Kohn-Sham Equations as Regularizer: Building Prior Knowledge into Machine-Learned Physics

Li Li (李力),[1,*] Stephan Hoyer,[1] Ryan Pederson,[2] Ruoxi Sun (孙若溪),[1] Ekin D. Cubuk,[1] Patrick Riley,[1] and Kieron Burke[2,3]

[1]*Google Research, Mountain View, California 94043, USA*
[2]*Department of Physics and Astronomy, University of California, Irvine, California 92697, USA*
[3]*Department of Chemistry, University of California, Irvine, California 92697, USA*

(Received 18 September 2020; accepted 3 December 2020; published 20 January 2021)

Including prior knowledge is important for effective machine learning models in physics and is usually achieved by explicitly adding loss terms or constraints on model architectures. Prior knowledge embedded in the physics computation itself rarely draws attention. We show that solving the Kohn-Sham equations when training neural networks for the exchange-correlation functional provides an implicit regularization that greatly improves generalization. Two separations suffice for learning the entire one-dimensional H_2 dissociation curve within chemical accuracy, including the strongly correlated region. Our models also generalize to unseen types of molecules and overcome self-interaction error.

DOI: 10.1103/PhysRevLett.126.036401

Differentiable programming [1] is a general paradigm of deep learning, where parameters in the computation flow are trained by gradient-based optimization. Based on the enormous development in automatic differentiation libraries [2–5], hardware accelerators [6], and deep learning [7], this emerging paradigm is relevant for scientific computing. It supports extremely strong physics prior knowledge and well-established numerical methods [8] and parametrizes the approximation by a neural network, which can approximate any continuous function [9]. Recent highlights include discretizing partial differential equations [10], structural optimization [11], sampling equilibrium configurations [12], differentiable molecular dynamics [13], differentiable programming tensor networks [14], optimizing basis sets in Hartree-Fock [15] method, and variational quantum Monte Carlo [16–19] calculations.

Density functional theory (DFT), an approach to electronic structure problems, took an enormous step forward with the creation of the Kohn-Sham (KS) equations [20], which greatly improve accuracy from the original DFT [21–23]. The results of solving the KS equations are reported in tens of thousands of papers each year [24]. Given an approximation to the exchange-correlation (XC) energy, the KS equations are solved self-consistently. Results are limited by the quality of such approximations, and a standard problem of KS-DFT is to calculate accurate bond dissociation curves [25]. The difficulties are an example of strong correlation physics as electrons localize on separate nuclei [26].

Naturally, there has been considerable interest in using machine learning (ML) methods to improve DFT approximations. Initial work [27,28] focused on the KS kinetic energy, as a sufficiently accurate approximation would allow bypassing the solving of the KS equations [29,30]. For XC, recent works focus on learning the XC potential (not functional) from inverse KS [31] and use it in the KS-DFT scheme [32–35]. An important step forward was made last year, when it was shown that a neural network could find functionals using only three molecules by training on both energies and densities [36], obtaining accuracy comparable to human-designed functionals and generalizing to yield accurate atomization energies of 148 small molecules [37]. But this pioneering work does not yield chemical accuracy or approximations that work in the dissociation limit. Moreover, it uses gradient-free optimization which usually suffers from poor convergence behavior on the large number of parameters used in modern neural networks [38–40].

Here, we show that all these limitations are overcome by incorporating the KS equations themselves into the neural network training by backpropagating through their iterations—a *KS regularizer* (KSR) to the ML model. In a traditional KS calculation, the XC is given, the equations are cycled to self-consistency, and all previous iterations are ignored in the final answer. In other ML work, functionals are trained on either energies alone [41–44], or even densities [33,34,45], but only after convergence. By incorporating the KS equations into the training, thereby learning the relation between density and energy at every

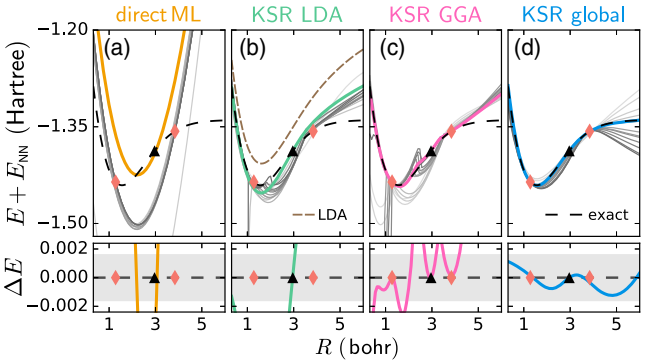

FIG. 1. One-dimensional H_2 dissociation curves for several ML models trained from two molecules (red diamonds) with optimal models (highlighted in color) selected by the validation molecule at $R = 3$ (black triangles). The top panel shows energy (with E_{NN}, the nucleus-nucleus repulsion energy) with exact values shown by the black dashed line. The bottom panel shows the difference from the exact curves with chemical accuracy in gray shadow. (a) directly predicts E from geometries and clearly fails to capture the physics from very limited data. (b)–(d) show our method (KSR) with different inputs to the model to align with the first two rungs of Jacob's ladder [47] (LDA and GGA) and then global (a fully nonlocal functional). Uniform gas LDA [46] is shown in brown. Gray lines denote 15 sampled functionals during training, with darker lines denoting later samples. Atomic units used throughout.

