Understanding Machine-Learned Density Functionals

Li Li,*[a] John C. Snyder,[b,c] Isabelle M. Pelaschier,[a,d] Jessica Huang,[e] Uma-Naresh Niranjan,[f] Paul Duncan,[e] Matthias Rupp,[g] Klaus-Robert Müller,*[b,h] and Kieron Burke[a,e]

Machine learning (ML) is an increasingly popular statistical tool for analyzing either measured or calculated data sets. Here, we explore its application to a well-defined physics problem, investigating issues of how the underlying physics is handled by ML, and how self-consistent solutions can be found by limiting the domain in which ML is applied. The particular problem is how to find accurate approximate density functionals for the kinetic energy (KE) of noninteracting electrons. Kernel ridge regression is used to approximate the KE of non-interacting fermions in a one dimensional box as a functional of their density. The properties of different kernels and methods of cross-validation are explored, reproducing the physics faithfully in some cases, but not others. We also address how self-consistency can be achieved with information on only a limited electronic density domain. Accurate constrained optimal densities are found via a modified Euler-Lagrange constrained minimization of the machine-learned total energy, despite the poor quality of its functional derivative. A projected gradient descent algorithm is derived using local principal component analysis. Additionally, a sparse grid representation of the density can be used without degrading the performance of the methods. The implications for machine-learned density functional approximations are discussed. © 2015 Wiley Periodicals, Inc.

DOI: 10.1002/qua.25040

Introduction

Since the early days of quantum mechanics, it has been known that sufficiently accurate solutions of Schrödinger's equation for electrons and nuclei yield good predictions of the properties of solids and molecules.[1] But the Coulomb repulsion between electrons causes the computational cost of solving the Schrödinger equation to grow rapidly with the number of electrons, *N*.[2] However, as Hohenberg and Kohn proved in 1964,[3] the one-electron density may be used as the basic variable of quantum mechanics instead of the wavefunction, greatly reducing the complexity of the computational problem. This is called density functional theory (DFT).[4] In principle, the mapping of the electron density to energy is exact, but in practice, both the kinetic energy (KE) and the energy of the interaction between electrons must be approximated. In the original, Thomas-Fermi theory,[5,6] a local density functional approximation to the KE is used. However, Thomas-Fermi theory proved unsuitable for chemical and solid-state applications, as it does not bind matter.[7] Shortly after the Hohenberg-Kohn theorems, Kohn and Sham (KS)[8] found a middle ground by mapping the many-body system onto a fictitious system of noninteracting electrons which reproduce the exact electron density. The main reason KS DFT became successful is because the KE of these non-interacting electrons is an excellent approximation to the many-body KE. Simple approximations to the interaction energy produce much greater accuracy and reliability compared with the standard orbital-free DFT schemes built on Thomas-Fermi theory. However, the accuracy of the results is still sensitive to the approximation of the exchange-correlation (XC) functional. In the past four decades, there has been extensive research on improving density functional XC approximations. Development of both empirical and non-empirical functionals requires great intuition built on years of experience, as well as painstaking trial and error.[9–11]

Although KS DFT has enjoyed great success, the computational cost formally scales as $O(N^3)$. Despite the existence of linear-scaling KS-DFT codes, there continues to be strong interest in constructing explicit density functional approximations to the KE, with the aim of further reductions in computational

[a] L. Li, I. M. Pelaschier, K. Burke
Department of Physics and Astronomy, University of California, Irvine, California 92697
E-mail: li.li@uci.edu

[b] J. C. Snyder, Klaus-Robert Müller
Machine Learning Group, Technical University of Berlin 10587, Germany

[c] J. C. Snyder
Max Planck Institute of Microstructure Physics, Weinberg 2, Halle, Saale 06120, Germany

[d] I. M. Pelaschier
Department of Physics, Vanderbilt University, Nashville, Tennessee 37235

[e] J. Huang, P. Duncan, K. Burke
Department of Chemistry, University of California, Irvine, California 92697

[f] Uma-Naresh Niranjan
Department of Computer Science, University of California, Irvine, California 92697

[g] M. Rupp
Institute of Physical Chemistry and National Center for Computational Design and Discovery of Novel Materials (MARVEL), Department of Chemistry, University of Basel CH-4056, Switzerland
Present address: Fritz Haber Institute of the Max Planck Society, 14195 Berlin, Germany

[h] Klaus-Robert Müller
Department of Brain and Cognitive Engineering, Korea University, Anam-Dong, Seongbuk-Gu, Seoul 136-713, Korea
E-mail: klaus-robert.mueller@tu-berlin.de

Contract grant sponsor: NSF; contract grant number: CHE-1240252.
Contract grant sponsor: SNF; contract grant number: PPOOP2_138932.

cost.[12–17] A sufficiently accurate approximation to $T_s[n]$, the KE of KS electrons as a functional of the ground-state density $n(\mathbf{r})$, would enable highly accurate orbital-free DFT calculations with the same accuracy as KS DFT at a fraction of the computational cost. For example, benchmark orbital-free DFT calculations are capable of treating millions of atoms in metals[18] or proteins in solvent.[19] Note that accuracy in T_s beyond that of current XC approximations would be unnecessary, since all standard orbital-free DFT schemes utilize the KS decomposition of the energy, so that standard XC approximations developed for KS DFT can be utilized.[20] However, since T_s is typically comparable to the total energy of the system,[4] an unavoidable problem is that a useful KE functional calls for much stricter relative accuracy than XC functionals. Additionally, accurate functional derivatives are required because one finds the ground state density by solving an Euler equation with the approximate KE functional. Continued efforts have been made in this research direction, with some notable progress.[14,21–31] For a review of state-of-the-art orbital-free DFT functionals, we refer the reader to Ref. [13].

In DFT, functionals typically fall into two categories. Nonempirical functionals derived from first principles are designed to work well across a broad range of systems, and may exhibit systemic errors in treating certain types of interactions. Semi-empirical functionals introduce parameters that are fitted to standard data sets, and are typically more accurate with less systematic errors on systems, which they are designed for.

Recently, some of us applied machine learning (ML) in a completely new approach to approximating density functionals.[20,32,33] In a proof of principle, kernel ridge regression was used to approximate the KE of noninteracting fermions confined to a one dimensional (1d) box as a functional of the electron density.[33] In that work, a modified orbital-free DFT scheme was able to produce highly accurate self-consistent densities and energies that were systematically improvable with additional training data. ML algorithms are capable of learning high-dimensional patterns by nonlinear interpolation between given data. These powerful methods have proved to be very successful in many applications,[34] including medical diagnoses,[35] brain imaging,[36] automated text categorization,[37,38] and others.

