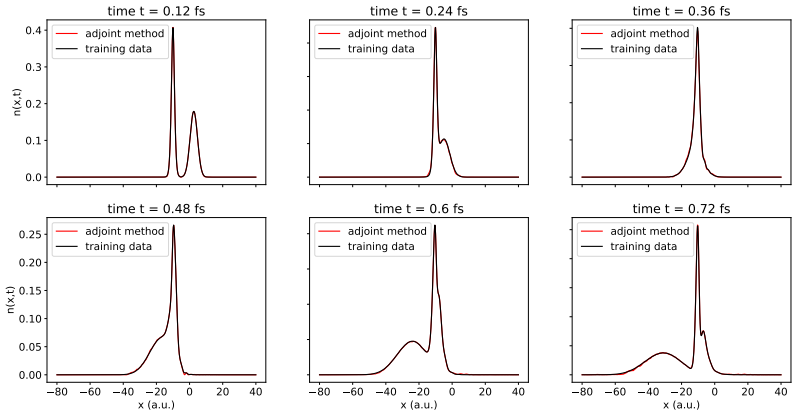

Figure 1: Training results for the problem of learning pointwise values of $\boldsymbol{V}^C$ on a grid consisting of $K = 30000$ points in time and $J = 600$ points in space. The adjoint method succeeds in producing $\boldsymbol{V}^C$ values that yield TDKS solutions such that the corresponding electron densities (red) match those computed from the 2D Schrödinger equation (black).

atom. Hence (1) becomes a partial differential equation (PDE) for a wave function $\Psi(x_1, x_2, t)$. We discretize this PDE using finite differences on an equispaced grid in (x_1, x_2) space with $J = 1201$ points along each axis. Here $-80 \leq x_1, x_2 \leq 40$, so that $\Delta x = 0.1$. After discretizing the kinetic and potential operators in space, we propagate forward in time until $T = 0.72$ fs, using second-order operator splitting with $\Delta t = 2.4 \times 10^{-5}$ fs (or, in a.u., $\Delta t \approx 9.99219 \times 10^{-4}$). Note that this is $1/100$-th the time step used by Suzuki et al. (2020). For further details, consult Section 5.2. After discretization, the wave function $\Psi(x_1, x_2, t)$ at time step k is a complex vector $\boldsymbol{\psi}_k$ of dimension $(J+1)^2$. For the initial vector $\boldsymbol{\psi}_0$, we follow Suzuki et al. (2020) and use a Gaussian wave packet that represents an electron initially centered at $x = 10$ a.u., approaching the H-atom localized at $x = -10$ a.u., with momentum p. We generate training/test data by numerically solving the Schrödinger system for initial conditions with $p \in \{-1.0, -1.2, -1.4, -1.5, -1.6, -1.8\}$. From the resulting time series of wave functions, we compute the time-dependent one-electron density $n(x, t)$; below, *we refer to this as the TDSE electron density*.

4. Results

Pointwise Results. Our first result concerns learning the pointwise values of $\boldsymbol{V}^C$. Here we use the same fine time step $\Delta t = 2.4 \times 10^{-5}$ used to generate the training data. However, we increase Δx by a factor of 2, taking $J = 600$ and sampling the initial condition $\boldsymbol{\phi}_0$ at every other grid point. We retain this subsampling in space in all training sets/results that follow. Still, our unknown $\boldsymbol{V}^C$ consists of a total of $30000 \cdot 601$ values.

We learn $\boldsymbol{V}^C$ by optimizing an objective function that consists of the first line of (9) together with a regularization term. The regularization consists of a finite-difference approximation of $\mu \sum_k \sum_j (\partial_x v^C(x_j, t_k))^2$, with $\mu = 10^{-5}$. This regularization is analogous to the $\int (f'')^2 dx$