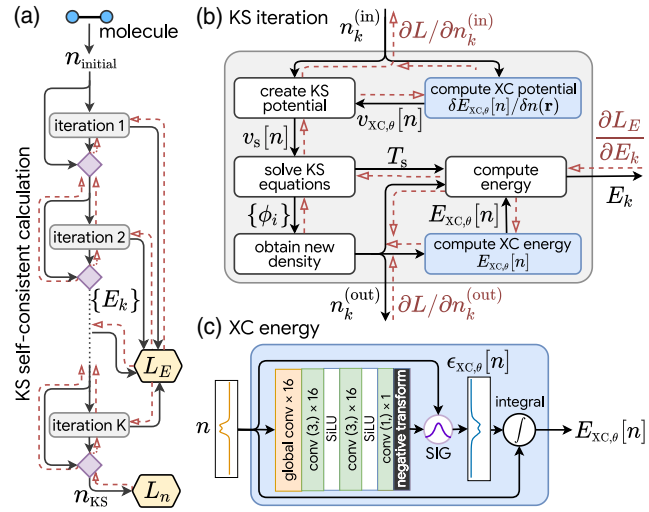

FIG. 2. KS-DFT as a differentiable program. Black arrows are the conventional computation flow. The gradients flow along red dashed arrows to minimize the energy loss L_E and density loss L_n. (a) The high-level KS self-consistent calculations with linear density mixing (purple diamonds). (b) A single KS iteration produces $v_{\text{XC},\theta}[n]$ and $E_{\text{XC},\theta}[n]$ by invoking the XC energy calculation twice, once directly and once calculating a derivative using automatic differentiation. (c) The XC energy calculation using the global XC functional.

iteration, we find accurate models with very little data and much greater generalizability.

Our results are illustrated in Fig. 1, which is for a one-dimensional mimic of H_2 designed for testing electronic structure methods [46]. The distribution of curves of the ML model directly predicting E from geometries (direct ML) in Fig. 1(a) clearly fails to capture the physics. Next, we demonstrate KSR with neural XC functionals from the first two rungs of Jacob's ladder [47] by constraining the receptive field of the convolutional neural network [48]. The local density approximation (LDA) has a receptive field of just the current point, while the generalized gradient approximation (GGA) includes the nearest-neighbor points, the minimal information for computing the spatial gradient of the density. In Figs. 1(b) and 1(c), the effect of the KSR yields reasonably accurate results in the vicinity of the data, but not beyond. The KSR LDA behaves similarly to the uniform gas LDA [46]. When an XC functional with a global receptive field is included in Fig. 1(d), chemical accuracy is achieved for all separations including the dissociation limit. Similar results can be achieved for H_4, the one-electron self-interaction error can easily be made to vanish, and the interaction of a pair of H_2 molecules can be found without any training on this type of molecule (discussed below).

Modern DFT finds the ground-state electronic density by solving the Kohn-Sham equations:

$$\left\{-\frac{\nabla^2}{2} + v_s[n](\mathbf{r})\right\}\phi_i(\mathbf{r}) = \epsilon_i \phi_i(\mathbf{r}). \tag{1}$$

The density is obtained from occupied orbitals $n(\mathbf{r}) = \sum_i |\phi_i(\mathbf{r})|^2$. Here, $v_s[n](\mathbf{r}) = v(\mathbf{r}) + v_H[n](\mathbf{r}) + v_{\text{XC}}[n](\mathbf{r})$ is the KS potential consisting of the external one-body potential and the density-dependent Hartree (H) and XC potentials. The XC potential $v_{\text{XC}}[n](\mathbf{r}) = \delta E_{\text{XC}}/\delta n(\mathbf{r})$ is the functional derivative of the XC energy functional $E_{\text{XC}}[n] = \int \epsilon_{\text{XC}}[n](\mathbf{r}) n(\mathbf{r}) d\mathbf{r}$, where $\epsilon_{\text{XC}}[n](\mathbf{r})$ is the XC energy per electron. The total electronic energy E is then given by the sum of the noninteracting kinetic energy $T_s[n]$, the external one-body potential energy $V[n]$, the Hartree energy $U[n]$, and XC energy $E_{\text{XC}}[n]$.

The KS equations are, in principle, exact given the exact XC functional [20,54], which in practice is the only term approximated in DFT. From a computational perspective, the eigenvalue problem of Eq. (1) is solved repeatedly until the density converges to a fixed point starting from an initial guess. We use linear density mixing [55] to improve convergence, $n_{k+1}^{(\text{in})} = n_k^{(\text{in})} + \alpha(n_k^{(\text{out})} - n_k^{(\text{in})})$. Figure 2(a) shows the unrolled computation flow. We approximate the XC energy per electron using a neural network $\epsilon_{\text{XC},\theta}[n]$, where θ represents the trainable parameters. Together with the self-consistent iterations in Fig. 2(b), the combined computational graph resembles a recurrent neural network [56] or deep equilibrium model [57] with additional fixed computational components. Density mixing improves

convergence of KS self-consistent calculations and parallels the now common residual connections in deep neural networks [58] for efficient backpropagation.