This new approach to density functional approximation does not suffer from the typical challenges found in semi-local approximations, such as bond breaking,[20] but presents many new ones. First and foremost, ML is data-driven: reference calculations are needed to build a model for the KE functional. Since every iteration in a KS DFT calculation provides an electron density and its exact noninteracting KE, reference data are relatively easy to obtain. Additionally, the ML approximation (MLA) to the KE may have thousands or millions of parameters and satisfy none of the standard exact conditions in DFT, such as positivity, scaling, and exactness for a uniform electron gas. However, the form of the MLA is completely general and thus directly approximates the functional itself, suffering none of the typical issues plaguing standard functionals starting from a local approximation. For example, some of us recently showed that an MLA for the KE has no problem accurately dissociating soft-Coulomb diatomics in 1d—a huge challenge for standard approximations.[20] However, kernel ridge regression is strictly a method of interpolation. An MLA can only be used on systems it was designed for.

ML has recently become popular for the prediction of properties from structures, while often being trained on DFT calculations.[39–50] However, using ML to find density functionals themselves is a much more subtle business. There are hidden difficulties behind the naive application of ML methods to finding density functionals. These make it impossible to immediately apply ML to modern electronic structure calculations and expect useful results. Examples of such difficulties include finding functional derivatives and taking advantage of symmetries and near symmetries in the density.

In such a novel area of research, it can be difficult to disentangle various sources of error. A reasonable application of ML will often work, but not improve with the number of training data as rapidly as one wishes. A simple insight is often sufficient to greatly increase convergence. The philosophy behind the current (and subsequent) work is to identify and resolve each of these issues in the simplest possible context. Here, we study the case of noninteracting fermions, namely 1d particles in a box with various potentials, and the problem of ML functional derivatives. This is a simple example of finding $T_s[\boldsymbol{n}]$, the KE of such electrons, as a functional of their density, the problem at the heart of all orbital-free DFT. Ref. [33] was a preliminary report; the present work is a thorough exploration. In particular, we investigate the use of various kernels and their properties and the efficiency of various cross validation methods, showing which methods capture the correct physics of the KE, and which do not. We discuss the issue of functional derivatives of the MLA in greater detail, and explain how a modified constraint to the standard Euler equation enables highly accurate self-consistent densities, or constrained optimal densities, to be found. Additionally, a projected gradient descent algorithm is derived using local principal component analysis (PCA) to solve the modified Euler equation. Finally, we explore the use of a sparse grid representation of the electron density.

Theory and Background

Throughout this work, we consider only noninteracting same-spin fermions in one dimension. Thus, all electron densities $n(x)$ are fully spin-polarized. Atomic units are used in symbolic equations, but energies are usually presented in kcal/mol.

Model system

Consider N non-interacting same-spin fermions subject to a smooth external potential in one dimension, with hard walls at $x=0$ and $x=1$. We restrict this study to a simple class of potentials, namely a sum of 3 Gaussian dips with varying heights, widths and centers:

$$v(x) = -\sum_{i=1}^{3} a_i \exp\left(-(x-b_i)^2/(2c_i^2)\right), \tag{1}$$

for $x \in [0,1]$, and $v(x)=\infty$ elsewhere. The Hamiltonian for this system is simply $\hat{H}=\hat{T}+\hat{V}$, where $\hat{T}=-\partial^2/2\partial x^2$ and $\hat{V}=v(x)$. We solve the Schrödinger equation

$$\left(-\frac{1}{2}\frac{\partial^2}{\partial x^2}+v(x)\right)\phi(x)=\epsilon\phi(x), \tag{2}$$

for the eigenvalues ϵ_j and orbitals $\phi_j(x)$. As our fermions are same-spin, each orbital $\phi_j(x)$ is singly occupied. Thus, the electron density is given by

$$n(x)=\sum_{j=1}^{N}|\phi_j(x)|^2, \tag{3}$$

and the KE is

$$T=\frac{1}{2}\sum_{j=1}^{N}\int_0^1 dx|\phi_j'(x)|^2. \tag{4}$$

A dataset is created by randomly sampling $a_i\in[1,10]$, $b_i\in[0.4,0.6], c_i\in[0.03,0.1]$, to generate 2000 different potentials. For each potential, the system is occupied with up to 4 fermions, and the exact densities and kinetic energies are computed. Numerically, the Schrödinger equation is solved by discretizing the density on a grid:

$$x_l=(l-1)/(N_G-1), \quad l=1,\ldots,N_G \tag{5}$$

where $\Delta x=1/(N_G-1)$ is the grid spacing. Numerov's method[51] together with a shooting method is used to solve for the eigenvalues and eigenfunctions of eq. (2). For $N_G=500$, the error in our reference kinetic energies is $<10^{-7}$. Figure 1 gives a few sample densities and their corresponding potentials.

The data used here is identical to that of Ref. [33]. The exact values of the parameters used in each sample are given in the Supporting Information of Ref. [33]. Of the 2000 samples generated, the first half was reserved for training while the second half was reserved for testing (which we refer to as the test set).

Orbital-free DFT

In orbital-free DFT, T_s is approximated as a functional of $n(x)$. For our model system with non-interacting fermions, the total energy is given as

$$E_v=\min_n \{T[n]+V[n]\}, \tag{6}$$

for a given potential $v(x)$. The potential is known exactly as a functional of $n(x)$:

$$V[n]=\int_0^1 dx\, n(x)v(x). \tag{7}$$

By the variational principle, the ground-state density is found by the Euler-Lagrange constrained search

$$\delta\left\{E_v[n]-\mu\left(\int n(x)\,dx-N\right)\right\}=0, \tag{8}$$

where the chemical potential μ is adjusted to produce the required particle number N. This becomes simply

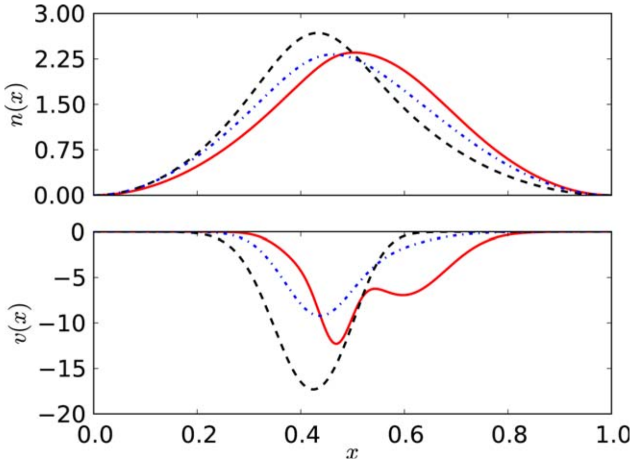

Figure 1. A few sample densities and their corresponding potentials, for $N=1$.

$$\frac{\delta T[n]}{\delta n(x)}=\mu-v(x). \tag{9}$$

The density that satisfies this equation, minimizing $E_v[n]$ with the normalization constraint, is found self consistently.