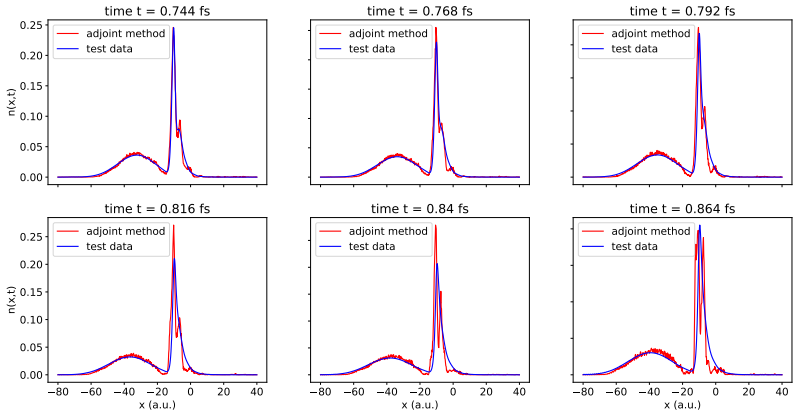

Figure 2: We use 300 time steps (corresponding to 0.72 fs) of the $p = -1.5$ data together with the adjoint method to train a neural network model of v^C that depends on the current and previous ϕ. Using the learned v^C, we propagate (5) for 60 additional time steps and plot the test set results (in red) against the reference electron density (in blue).

penalty used in smoothing splines (Hastie et al., 2009). We penalize the square of the first (rather than second) derivative as we find this is sufficient to smooth v^C in space. The precise value of μ is unimportant; taking $\mu \in [10^{-6}, 10^{-4}]$ yields similar results. For training data, we use only the TDSE one-electron densities computed from the $p = -1.5$ initial condition. To optimize, we use the quasi-Newton L-BFGS-B method, with gradients $\nabla_{\boldsymbol{V}^C}\mathscr{L}$ computed via the procedure described just below (11). We initialize the optimizer with $\boldsymbol{V}^C \equiv 0$ and use default tolerances of 10^{-6}.

In Figure 1, we present the results of this approach. Each panel shows a snapshot of both the training electron density (in black, computed from TDSE data) and the electron density $n = 2|\phi|^2$ (in red) obtained by solving TDKS (5) using the learned $\boldsymbol{V}^C$ values. Note the close quantitative agreement between the black and red curves. The overall mean-squared error (MSE) across all points in space and time is 2.035×10^{-6}. Note that no exact $\boldsymbol{V}^C$ data was used; the learned $\boldsymbol{V}^C$ does not match the exact $\boldsymbol{V}^C$ quantitatively, but does have some of the same qualitative features.

This problem suits the adjoint method well: regardless of the dimensionality of $\boldsymbol{V}^C$, the dimensionality of the adjoint system is the same as that of the discretized TDKS system. Note that, for this one-dimensional TDKS problem (5), it is possible to solve for $\boldsymbol{V}^C$ on a grid (Elliott et al., 2012). If we encounter solutions of *higher-dimensional, multi-electron* ($d \geq 2$ and $N \geq 2$) Schrödinger systems from which we seek to learn $\boldsymbol{V}^C$, we will not be able to employ an exact procedure. In this case, the adjoint-based method may yield numerical values $\boldsymbol{V}^C$, with which we can pursue supervised learning of a functional from electron densities $\boldsymbol{n}$ to correlation potentials $\boldsymbol{V}^C$.

Functional Results. Next we present results in which we learn v^C functionals. In preliminary work, we sought to model $v^C[\phi](x,t)$ as purely a function of $\phi(x,t)$, a model without memory. These models did not yield satisfactory training set results, and hence were abandoned. We focus first on models $v^C[\phi](x,t)$ that allow for arbitrary dependence on the real and imaginary parts of both $\phi(x,t)$ and $\phi(x,t-\Delta t)$. The TDDFT literature emphasizes that v^C should depend on ϕ