If the neural XC functional were exact, KS self-consistent calculations would output the exact density, and the intermediate energies over iterations would converge to the exact energy. This intention can be translated into a loss function, and the neural XC functional can be updated end to end by backpropagating through the KS self-consistent calculations. Throughout, experiments are performed in one dimension where accurate quantum solutions could be relatively easily generated via the density matrix renormalization group (DMRG) [59]. The electron-electron repulsion is $A\exp(-\kappa|x-x'|)$, and attraction to a nucleus at $x=0$ is $-A\exp(-\kappa|x|)$ [48]. We design the loss function as an expectation $\mathbb{E}$ over training molecules,

$$L(\theta) = \underbrace{\mathbb{E}_{\text{train}}\left[\int dx (n_{\text{KS}} - n_{\text{DMRG}})^2/N_e\right]}_{\text{density loss } L_n} + \underbrace{\mathbb{E}_{\text{train}}\left[\sum_{k=1}^{K} w_k (E_k - E_{\text{DMRG}})^2/N_e\right]}_{\text{energy loss } L_E}, \quad (2)$$

where N_e is the number of electrons and w_k are nonnegative weights. L_n minimizes the difference between the final density with the exact density. The gradient from L_n backpropagates through $v_{\text{XC},\theta}[n]$ in all KS iterations. However, if L_E only optimizes the final energy, no gradient flows through $E_{\text{XC},\theta}[n]$ except for the final iteration. To make backpropagation more efficient for $E_{\text{XC},\theta}[n]$, L_E optimizes the trajectory of energies over all iterations, which directly flows gradients to early iterations [60]. This makes the neural XC functional output accurate ϵ_{XC} at each iteration and also drives the iterations to quickly converge to the exact energy. The optimal model is selected with minimal mean absolute energy per electron on the validation set.

Hundreds of useful XC functional approximations have been proposed [61]. Researchers typically design the symbolic form from physics intuition, with some (or no) fitting parameters. Here we build a neural XC functional with several differentiable components with physics intuition tailored for XC in Fig. 2(c). A global convolution layer captures the long-range interaction, $G(n(x),\xi_p) = (1/2\xi_p)\int dx' n(x')\exp(-|x-x'|/\xi_p)$. Note two special cases retrieve known physics quantities, Hartree energy density $G(n(x),\kappa^{-1}) \propto \epsilon_H$, and electronic density $G(n(x),0) = n(x)$. Global convolution contains multiple channels, and ξ_p of each channel is trainable to capture interaction in different scales. Although the rectified linear unit [62] is popular, we use the sigmoid linear unit (SiLU) [63,64] $f(x) = x/[1+\exp(-x)]$ because the infinite differentiability of SiLU guarantees the smoothness of v_{XC}, the first derivative, and the second and higher order derivatives of the neural network used in the Limited-memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS) training [49]. We do not enforce a specific choice of ϵ_{XC} (sometimes called a gauge [65]), but we do enforce some conditions, primarily to aid convergence of the algorithm. We require ϵ_{XC} to vanish whenever the density does and that it be negative if at all possible. We achieved the former using the linearity of SiLU near the origin and turning off the bias terms in convolution layers. We softly impose the latter by a negative transform layer at the end, where a negative SiLU makes most output values negative. Finally, we design a self-interaction gate (SIG) that mixes in a portion of $-\epsilon_H$ to cancel the self-interaction error, $\epsilon_{\text{XC}}^{(\text{out})} = \epsilon_{\text{XC}}^{(\text{in})}(1-\beta) - \epsilon_H\beta$. The portion is a gate function $\beta(N_e) = \exp[-(N_e-1)^2/\sigma^2]$. When $N_e = 1$, then $\epsilon_{\text{XC}}^{(\text{out})} = -\epsilon_H$. For more electrons, σ can be fixed or adjusted by the training algorithm to decide the sensitivity to N_e. For H_2 as $R \to \infty$, ϵ_{XC} tends to a superposition of the negative of the Hartree energy density at each nucleus and approaches half that for H_2^+.

Now we dive deeper into the outstanding generalization we observed in a simple but not easy task: predicting the entire H_2 dissociation curve, as shown in Fig. 1. It is not surprising that the direct ML model completely fails. Neural networks are usually underdetermined systems as there are more parameters than training examples. Regularization is crucial to improve generalization [66,67], especially when data are limited. Most existing works regularize models with particular physics prior knowledge by imposing *constraints* via feature engineering and preprocessing [68,69], architecture design [70–73], or physics-informed loss terms [74–76]. Another strategy is to generate extra data for training using prior knowledge: In image classification problems, data are augmented by operations like flipping and cropping given the prior knowledge that labels are invariant to those operations [77]. KSR provides a natural data augmentation because although the exact densities and energies of only two separations are given, KSR samples different trajectories from an initial density to the exact density at each training step. More importantly, KSR focuses on learning an XC functional that can lead the KS self-consistent calculations to converge to the exact density from the initial density. Figure 3 visualizes the density trajectories sampled by KSR for one training separation $R = 3.84$. The functional with untrained parameters ($t = 0$) samples densities near the initial guess but soon learns to explore broadly and finds the trajectories toward the vicinity of the exact density.