Given the exact functional $T[n]$, solving eq. (9) will yield the exact ground-state density of the system. But in practice, T must be approximated. Let $\tilde{T}$ be such an approximation, $n(x)$ be the exact density, and $\tilde{n}(x)$ be the self-consistent density found with $\tilde{T}$. There are two measures of the error of such an approximate $\tilde{T}$.[52] The first is to compute the functional-driven error $\Delta T_F=\tilde{T}[n]-T[n]$, which is simply the error in the KE evaluated on the exact density. The second (and much more difficult) test is to insert $\tilde{T}$ into eq. (9), solve for the approximate density $\tilde{n}$, and compute its error relative to the KE of the exact density $\Delta E=\tilde{E}_v[\tilde{n}]-E_v[n]$. Then the density-driven error is defined as $\Delta E_D=\Delta E-\Delta T_F$.[52] This is the additional error incurred by the approximate density. In practice, a functional, which only satisfies the first test, is not much use, as the ground-state density itself must also be obtained from this approximation. In orbital-free DFT, self-consistent results can be much worse than energies of KS densities, as inaccuracies in the functional derivative can cause large errors in the corresponding density. In the case of the KE functional for real systems, functional derivatives of traditional approximations can have singularities at the nuclei, making all-electron calculations very difficult, if not impossible, to converge.[13] Many of these problems can be avoided through use of pseudopotentials,[13,31] but in general the solution for eq. (9) is nontrivial.

As mentioned above, the simplest density functional approximation to T_s is the local approximation,[4] which for spin-polarized densities in 1d is

$$T^{\mathrm{loc}}[n]=\frac{\pi^2}{6}\int dx\, n^3(x). \tag{10}$$

For $N=1$, the exact KE has the von Weizsäcker[21] form:

$$T^{\mathrm{W}}[n]=\int dx\,\frac{n'(x)^2}{8n(x)}. \tag{11}$$

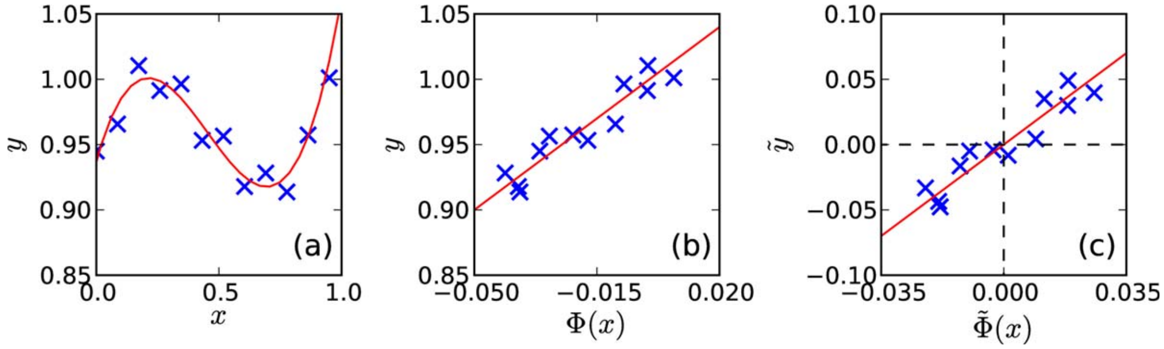

Figure 2. (a) An example 1d noisy data set. (b) Transformation to feature space $\Phi(x)$. (c) Centering of data in feature space.

As was shown in Ref. [33], the local approximation does poorly. The mean absolute error (MAE) on the test set is 217 kcal/mol, and self-consistent results are even worse at 1903 kcal/mol. A standard extension of the local approximation to a semi-local form is to add a fraction of $T^{\mathrm{W}}[n]$ to $T^{\mathrm{loc}}[n]$, forming a modified gradient expansion approximation. It was shown in Ref. [33] that this did little to improve upon the local approximation.

Data topology and representation

Typically in ML, the data has a finite representation. For example, in Ref. [39], molecular structures are represented by a Coulomb matrix and the model predicts atomization energies. In contrast, the electronic density $n(x)$ is a continuous function restricted to the domain[53]

$$\mathcal{J}_N \equiv \left\{ n \,|\, n(x) \geq 0, n^{1/2}(x) \in H^1(\mathbb{R}), \int n(x)\, dx = N \right\}, \qquad (12)$$

where $H^1(\mathbb{R})$ is a Sobolev space.* Although $\mathcal{J}_\mathrm{N}$ is infinite dimensional, in practice $n(x)$ is expanded in a finite basis (with N_G basis functions). In this work, we use a real space grid to represent the density, since our reference calculations are done using the same grid. We use the L^2 inner product and norm between densities $n_i(x), n_j(x)$

$$\langle n_i, n_j \rangle = \int_{-\infty}^{\infty} dx\, n_i(x) n_j(x), \qquad \|n\| = \sqrt{\langle n, n \rangle}. \qquad (13)$$

(In actual calculations, all densities are represented on a finite basis, and thus will have have a finite L^2-norm). Since the ML algorithm is expressed in terms of this inner product, the results are independent of the specific representation used as long as the basis is converged.

Even with a truncated basis, $\mathcal{J}_N$ is still high-dimensional and applying ML to learn the KE of all densities in $\mathcal{J}_N$ would not be feasible. Fortunately, we are only interested in a subspace of $\mathcal{J}_N$ related to a specific class of potentials (e.g., Gaussian dips), which greatly reduces the variety of possible densities. In general, let the potential $v(x)$ be parametrized by the parameters $\{p_1, \ldots, p_d\}$. We define the density manifold[54] $\mathcal{M}_N \subset \mathcal{J}_N$ as the set of all densities that come from these potentials with a given particle number N. In general, $\mathcal{M}_N$ is a d-dimensional manifold. The training densities, $n_j(x)$ for $j = 1, \ldots, N_T$, are sampled from $\mathcal{M}_N$. In this work, the external potential has 9 parameters, and thus d is at most 9.