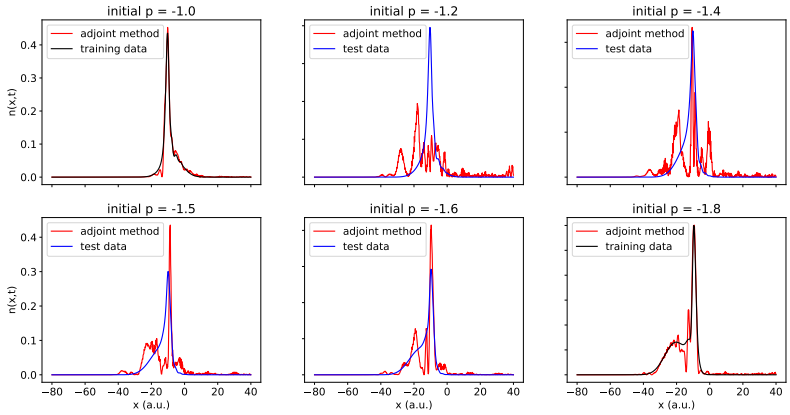

Figure 3: We plot training and test results at time $t = 0.432$ fs for the adjoint method, applied to estimating neural network models $v^C[\phi]$ that, at time t, depend on both $\phi(x,t)$ and $\phi(x,t-\Delta t)$. Propagating TDKS (5) with the learned v^C yields the red curves.

through present/past electron densities n, where $n = 2|\phi|^2$. How important is it to incorporate such physics-based constraints into our v^C model? Let us see how well a direct neural network model of $v^C[\phi]$ captures the dynamics. The input layer is of dimension $4(J+1)$—see Section 3.

To train such a model, we again apply the L-BFGS-B optimizer with objective function given by the first line of (13) and gradients computed with the adjoint system (14-15). We initialize neural network parameters $\boldsymbol{\theta}$ by sampling a mean-zero normal distribution with standard deviation $\sigma = 0.01$. For training data, we subsample the $p = -1.5$ TDSE electron density time series by a factor of 100 in time, so that $\Delta t = 2.4 \times 10^{-3}$ fs and the entire training trajectory consists of $K = 301$ time steps. We retain this time step in all training sets and results that follow.

We omit the training set results here as they show excellent agreement between training and model-predicted electron densities—see Section 5.6. The overall training set mean-squared error (MSE) is 7.668×10^{-6}. In Figure 2, we display test set results obtained by propagating for 60 additional time steps beyond the end of the training data. On this test set, we see close quantitative agreement near $t = 0.72$ fs, which slowly degrades. Still, the learned v^C leads to TDKS electron densities that capture essential features of the reference trajectory. Note that no regularization was used during training of the v^C functional, leading to a learned v^C that is not particularly smooth in space. We hypothesize that, with careful and perhaps physically motivated regularization, the learned v^C will yield improved test set results over longer time intervals.

In the next set of results, we retrain our model using TDSE electron densities with initial momenta equal to $p = -1.0$ and $p = -1.8$. We train two models: a $v^C[\phi](x,t)$ model that depends on ϕ at times t and $t - \Delta t$, and a $v^C[n](x,t)$ model that depends on n at times t and $t - \Delta t$. This latter model incorporates the physics-based constraints mentioned above. We view the $v^C[n]$ model as more constrained because its the first hidden layer can depend on $\phi(x,t)$ and $\phi(x,t-\Delta t)$ *only through* the electron densities $n(x,t)$ and $n(x,t-\Delta t)$. We keep all other details of training the

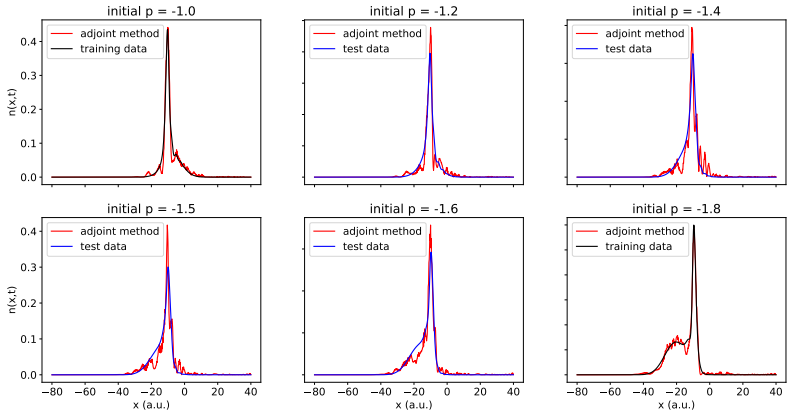

Figure 4: We plot training and test results at time $t = 0.432$ fs for the adjoint method, applied to estimating a neural network model $v^C[n]$ that, at time t, depends on both $n(x,t)$ and $n(x,t-\Delta t)$. Propagating TDKS (5) with the learned $v^C[n]$ yields the red curves.

same. The final training set MSE values are 4.645×10^{-5} for the $v^C[\phi]$ model and 8.098×10^{-5} for the $v^C[n]$ model.