In contrast, most existing ML functionals learn to predict the output of a single iteration from the exact density, which is a poor surrogate for the full self-consistent calculations [79]. These standard ML models have two

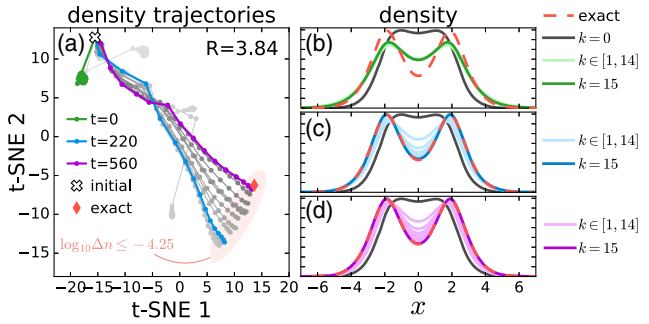

FIG. 3. (a) t-distributed stochastic neighbor embedding (t-SNE) visualization [78] of density trajectories (gray dots) sampled by KSR during training for $R = 3.84$ from initial guess (cross) to exact density (red diamond). Darker trajectories denote later optimization steps t. Note t-SNE projection does not perfectly preserve the distance between densities. The light red ellipse illustrates the vicinity of the exact density within $\log_{10}[\int dx(n_{\mathrm{KS}} - n_{\mathrm{DMRG}})^2/N_e] \leq -4.25$. Densities from each KS iteration in trajectories are plotted in the corresponding highlighted colors for (b) $t = 0$ untrained, (c) $t = 220$ optimal in Fig. 1, and (d) $t = 560$ overfitting to training with bad generalization on validation.

major shortcomings. First, the exact density is unknown for new systems, so the model is not expected to behave correctly on unseen initial densities for KS calculations. Second, even if a model is trained on many densities for single iteration prediction, it is not guaranteed to converge the self-consistent calculations to a good solution [80]. On the other hand, since KSR allows the model access to all the KS iterations, it learns to optimize the entire self-consistent procedure to avoid the error accumulation from greedy optimization of single iterations. Further comparison for training without or with "weaker" KSR is in the Supplemental Material [48].

Next, we retrain our neural XC functional with KSR on $N_{\mathrm{train}}/2$ examples each of H_2 and H_4 molecules. Figure 4 shows the prediction accuracy of KSR with both energy and density loss (full KSR), in comparison to KSR with only energy loss (energy-only KSR) and the direct ML model. We compute the energy mean absolute error on the holdout sets of H_2 ($R \in [0.4, 6]$) and H_4 ($R \in [1.04, 6]$). The average mean absolute error of H_2 and H_4 with various N_{train} is shown in Fig. 4(a). Full KSR has the lowest error at minimum $N_{\mathrm{train}} = 4$, reaching chemical accuracy at 6. As the size of the training set increases, energy-only KSR reaches chemical accuracy at $N_{\mathrm{train}} = 10$, but the direct ML model never does (even at 20). Then we test models on unseen types of molecules. In Fig. 4(b), both KSR models have perfect prediction on H_2^+ ($R \in [0.64, 8.48]$) because of the SIG in the neural XC functionals, while direct ML models always have large errors. Finally, we take a pair of equilibrium H_2 and separate them with $R = 0.16$ to 9.76 bohr denoted as H_2H_2. KSR models generalize much better than ML for "zero-shot" prediction [81], where H_2H_2 has never been exposed to the model during training.

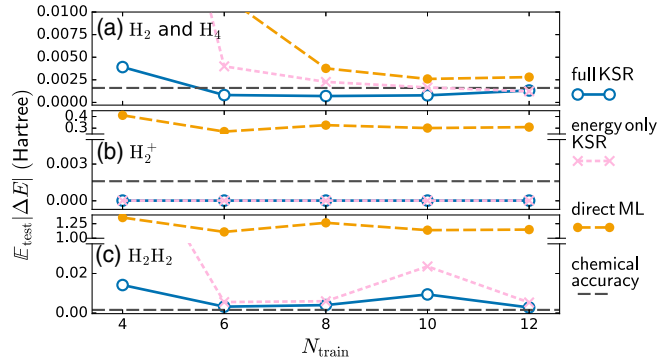

FIG. 4. Test generalization of models as a function of the total number of training examples N_{train}: full KSR (blue), energy only KSR (pink), and direct ML (orange) on (a) holdout H_2 and H_4, and unseen types of molecules (b) H_2^+, (c) H_2H_2. Black dashed lines show chemical accuracy. See the Supplemental Material [48] for training details.