The kernel trick and feature space

In finding the structure of low-dimensional data, it is often sufficient to optimize parametrized nonlinear forms (e.g., using a polynomial to fit a sinusoid). For high-dimensional, nonlinear data this becomes increasingly difficult. In kernel-based ML, the approach is to transform the data itself nonlinearly to a high-dimensional space known as a feature space, such that the data become linear.[34,55–58]

Let the data points belong to a vector space χ, also called input space, and let $\Phi : \chi \to F$ be the map to feature space F. As a conceptual example, Figures 2a and 2b show the transformation of nonlinear data to feature space, where linear regression can be used to fit the data. For the sake of illustration, both the input vector x and feature vector $\Phi(x)$ are 1d. In practice, the dimensionality of the feature space F is much higher than the input space (even infinite).

Assuming we wish to apply a linear method such as regression in feature space F, we note that regression can be expressed solely in terms of the inner product between feature vectors $\Phi(x)$ and $\Phi(y)$, where $x, y \in \chi$. We define the kernel k such that

$$k(x, y) = \langle \Phi(x), \Phi(y) \rangle. \qquad (14)$$

The kernel can generally be thought of as a measure of similarity between data, but must satisfy positive definiteness:

$$\text{For all } x_i \in \chi, c_i \in \mathbb{R} : \sum_{i,j} c_i c_j k(x_i, x_j) \geq 0. \qquad (15)$$

Positive definiteness guarantees the existence of a feature space F,[59] which is a reproducing kernel Hilbert space.[60]

*A Sobolev space $W^{k,p}(\mathbb{R})$ is a vector space of functions with a norm that is a combination of L^p-norms of the function itself and its derivatives up to a given order k. It is conventional to write $W^{1,2}(\mathbb{R})$ as $H^1(\mathbb{R})$. $f \in H^1(\mathbb{R})$ means that f and its first order derivative are in L^2.

Since the linear algorithm in F may be expressed in terms of the kernel in eq. (14), Φ need never be explicitly computed. This procedure, known as the kernel trick, enables easy nonlinearization of any method that can be expressed via an inner product.[61]

Kernel ridge regression

Kernel ridge regression is a nonlinear version of regression with a regularization term to prevent overfitting.[62] Our MLA for the KE has the form

$$T^{\mathrm{ML}}[n]=\sum_{j=1}^{N_T}\alpha_j k[n,n_j], \tag{16}$$

where N_T is the number of training densities, α_j are weights to be determined, n_j are training densities and $k[n,n_j]$ is the kernel. The weights are found by minimizing the quadratic cost plus regularization

$$\mathcal{C}(\boldsymbol{\alpha})=\sum_{p=1}^{M}\left(T^{\mathrm{ML}}[n_p]-T[n_p]\right)^2+\lambda\boldsymbol{\alpha}^{\top}\boldsymbol{K}\boldsymbol{\alpha}, \tag{17}$$

where $\boldsymbol{\alpha}=(\alpha_1,\ldots,\alpha_{N_T})$, $\boldsymbol{K}$ is the kernel matrix, $\boldsymbol{K}_{ij}=k[n_i,n_j]$, and λ is called the regularization strength. The second term penalizes weights with large magnitudes to prevent overfitting.[†] By setting the gradient of eq. (17) to zero, minimizing $\mathcal{C}(\boldsymbol{\alpha})$ gives

$$\boldsymbol{\alpha}=(\boldsymbol{K}+\lambda\boldsymbol{I})^{-1}\boldsymbol{T}, \tag{18}$$

where $\boldsymbol{I}$ is the identity matrix and $\boldsymbol{T} = (T[n_1],\ldots,T[n_{N_T}])$. The hyperparameters, which include the regularization strength λ and the parameters of the kernel are found via cross validation (see Ref. [40] and Section "Model selection").

The choice of the kernel will depend on the given data. Some kernels are designed to be generally robust and applicable (e.g., the Gaussian kernel), while others are designed for a specific type of data (see e.g., Ref. [34,64,65]). A good choice of kernel can reflect the characteristics of the data (see Ref. [66]). In Ref. [33], we chose the Gaussian kernel

$$k[n_i,n_j]=\exp\left(-\|n_i-n_j\|^2/2\sigma^2\right), \tag{19}$$

where σ is the length scale. Since the density is represented on a uniform grid, the L^2-norm can be approximated by[‡]

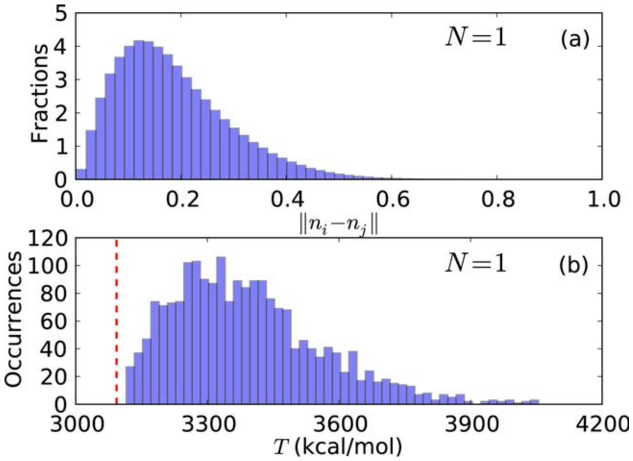

Figure 3. (a) Normalized distribution of the Euclidean distance between all distinct pairs of densities in the dataset (2000 densities). The maximum distance between any pair is 0.9. (b) Histogram of the KE in the dataset. The vertical dashed line at 3093 kcal/mol is the ground-state energy of one fermion in a flat box of length 1.

$$\|n_i-n_j\|^2 \approx \Delta x\sum_{l=1}^{N_G}\left(n_i(x_l)-n_j(x_l)\right)^2, \tag{20}$$

where x_l is given by the grid defined in eq. (5). This approximation becomes exact as $\Delta x \to 0$. Figure 3 shows the range and distribution of Euclidean distances between all pairs of densities and KE of all densities in the dataset with $N=1$.