In Figures 3 and 4, we plot both training and test set results for these models. Here we have chosen a particular time ($t = 0.432$ fs) and plotted the electron density at this time for six different trajectories, each with a different initial momentum p. We have chosen this time to highlight the large, obvious differences between the $p = -1.0$ and $p = -1.8$ curves. The $p = -1.0$ and $p = -1.8$ panels contain training set results; here the TDKS electron densities (in red, produced using the learned v^C) lie closer to the ground truth TDSE electron densities (in black).

Note that, despite the greater freedom enjoyed by the $v^C[\phi]$ model, its generalization to trajectories *outside the training set* ($-1.2 \leq p \leq -1.6$) is noticeably worse than that of the more constrained $v^C[n]$ model. In fact, the $v^C[n]$ model's results (Figure 4, in red) show broad qualitative agreement with the test set TDSE curves (in blue). The test set MSE values are 9.363×10^{-4} for the $v^C[\phi]$ model and 2.482×10^{-4} for the $v^C[n]$ model. Overall, these results support the view that v^C should depend on ϕ through n. Again, we hypothesize that if we were to filter out short-wavelength oscillations in the electron density—perhaps by regularizing the $v^C[n]$ model or by training on a larger set of trajectories—the agreement could be improved.

Conclusion. For a low-dimensional model problem, we have developed adjoint-based methods to learn the correlation potential v^C using data from TDSE simulations. The adjoint method can be used to *directly train* $v^C[n]$ models, sidestepping the need for either exact v^C values or density-to-potential inversion. Our work provides a foundation for learning models that depend on present and past snapshots of the electron density. We find that our trained $v^C[n]$ models (with memory) generalize well to trajectories outside the training set. Further improvements to the model may be possible, e.g., by incorporating known physics in the form of model constraints. Overall, the results show the promise of learning v^C via TDKS-constrained optimization.

Acknowledgments

This work was supported by the U.S. Department of Energy, Office of Science, Basic Energy Sciences under Award Number DE-SC0020203. This research used resources of the National Energy Research Scientific Computing Center (NERSC), a U.S. Department of Energy Office of Science User Facility located at Lawrence Berkeley National Laboratory, operated under Contract No. DE-AC02-05CH11231 using NERSC award BES-m2530 for 2021. We acknowledge computational time on the Pinnacles cluster at UC Merced (supported by NSF OAC-2019144). We also acknowledge computational time on the Nautilus cluster, supported by the Pacific Research Platform (NSF ACI-1541349), CHASE-CI (NSF CNS-1730158), and Towards a National Research Platform (NSF OAC-1826967). Additional funding for Nautilus has been supplied by the University of California Office of the President.

References

Alfio Borzì. Quantum optimal control using the adjoint method. *Nanoscale Systems: Mathematical Modeling, Theory and Applications*, 1:93–111, 2012. URL http://eudml.org/doc/266625.

A. E. Bryson and Y.-C. Ho. *Applied Optimal Control: Optimization, Estimation and Control*. Halsted Press Book. Taylor & Francis, 1975. Revised printing.

A. Castro, J. Werschnik, and E. K. U. Gross. Controlling the dynamics of many-electron systems from first principles: A combination of optimal control and time-dependent density-functional theory. *Phys. Rev. Lett.*, 109:153603, Oct 2012. doi: 10.1103/PhysRevLett.109.153603. URL https://link.aps.org/doi/10.1103/PhysRevLett.109.153603.

Alberto Castro and E. K. U. Gross. Optimal control theory for quantum-classical systems: Ehrenfest molecular dynamics based on time-dependent density-functional theory. *Journal of Physics A: Mathematical and Theoretical*, 47(2):025204, 2013.

Alberto Castro, Miguel A. L. Marques, and Angel Rubio. Propagators for the time-dependent Kohn-Sham equations. *The Journal of Chemical Physics*, 121(8):3425–3433, 2004.