Why is the density important in training, and what use is the nonconverged iterations? The density is the functional derivative of the energy with respect to the potential, so it gives the exact slope of the energy with respect to any change in the potential, including stretching (or compressing) the bond. Thus, the density implicitly contains energetic information including the correct derivative at that point in the binding curve. KS iterations produce information about the functional in the vicinity of the minimum. During training, the network learns to construct a functional with both the correct minimum and all correct derivatives at this minimum. In the paradigm of differentiable programming, density is the hidden state carrying the information through the recurrent structure in Fig. 2(a). Correct supervision from L_n greatly helps generalization from very limited data; see $N_{\mathrm{train}} \leq 6$ in Fig. 4. But as N_{train} increases, both KSRs with and without L_n perform well in energy prediction. We show the solution of H_4 with $R = 4.32$

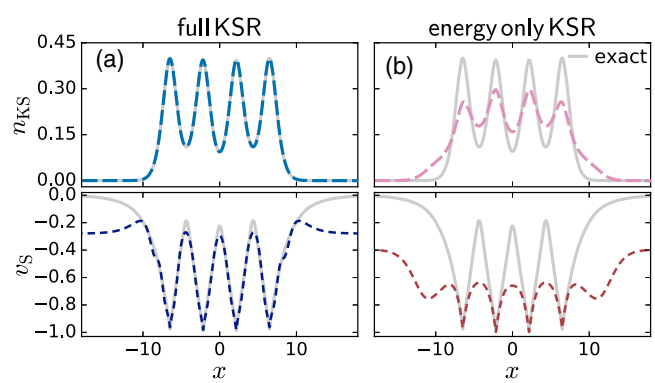

FIG. 5. Density and KS potential of H_4 with $R = 4.32$ from neural XC functionals trained with (a) full KSR (blue) and (b) energy only KSR (pink) on training set of size $N_{\mathrm{train}} = 20$. Exact curves are in gray. v_s are shifted by a constant for better comparison.

in Fig. 5. With L_n, the density is clearly much more accurate than KSR without L_n [$\int(n_{\mathrm{KS}} - n_{\mathrm{DMRG}})^2 dx = 9.2 \times 10^{-5}$ versus 9.8×10^{-2}]. Then we compute the corresponding exact v_s using the inverse KS method [31]. Both functionals do not reproduce the exact v_s. However, the functional trained with L_n recovered most of the KS potential. Unlike previous works [33–35] that explicitly included the KS or XC potential in the loss function, our model never uses the exact KS potential. In our KSR setup, the model aims to predict ϵ_{XC}, from which the derived v_s yields accurate density. Therefore, predicting v_{XC} is a side product. We also address some concerns on training explicitly with v_{XC}. One artifact is that generating the exact v_s requires an additional inverse calculation, which is known to be numerically unstable [31]. Schmidt *et al.* [33] observe outliers while generating training v_{XC} from inverse KS. While v_{XC} is a fascinating and useful object for theoretical study because its relation to the density is extremely delicate, it is far more practical to simply use the density to train on [36].

Differentiable programming blurs the boundary between physics computation and ML. Our results for KS-DFT serve as proof of principle for rethinking computational physics in this new paradigm. Although there is no explicit limitation of our algorithm to one dimension, we expect practical challenges with real molecules, which will require rewriting or extending a mature DFT code to support automatic differentiation. For example, our differentiable eigensolver for dense matrices [82] is not suitable for large problems and will need to be replaced with methods for partial eigendecomposition of sparse matrices [83,84]. Beyond density functionals in principle, all heuristics in DFT calculations, e.g., initial guess, density update, preconditioning, basis sets, even the entire self-consistent calculations as a meta-optimization problem [60], could be learned and optimized while maintaining rigorous physics—getting the best of both worlds.

The authors thank Michael Brenner, Sam Schoenholz, Lucas Wagner, and Hanjun Dai for helpful discussion. K. B. is supported by National Science Foundation Grant No. CHE-1856165 and R. P. by Department of Energy Grant No. DE-SC0008696. Code is available at [85].

*leeley@google.com

[1] A. G. Baydin, B. A. Pearlmutter, A. A. Radul, and J. M. Siskind, Automatic differentiation in machine learning: A survey, J. Mach. Learn. Res. **18**, 5595 (2017).

[2] J. Bradbury, R. Frostig, P. Hawkins, M. J. Johnson, C. Leary, D. Maclaurin, and S. Wanderman-Milne, JAX: Composable transformations of PYTHON+NumPy programs, http://github.com/google/jax, 2018.

[3] M. Innes, E. Saba, K. Fischer, D. Gandhi, M. C. Rudilosso, N. M. Joy, T. Karmali, A. Pal, and V. Shah, Fashionable modelling with flux, arXiv:1811.01457.

[4] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga *et al.*, PyTorch: An Imperative Style, High-Performance Deep Learning Library, in *Advances in Neural Information Processing Systems* (MIT Press, Cambridge, 2019), pp. 8026–8037.

[5] M. Abadi *et al.*, TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems, arXiv:1603.04467.

[6] N. P. Jouppi, C. Young, N. Patil, D. Patterson, G. Agrawal, R. Bajwa, S. Bates, S. Bhatia, N. Boden, A. Borchers *et al.*, In-datacenter performance analysis of a tensor processing unit, in *Proceedings of the 44th Annual International Symposium on Computer Architecture* (Association for Computing Machinery, New York, 2017), pp. 1–12.

[7] Y. LeCun, Y. Bengio, and G. Hinton, Deep learning, Nature (London) **521**, 436 (2015).