Ordinary linear regression models frequently employ a bias term to account for the fact that the data might lie away from the origin. Without this term, the regression line is forced to go through the origin, causing a systematic error if the data does not. The bias term can be implemented directly, or by centering both the densities and KEs of the samples such that the mean is zero. Figure 2 illustrates the transformation to feature space for an example 1d data set and linear regression in feature space. If the data are centered in feature space, the bias term is unnecessary. Here, we center the densities in features space such that $\sum_{j=1}^{N_T}\Phi(n_j) = 0$. We define the centered map to feature space $\tilde{\Phi}(n)=\Phi(n)-\sum_{j=1}^{N_T}\Phi(n_j)/N_T$. Then the centered kernel is[61]

$$\begin{aligned}\tilde{k}[n,n'] &= \langle\tilde{\Phi}(n),\tilde{\Phi}(n')\rangle\\ &= k[n,n']-\frac{1}{N_T}\sum_{j}^{N_T}\left(k[n',n_j]+k[n,n_j]\right)\\ &\quad+\frac{1}{N_T^2}\sum_{i,j=1}^{N_T}k[n_i,n_j].\end{aligned} \tag{21}$$

And the KEs should be centered as

$$\tilde{T}[n]=T[n]-\sum_{j=1}^{N_T}T[n_j]/N_T. \tag{22}$$

For simplicity, all equations given in this work assume that the data are centered (i.e., $k=\tilde{k}$). In fact, kernels such as the Gaussian kernel in eq. (19), whose induced reproducing kernel

[†] The regularization term imposes certain smoothness conditions on the model (see Ref. [58,63]), making it robust against noisy data (e.g., experimental data) and generalizable. Although our reference data is deterministic and thus noise-free in this sense, the limited precision of our reference calculations may be attributed as numerical noise. Moreover, regularization makes the matrix inversion in Eq. (18) stable.

[‡] Note that, in Ref. [33], the same representation for the density was used, but the densities were treated as vectors, so that the standard Euclidean distance was used in the kernel. This is equivalent to the formulation here, except our notation is more general now (e.g., Simpson's rule could be used to approximation the L^2-norm instead of a Riemann sum), and the length scale in Gaussian kernel here is related to the scale of the kernel in Ref. [33] by a factor of $\sqrt{\Delta x}$.

Table 1. Standard kernels.

Kernel	$k[n, n']$
Gaussian	$\exp\left(-\Vert n-n'\Vert^2/2\sigma^2\right)$
Cauchy	$\left(1+\Vert n-n'\Vert^2/\sigma^2\right)^{-1}$
Laplacian	$\exp\left(-\Vert n-n'\Vert/2\sigma\right)$
Wave	$\frac{\theta}{\Vert n-n'\Vert}\sin\frac{\Vert n-n'\Vert}{\theta}$
Power	$-\Vert n-n'\Vert^d$
Linear	$\langle n, n'\rangle$

The parameters σ, θ, and d are kernel parameters. The linear kernel has no parameters.

Hilbert space on a bounded domain is dense in the space of continuous functions on this domain do not require centering.[67]

Model Selection

Kernels

Model selection refers to the process of selecting a kernel and the corresponding hyperparameters. In kernel ridge regression, this includes the regularization strength λ and the kernel parameters (e.g., in the Gaussian kernel, the length scale σ). Table 1 lists some standard kernels. Radial basis function (RBF) kernels, which include the Gaussian, Cauchy, and Laplacian kernels, all behave similarly and tend to work for a broad range of problems. Other kernels work well for specific data structures[34,64,65] and regularization properties.[61]

Figure 4 shows the contours of the functional-driven MAE over the test set as a function of the regularization strength λ and the kernel parameter σ. We see that the qualitative behavior is similar for the Gaussian, Cauchy, and Laplacian kernels. In the left region (where the contour lines are vertical), the length scale σ is much smaller than the distance between neighboring training densities. Thus the RBF-type kernel functions centered at each training density have minimal overlap, yielding a poor approximation to the KE functional. The kernel matrix becomes nearly unity, and the regularization λ has negligible effect. On the right side of the contour plot, the length scale is comparable to the global scale of the data. In these regions, the kernel functions are slowly varying and do not have enough flexibility to fit the nonlinearity in the data. The region with minimum MAE lies in the middle. The Gaussian and Cauchy kernels both give the same performance, with errors less than 1 kcal/mol in the middle region (enclosed by the dashed line), while the Laplacian kernel behaves poorly in comparison. This is likely due to the cusp in the form of the kernel, which cannot fit the smooth KE functional.

Optimization of hyperparameters

After picking a kernel family, the values of the hyperparameters must be chosen. Ideally, we select the hyperparameters such that the generalization error, which is the error not only on our training set but also on all future data, is minimal. The out-of-sample error must be estimated without looking at the test set (the test set is never touched during model selection, so that it can give a true test of the final performance of the model).[34,40] This procedure, performed by cross-validation, is essential for model selection in preventing overoptimistic performance estimates (overfitting).

Various schemes for cross validation exist,[34,40,68] but all obey a basic principle: the available data are subdivided into three parts: the training, validation, and the test sets. The ML model is built from the training set and the hyperparameters are optimized by minimizing the error on the validation set (Fig. 5). The test set is never touched until the weights and hyperparameters have been determined. Then and only then, the generalization ability of the model can be assessed with the test data[34,40] (see also Ref. [36]). Typically, the data are shuffled to ensure its random distribution between training and validation division. This can be repeated with different subdivisions. A few schemes, which will be analyzed for our KE functional estimation problem, are described below. For each scheme, a test set of 1000 samples is used to estimate the generalization error after the ML model is selected.

Simple Cross Validation. The training data (N_T samples) are randomly divided into a training set of 70% and a validation set (hold-out set) of 30%. The hyperparameters are optimized by minimizing the MAE on the validation set.