Peter Elliott, Johanna I Fuks, Angel Rubio, and Neepa T Maitra. Universal dynamical steps in the exact time-dependent exchange-correlation potential. *Physical Review Letters*, 109(26):266404, 2012.

M. D. Feit, J. A. Fleck Jr, and A. Steiger. Solution of the Schrödinger equation by a spectral method. *Journal of Computational Physics*, 47(3):412–433, 1982.

J. A. Fleck Jr, J. R. Morris, and M. D. Feit. Time-dependent propagation of high energy laser beams through the atmosphere. *Applied Physics*, 10(2):129–160, 1976.

L. Hasdorff. *Gradient Optimization and Nonlinear Control*. Wiley, 1976.

Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*. Springer, second edition, 2009.

Bhupalee Kalita, Li Li, Ryan J. McCarty, and Kieron Burke. Learning to approximate density functionals. *Accounts of Chemical Research*, 54(4):818–826, 2021. doi: 10.1021/acs.accounts.0c00742. URL https://doi.org/10.1021/acs.accounts.0c00742.

Günter Klambauer, Thomas Unterthiner, Andreas Mayr, and Sepp Hochreiter. Self-Normalizing Neural Networks. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, NIPS'17, pages 972–981, 2017.

Neepa T. Maitra. Perspective: Fundamental aspects of time-dependent density functional theory. *The Journal of Chemical Physics*, 144(22):220901, 2016. doi: 10.1063/1.4953039. URL https://doi.org/10.1063/1.4953039.

Neepa T. Maitra, Kieron Burke, and Chris Woodward. Memory in time-dependent density functional theory. *Phys. Rev. Lett.*, 89:023002, Jun 2002. doi: 10.1103/PhysRevLett.89.023002. URL https://link.aps.org/doi/10.1103/PhysRevLett.89.023002.

Ryo Nagai, Ryosuke Akashi, Shu Sasaki, and Shinji Tsuneyuki. Neural-network Kohn-Sham exchange-correlation potential and its out-of-training transferability. *The Journal of Chemical Physics*, 148(24):241737, 2018.

Erich Runge and E. K. U. Gross. Density-functional theory for time-dependent systems. *Phys. Rev. Lett.*, 52:997–1000, Mar 1984. doi: 10.1103/PhysRevLett.52.997. URL https://link.aps.org/doi/10.1103/PhysRevLett.52.997.

J. M. Sanz-Serna. Symplectic Runge–Kutta schemes for adjoint equations, automatic differentiation, optimal control, and more. *SIAM Review*, 58(1):3–33, 2016. doi: 10.1137/151002769.

Martin Sprengel, Gabriele Ciaramella, and Alfio Borzì. A Theoretical Investigation of Time-Dependent Kohn–Sham Equations. *SIAM Journal on Mathematical Analysis*, 49(3):1681–1704, 2017.

Martin Sprengel, Gabriele Ciaramella, and Alfio Borzì. Investigation of optimal control problems governed by a time-dependent Kohn-Sham model. *Journal of Dynamical and Control Systems*, 24(4):657–679, 2018. https://arxiv.org/abs/1701.02679.

Yasumitsu Suzuki, Ryo Nagai, and Jun Haruyama. Machine learning exchange-correlation potential in time-dependent density-functional theory. *Physical Review A*, 101(5):050501, 2020.

Carsten A. Ullrich. *Time-Dependent Density-Functional Theory: Concepts and Applications*. Oxford Graduate Texts. Oxford University Press, Oxford, 2011. doi: 10.1093/acprof:oso/9780199563029.001.0001.

Robert van Leeuwen. Mapping from densities to potentials in time-dependent density-functional theory. *Phys. Rev. Lett.*, 82:3863–3866, May 1999. doi: 10.1103/PhysRevLett.82.3863. URL https://link.aps.org/doi/10.1103/PhysRevLett.82.3863.

Lucas O Wagner, Zeng-hui Yang, and Kieron Burke. Exact conditions and their relevance in TDDFT. In *Fundamentals of Time-Dependent Density Functional Theory*, pages 101–123. Springer, 2012.