[8] M. Innes, A. Edelman, K. Fischer, C. Rackauckas, E. Saba, V. B. Shah, and W. Tebbutt, A differentiable programming system to bridge machine learning and scientific computing, arXiv:1907.07587.

[9] K. Hornik, Approximation capabilities of multilayer feedforward networks, Neural Netw. **4**, 251 (1991).

[10] Y. Bar-Sinai, S. Hoyer, J. Hickey, and M. P. Brenner, Learning data-driven discretizations for partial differential equations, Proc. Natl. Acad. Sci. U.S.A. **116**, 15344 (2019).

[11] S. Hoyer, J. Sohl-Dickstein, and S. Greydanus,Neural reparameterization improves structural optimization, arXiv:1909.04240.

[12] F. Noé, S. Olsson, J. Köhler, and H. Wu, Boltzmann generators: Sampling equilibrium states of many-body systems with deep learning, Science **365**, eaaw1147 (2019).

[13] S. S. Schoenholz and E. D. Cubuk, JAX, M.D.: A framework for differentiable physics, arXiv:1912.04232.

[14] H.-J. Liao, J.-G. Liu, L. Wang, and T. Xiang, Differentiable Programming Tensor Networks, Phys. Rev. X **9**, 031041 (2019).

[15] T. Tamayo-Mendoza, C. Kreisbeck, R. Lindh, and A. Aspuru-Guzik, Automatic differentiation in quantum chemistry with applications to fully variational Hartree–Fock, ACS Cent. Sci. **4**, 559 (2018).

[16] L. Yang, W. Hu, and L. Li, Scalable variational Monte Carlo with graph neural ansatz, arXiv:2011.12453.

[17] J. Hermann, Z. Schätzle, and F. Noé, Deep-neural-network solution of the electronic Schrödinger equation, Nat. Chem. **12**, 891 (2020).

[18] D. Pfau, J. S. Spencer, A. G. D. G. Matthews, and W. M. C. Foulkes, *Ab initio* solution of the many-electron Schrödinger equation with deep neural networks, Phys. Rev. Research **2**, 033429 (2020).

[19] L. Yang, Z. Leng, G. Yu, A. Patel, W.-J. Hu, and H. Pu, Deep learning-enhanced variational Monte Carlo method for quantum many-body physics, Phys. Rev. Research **2**, 012039 (2020).

[20] W. Kohn and L. J. Sham, Self-consistent equations including exchange and correlation effects, Phys. Rev. **140**, A1133 (1965).

[21] P. Hohenberg and W. Kohn, Inhomogeneous electron gas, Phys. Rev. **136**, B864 (1964).

[22] L. H. Thomas, The calculation of atomic fields, Math. Proc. Cambridge Philos. Soc. **23**, 542 (1927).

[23] E. Fermi, Un metodo statistico per la determinazione di alcune priorieta dell'atome, Rend. Accad. Naz. Lincei **6**, 5 (1927).

[24] R. O. Jones, Density functional theory: Its origins, rise to prominence, and future, Rev. Mod. Phys. **87**, 897 (2015).

[25] E. M. Stoudenmire, L. O. Wagner, S. R. White, and K. Burke, One-Dimensional Continuum Electronic Structure with the Density-Matrix Renormalization Group and Its Implications for Density-Functional Theory, Phys. Rev. Lett. **109**, 056402 (2012).

[26] A. J. Cohen, P. Mori-Sánchez, and W. Yang, Insights into current limitations of density functional theory, Science **321**, 792 (2008).

[27] L. Li, J. C. Snyder, I. M. Pelaschier, J. Huang, U.-N. Niranjan, P. Duncan, M. Rupp, K.-R. Müller, and K. Burke, Understanding machine-learned density functionals, Int. J. Quantum Chem. **116**, 819 (2016).

[28] J. C. Snyder, M. Rupp, K. Hansen, K.-R. Müller, and K. Burke, Finding Density Functionals with Machine Learning, Phys. Rev. Lett. **108**, 253002 (2012).

[29] F. Brockherde, L. Vogt, L. Li, M. E. Tuckerman, K. Burke, and K.-R. Müller, Bypassing the Kohn-Sham equations with machine learning, Nat. Commun. **8**, 872 (2017).

[30] L. Li, T. E. Baker, S. R. White, and K. Burke, Pure density functional for strong correlation and the thermodynamic limit from machine learning, Phys. Rev. B **94**, 245129 (2016).

[31] D. S. Jensen and A. Wasserman, Numerical methods for the inverse problem of density functional theory, Int. J. Quantum Chem. **118**, e25425 (2018).

[32] D. J. Tozer, V. E. Ingamells, and N. C. Handy, Exchange-correlation potentials, J. Chem. Phys. **105**, 9200 (1996).

[33] J. Schmidt, C. L. Benavides-Riveros, and M. A. Marques, Machine learning the physical nonlocal exchange–correlation functional of density-functional theory, J. Phys. Chem. Lett. **10**, 6425 (2019).

[34] Y. Zhou, J. Wu, S. Chen, and G. Chen, Toward the exact exchange–correlation potential: A three-dimensional convolutional neural network construct, J. Phys. Chem. Lett. **10**, 7264 (2019).