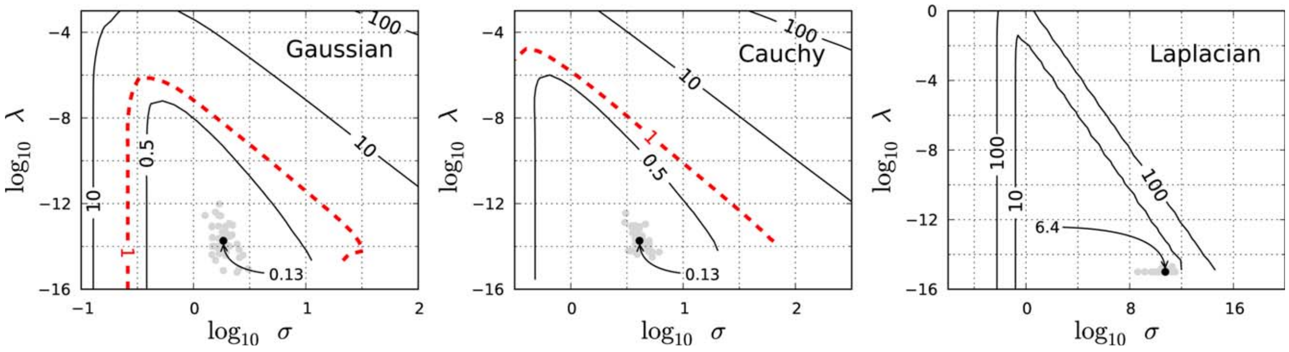

Figure 4. Contour plots of the functional-driven MAE $\overline{|\Delta T_F|}$ over the test set in kcal/mol for selected kernels with $N_T = 100$ for $N = 1$. The dashed line delineates the region where the model achieves chemical accuracy. Each gray dot gives the optimal choice of hyperparameters from a randomized 10-fold cross validation. The black dot denotes the median over 40 repetitions. In the lower right region (i.e., small λ and large σ), the matrix inverse in eq. (18) is numerically unstable due to the limited precision of the calculation.

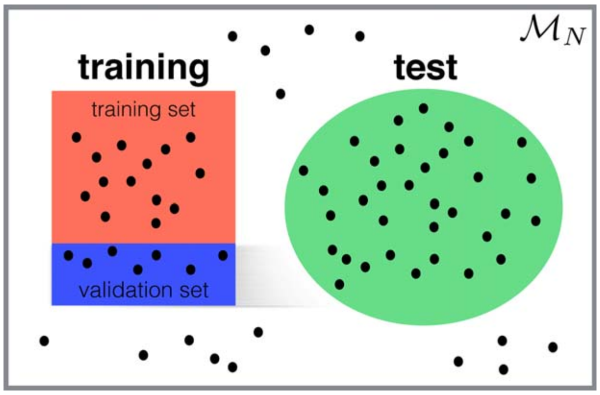

Figure 5. Cartoon shows the relation of each data set in $\mathcal{M}_N$. Each black dot represents a sample (density and its corresponding KE). Training, validation, and test set are subsets of the full data set.

k-Fold Cross Validation.

Step 1 The N_T training data are randomly divided into k bins.

Step 2 The jth bin is used as the validation set and the remaining $k-1$ bins as training set. The model is built on the training set and the hyperparameters are selected by minimizing the MAE on the validation set.

Step 3 Repeat step 2 k times such that all bins have been used as validation sets. We will then have k models in total and the final hyperparameters are selected as the median over all models.

Because the mean cross validation error still depends on the initial random partitioning of data in cross validation, we repeat the procedure with different subdivisions.[34]

Leave-One-Out (LOO). LOO is a special case of k-fold cross validation, when $k=N_T$. Thus each bin contains only one sample.

Typically, it is better to leave out as little data as possible to exploit the statistical power in the data. Simple cross validation is computationally expedient, but wasteful since not all training data participates in the optimization. k-fold cross validations are used in situations where data are very limited, or expensive to collect. LOO is often used with limited data and it becomes computationally intensive if N_T is large. k-fold cross validation gives a good balance on all counts.

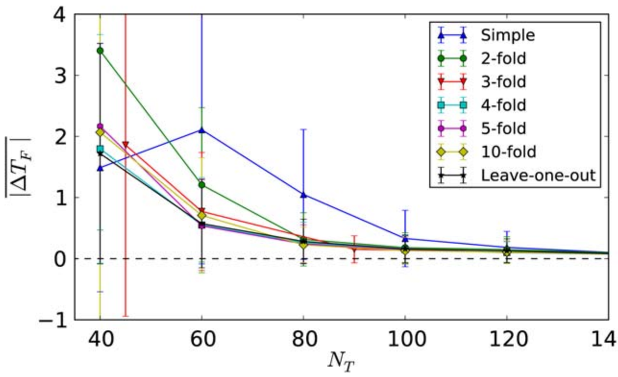

Figure 6. Comparison of different cross validation methods, including simple, 2-fold, 3-fold, 4-fold, 5-fold, 10-fold, and LOO. The mean of the absolute functional-driven error $\overline{|\Delta T_F|}$ (in kcal/mol) is evaluated on the test set and the error bars represent the standard deviation.

For the Gaussian kernel, Figure 6 shows the MAE on the test set with the hyperparameters optimized with different cross validation methods. With 120 training densities, all schemes give a similar MAE, despite the large variations in σ and λ. This means that multiple models exist that give comparable performance. As expected, the only variations in MAE occur for more limited data.

Figure 4 shows how 10-fold cross validation performs in selecting hyperparameters that generalize well to the test set, for a few kernels. The gray dots represent the optimal parameter choice for each repetition, and the black dot is the median over all repetitions. In this case, the global minimum of the MAE lies in a relatively flat basin. Each randomized cross validation lies near the true minimum, indicating the model generalizes well to the test set.

Finally, we use 10-fold cross validation (repeated 40 times) to optimize the hyperparameters. Table 2 shows the optimal hyperparameters and functional driven errors for the kernels listed in Table 1. Some optimum values for the Gaussian kernel are listed in Table 3. Detailed information with optimum values of other kernels are shown in Supporting Information.

Results and Discussion

In the main work of this article, we test in greater detail some of the methods that were introduced in Ref. [33] using only the Gaussian kernel, as it performs the best (together with the Cauchy kernel).

Errors on exact densities

In Table 3, we evaluate our MLA, constructed using the first N_T training densities in our data set, on the exact densities of the test set and compute the errors $\Delta T_F = T^{ML}[n]-T[n]$. The Gaussian and Cauchy kernels work well, capturing the physics behind the KE functional. For the Gaussian kernel with $N=1$ chemical accuracy is achieved (i.e., MAE less than 1 kcal/mol) at $N_T=60$. Just as we saw in Ref. [33], the performance is systematically improvable with increasing number of training densities. The Laplacian kernel gives a MAE of 6.9 kcal/mol at

Table 2. The optimal hyperparameters found through 10-fold cross validation and the MAE over the test set for various kernels with $N=1$ and $N_T=100$.

Kernel	λ	p	$\overline{\lvert\Delta T_F\rvert}$	$\lvert\Delta T_F\rvert^{max}$
Gaussian	$4.5\cdot10^{-14}$	1.6	0.13	3.4
Cauchy	$7.8\cdot10^{-14}$	3.5	0.13	2.9
Laplacian	$1.0\cdot10^{-15}$	$3.6\cdot10^{5}$	6.4	231
Linear	$6.2\cdot10^{-1}$	–	53.1	380
Wave	$4.5\cdot10^{-1}$	0.14	19.2	252
Power	$1.0\cdot10^{-13}$	1.96	3.3	104

The kernel parameter p refers to σ for the Gaussian, Cauchy, and Laplacian kernels, θ for the wave kernel and d for the power kernel. The linear kernel has no parameter. Errors are given in kcal/mol.

Table 3. Hyperparameters and errors measured over the test set using the Gaussian kernel, for different N and N_T.

				$\|\Delta T_F\|$		$\|\Delta T\|$		$\|\Delta E\|$	
N	N_T	$\lambda \cdot 10^{14}$	σ	Mean	Max	Mean	Max	Mean	Max
1	40	50	4.2	1.9	30	15	120	5.1	32
	60	10	1.8	0.62	11	3.0	19	0.66	4.4
	80	54	1.5	0.23	3.1	1.1	11	0.44	2.6
	100	4.5	1.6	0.13	3.5	1.4	16	0.41	2.3
	150	1.2	1.3	0.06	1.0	0.81	5.1	0.27	1.9
	200	1.3	1.0	0.03	0.87	0.67	10	0.28	1.6
2	60	60	3.0	0.46	4.8	1.79	9.9	0.73	3.6
	100	1.0	2.2	0.14	1.7	1.25	5.0	0.44	2.5
3	60	6.0	5.8	0.31	3.9	1.03	5.0	0.82	6.5
	100	1.9	2.5	0.13	1.7	1.11	8.3	0.59	3.8
4	60	0.6	14	0.46	5.4	2.44	9.5	0.93	6.3
	100	1.4	2.7	0.08	2.6	1.12	9.8	0.63	5.0
1−4	400	1.7	2.2	0.12	3.0	1.28	12.6	0.52	5.1

$N_T = 100$ (still better than LDA), which improves as N_T increases. However, the performance of the wave kernel does not improve as N_T increases (see Supporting Information). This indicates the form of the wave kernel is not flexible enough to fit the form of the KE functional.

Sparse grid

Note that the choice of N_G used in the reference calculations is needed to converge our reference energies and densities, but may be larger than the grid needed to "converge" our ML functional. As the ML model depends only on the inner product between densities, this will typically converge much faster than, e.g. Numerov's method. To demonstrate this, we define a "sparse" grid, $\{x_{s(l-1)+1} | l = 1, \dots, N_G/s\}$, using every sth point in the grid (we only choose s such that N_G is divisible by s).

Figure 7 shows that performance of the model is unaffected until N_G is reduced to about 10 grid points. The model is cross-validated each time, but the hyperparameters change only slightly. Thus, ML can accurately learn the KE functional with a far less complete basis than is required to accurately solve the Schrödinger equation. This is possible because we have restricted the learning problem to a simple type of potential with a limited range of possible densities and energies. The underlying dimensionality of the data is about 9, comparable to the number of parameters that determine the potential. The model needs only enough degrees of freedom in the representation of the density to distinguish between densities, but no more. Thus, it is no coincidence that the minimum grid required is comparable to the dimensionality of the data (i.e., the dimensionality of the density manifold $\mathcal{M}_N$).

However, we also need a sufficiently fine grid to compute the integral in eq. (7) to the desired accuracy. In the problem shown here, the dimensionality of the data is relatively small, and will increase for larger systems (e.g., real molecules with many degrees of freedom). In general, however, we need to consider both factors in choosing a suitable basis. However, we may be able to use a basis that is more sparse than that of the reference data, which would greatly reduce the computational cost of the method.

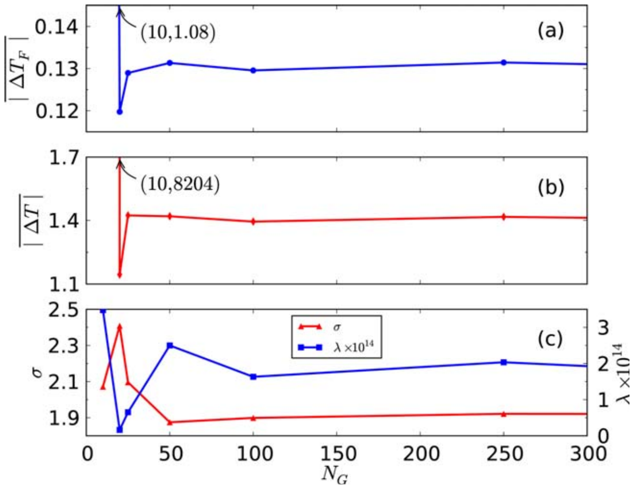

Figure 7. The effect of using a sparse grid to represent the density on the performance of the MLA, for $N = 1$, $N_T = 100$, with the Gaussian kernel. Here (a) $\overline{|\Delta T_F|} = \overline{T^{ML}[n] - T[n]}$ is the mean absolute functional-driven error of the MLA evaluated on the test set in kcal/mol, (b) $\overline{|\Delta T|} = \overline{T^{ML}[\tilde{n}] - T[n]}$ gives the error of KE evaluated on constrained optimal densities in kcal/mol and (c) the corresponding re-crossvalidated hyperparameters λ and σ. The MAE is completely unaffected as N_G is reduced until $\sim N_G = 10$, when it jumps sharply.

Challenge of finding density

Thus far, we have focused on the discussion of the performance of the MLA evaluated on exact densities (i.e., the functional-driven errors). However, in order for a functional to be useful, it must also predict the ground-state density. As discussed previously, an accurate functional derivative is necessary in order to solve eq. (9) and yield an accurate density. The functional derivative of our MLA is given by:

$$\frac{\delta T^{ML}[n]}{\delta n(x)} = \sum_{j=1}^{N_T} \alpha_j \frac{\delta k[n, n_j]}{\delta n(x)}, \tag{23}$$

where, for the Gaussian kernel,

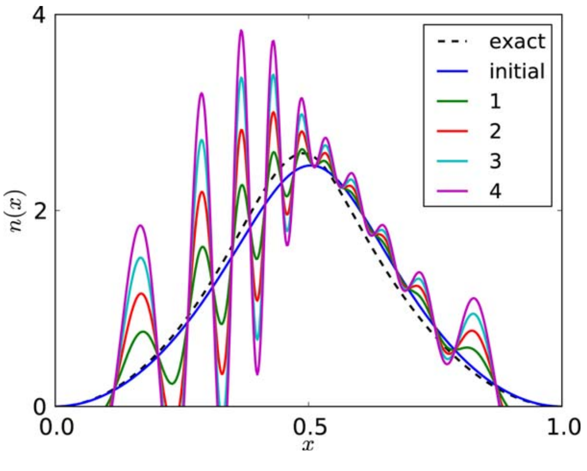

Figure 8. The first few steps in a standard gradient descent solving the Euler equation in eq. (9) using our MLA for the KE of $N = 1$ with $N_T = 100$ starting from a sample training density. The dashed line shows the exact self-consistent solution. The noise in the bare functional derivative quickly causes large corresponding errors in the density.