[35] R. Nagai, R. Akashi, S. Sasaki, and S. Tsuneyuki, Neural-network Kohn-Sham exchange-correlation potential and its out-of-training transferability, J. Chem. Phys. **148**, 241737 (2018).

[36] R. Nagai, R. Akashi, and O. Sugino, Completing density functional theory by machine learning hidden messages from molecules, npj Comput. Mater. **6**, 43 (2020).

[37] L. A. Curtiss, K. Raghavachari, G. W. Trucks, and J. A. Pople, Gaussian-2 theory for molecular energies of first- and second-row compounds, J. Chem. Phys. **94**, 7221 (1991).

[38] J. C. Duchi, M. I. Jordan, M. J. Wainwright, and A. Wibisono, Optimal rates for zero-order convex optimization: The power of two function evaluations, IEEE Trans. Inf. Theory **61**, 2788 (2015).

[39] N. Maheswaranathan, L. Metz, G. Tucker, D. Choi, and J. Sohl-Dickstein, Guided evolutionary strategies: augmenting random search with surrogate gradients, in *Proceedings of the International Conference on Machine Learning* (Proceedings of Machine Learning Research, 2019), pp. 4264–4273.

[40] L. M. Rios and N. V. Sahinidis, Derivative-free optimization: a review of algorithms and comparison of software implementations, J. Global Optim. **56**, 1247 (2013).

[41] J. Gilmer, S. S. Schoenholz, P. F. Riley, O. Vinyals, and G. E. Dahl, Neural message passing for quantum chemistry, in *Proceedings of the 34th International Conference on Machine Learning* (2017), 1263–1272.

[42] K. Schütt, P.-J. Kindermans, H. E. S. Felix, S. Chmiela, A. Tkatchenko, and K.-R. Müller, SchNet: A continuous-filter convolutional neural network for modeling quantum interactions, in *Advances in Neural Information Processing Systems* (MIT Press, Cambridge, 2017), pp. 991–1001.

[43] J. Behler and M. Parrinello, Generalized Neural-Network Representation of High-Dimensional Potential-Energy Surfaces, Phys. Rev. Lett. **98**, 146401 (2007).

[44] M. Rupp, A. Tkatchenko, K.-R. Müller, and O. A. von Lilienfeld, Fast and Accurate Modeling of Molecular Atomization Energies with Machine Learning, Phys. Rev. Lett. **108**, 058301 (2012).

[45] J. R. Moreno, G. Carleo, and A. Georges, Deep Learning the Hohenberg-Kohn Maps of Density Functional Theory, Phys. Rev. Lett. **125**, 076402 (2020).

[46] T. E. Baker, E. M. Stoudenmire, L. O. Wagner, K. Burke, and S. R. White, One-dimensional mimicking of electronic structure: The case for exponentials, Phys. Rev. B **91**, 235141 (2015).

[47] J. P. Perdew, A. Ruzsinszky, J. Tao, V. N. Staroverov, G. E. Scuseria, and G. I. Csonka, Prescription for the design and selection of density functional approximations: More constraint satisfaction with fewer fits, J. Chem. Phys. **123**, 062201 (2005).

[48] See Supplemental Material at http://link.aps.org/supplemental/10.1103/PhysRevLett.126.036401 for information about 1D model systems, computational details of DMRG, KS calculations, training, validation and test, neural networks, training without KSR, and training with weaker KSR. It includes Refs. [2,33,42,49–53].

[49] D. C. Liu and J. Nocedal, On the limited memory BFGS method for large scale optimization, Math. Program. **45**, 503 (1989).

[50] M. Fishman, S. R. White, and E. M. Stoudenmire, The ITensor Software Library for Tensor Network Calculations, arXiv:2007.14822.

[51] K. He, X. Zhang, S. Ren, and J. Sun, Delving deep into rectifiers: surpassing human-level performance on ImageNet classification, in *Proceedings of the IEEE International Conference on Computer Vision* (IEEE, New York, 2015), pp. 1026–1034.

[52] X. Glorot and Y. Bengio, Understanding the difficulty of training deep feedforward neural networks, in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics* (2010), pp. 249–256, http://proceedings.mlr.press/v9/glorot10a.html.

[53] P. Virtanen *et al.*, SciPy 1.0: fundamental algorithms for scientific computing in Python, Nat. Methods **17**, 261 (2020).

[54] L. O. Wagner, E. M. Stoudenmire, K. Burke, and S. R. White, Guaranteed Convergence of the Kohn-Sham Equations, Phys. Rev. Lett. **111**, 093003 (2013).

[55] G. Kresse and J. Furthmüller, Efficient iterative schemes for ab initio total-energy calculations using a plane-wave basis set, Phys. Rev. B **54**, 11169 (1996).

[56] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, Learning internal representations by error propagation, in *Parallel Distributed Processing: Explorations in the Microstructure of Cognition* (1986).

[57] S. Bai, J. Z. Kolter, and V. Koltun, Deep equilibrium models, in *Advances in Neural Information Processing Systems (NeurIPS)* (MIT Press, Cambridge, 2019).

[58] K. He, X. Zhang, S. Ren, and J. Sun, Deep Residual Learning for Image Recognition, in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (Computer Vision Foundation, New York, 2016), pp. 770–778.

[59] S. R. White, Density Matrix Formulation for Quantum Renormalization Groups, Phys. Rev. Lett. **69**, 2863 (1992).

[60] M. Andrychowicz, M. Denil, S. Gomez, M. W. Hoffman, D. Pfau, T. Schaul, B. Shillingford, and N. De Freitas, Learning to learn by gradient descent by gradient descent, in *Advances in Neural Information Processing Systems* (MIT Press, Cambridge, 2016), pp. 3981–3989.

[61] N. Mardirossian and M. Head-Gordon, Thirty years of density functional theory in computational chemistry: an overview and extensive assessment of 200 density functionals, Mol. Phys. **115**, 2315 (2017).

[62] V. Nair and G. E. Hinton, Rectified linear units improve restricted Boltzmann machines, in *Proceedings of the International Conference on Machine Learning* (Omnipress, Madison, 2010).

[63] S. Elfwing, E. Uchibe, and K. Doya, Sigmoid-weighted linear units for neural network function approximation in reinforcement learning, Neural Netw. **107**, 3 (2018).

[64] P. Ramachandran, B. Zoph, and Q. V. Le, Searching for activation functions, arXiv:1710.05941.

[65] J. P. Perdew, A. Ruzsinszky, J. Sun, and K. Burke, Gedanken densities and exact constraints in density functional theory, J. Chem. Phys. **140**, 18A533 (2014).

[66] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (MIT Press, Cambridge, MA, 2016), http://www.deeplearningbook.org.

[67] J. Kukačka, V. Golkov, and D. Cremers, Regularization for deep learning: A taxonomy, arXiv:1710.10686.

[68] E. D. Cubuk, A. D. Sendek, and E. J. Reed, Screening billions of candidates for solid lithium-ion conductors: A transfer learning approach for small data, J. Chem. Phys. **150**, 214701 (2019).

[69] J. Hollingsworth, L. Li, T. E. Baker, and K. Burke, Can exact conditions improve machine-learned density functionals?, J. Chem. Phys. **148**, 241743 (2018).

[70] N. Thomas, T. Smidt, S. Kearnes, L. Yang, L. Li, K. Kohlhoff, and P. Riley, Tensor field networks: Rotation- and translation-equivariant neural networks for 3D point clouds, arXiv:1802.08219.

[71] K. Schütt, M. Gastegger, A. Tkatchenko, K.-R. Müller, and R. J. Maurer, Unifying machine learning and quantum chemistry with a deep neural network for molecular wavefunctions, Nat. Commun. **10**, 5024 (2019).

[72] R. Kondor, H. T. Son, H. Pan, B. Anderson, and S. Trivedi, Covariant compositional networks for learning graphs, arXiv:1801.02144.

[73] S. Seo and Y. Liu, Differentiable physics-informed graph networks, arXiv:1902.02950.

[74] M. Tsubaki and T. Mizoguchi, Quantum Deep Field: Data-Driven Wave Function, Electron Density Generation, and Atomization Energy Prediction and Extrapolation with Machine Learning, Phys. Rev. Lett. **125**, 206401 (2020).

[75] M. Raissi, P. Perdikaris, and G. E. Karniadakis, Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations, J. Comput. Phys. **378**, 686 (2019).

[76] R. Sharma, A. B. Farimani, J. Gomes, P. Eastman, and V. Pande, Weakly-supervised deep learning of heat transport via physics informed loss, arXiv:1807.11374.

[77] A. Krizhevsky, I. Sutskever, and G. E. Hinton, ImageNet classification with deep convolutional neural networks, in *Advances in Neural Information Processing Systems* (Curran Associates, Inc., Red Hook, 2012), pp. 1097–1105.

[78] L. v. d. Maaten and G. Hinton, Visualizing Data using t-SNE, J. Mach. Learn. Res. **9**, 2579 (2008).

[79] G. Tucker, A. Mnih, C. J. Maddison, J. Lawson, and J. Sohl-Dickstein, REBAR: Low-variance, unbiased gradient estimates for discrete latent variable models, in *Advances in Neural Information Processing Systems* (MIT Press, Cambridge, 2017), pp. 2627–2636.

[80] S. Ross and D. Bagnell, Efficient reductions for imitation learning, in *Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics* (2010), pp. 661–668, http://proceedings.mlr.press/v9/ross10a.html.

[81] A. Mirhoseini, A. Goldie, M. Yazgan, J. Jiang, E. Songhori, S. Wang, Y.-J. Lee, E. Johnson, O. Pathak, S. Bae *et al.*, Chip placement with deep reinforcement learning, arXiv:2004.10746.

[82] K. B. Petersen and M. S. Pedersen, , *The Matrix Cookbook* (Technical University of Denmark, Kongens Lyngby, 2012).

[83] T. H. Lee, Adjoint method for design sensitivity analysis of multiple eigenvalues and associated eigenvectors, AIAA J. **45**, 1998 (2007).

[84] H. Xie, J.-G. Liu, and L. Wang, Automatic differentiation of dominant eigensolver and its applications in quantum physics, Phys. Rev. B **101**, 245139 (2020).

[85] https://github.com/google-research/google-research/tree/master/jax_dft.