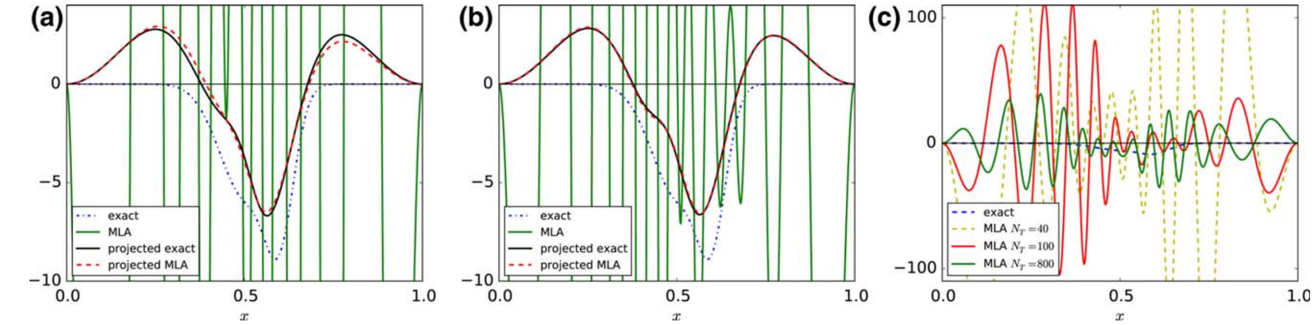

Figure 9. The functional derivative of our MLA (green) cannot reproduce the exact derivative *v*(*x*) (blue dot dashed) evaluated at the ground-state density, because this information is not completely contained in the data. However, both agree when projected onto the tangent of the data manifold $\mathcal{M}_N$ at *n* (black and red dashed). Shown for $N=1$, for (a) $N_T=40$ and (b) $N_T=100$, for a typical test sample. (c) The functional derivative of MLA for $N_T=40, 100, 800$. Oscillations are reduced, but not eliminated, by using larger training set size N_T.

$$\delta k[n, n_j]/\delta n(x)=(n_j(x)-n(x))k[n, n_j]/\sigma^2. \tag{24}$$

In Figures 9a and 9b, we plot the functional derivative of our model compared to the exact derivative. The model displays a highly inaccurate functional derivative, with a huge amount of apparent "noise," as was found in Ref. [33].

What is the source of this noise? In general, if the underlying dimensionality of the data is much less than the dimensionality of $\mathcal{J}_N$ (which in this case is essentially infinite), ML will be unable to capture the functional derivative. The functional derivative contains information on how the KE changes along any direction, but ML cannot learn this because it only has information in directions in which it has data (i.e., along $\mathcal{M}_N$). Figure 10 illustrates the problem: standard minimization techniques will rapidly exit the "interpolation" region in which the MLA is expected to be accurate. The MLA is only given information about how the KE changes along the density manifold $\mathcal{M}_N$. In the many dimensions orthogonal to $\mathcal{M}_N$, the MLA produces an inaccurate derivative (each of these dimensions produces a large relative error since no data exists in these directions; the sum over many dimensions creates a large total error in the functional derivative). As demonstrated in Figure 9c, adding training data containing more information along $\mathcal{M}_N$ reduces the oscillations of the inaccurate functional derivative but does not eliminate them.[69]

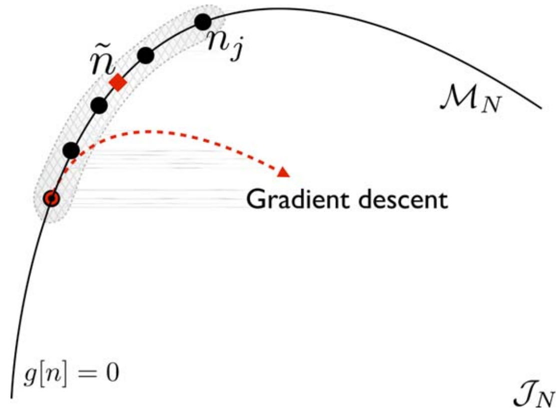

Figure 10. Cartoon illustrating the difficulty in solving for the self-consistent density with our MLA. Pictured are the density manifold $\mathcal{M}_N$ (curved solid line), the training densities $n_j \in \mathcal{M}_N$ (black circles), and the exact self-consistent density $\tilde{n}$ (red square). Here *g* is a functional that is identically zero on $\mathcal{M}_N$ and positive elsewhere. Thus, $\mathcal{M}_N$ is defined implicitly by $g[n]=0$. The shaded area, called the interpolation region, shows where the MLA is accurate. The solution of eq. 9 via exact gradient descent is given by the red dashed line, which becomes unstable and soon leaves the shaded area.

A standard gradient descent will quickly venture off of $\mathcal{M}_N$ into regions of $\mathcal{J}_N$ where the model is guaranteed to fail. Figure 8 shows the deviation of self-consistent density if the search is not constrained to $\mathcal{M}_N$. To fix this, we further constrain the minimization in eq. (9) to stay on $\mathcal{M}_N$.[70] The Euler-Lagrange minimization for the ground-state density can be expressed as

$$\delta\{E[n]-\zeta g[n]\}=0, \tag{25}$$

where *g* is any functional that is zero on $\mathcal{M}_N$ and positive elsewhere. Thus $g[n]=0$ implicitly defines the density manifold $\mathcal{M}_N$. Since any $n \in \mathcal{M}_N$ satisfies the normalization condition, the previous constraint is no longer necessary. Because the minimizing density (i.e. the ground-state density) is in $\mathcal{M}_N$ and thus satisfies the constraint $g[n]=0$, eq. (25) gives the same solution as eq. (9). Essentially, we have vastly reduced the domain of the search from $\mathcal{J}_N$ to $\mathcal{M}_N$. To avoid confusion, we call the minimizing density of this equation the constrained optimal density. It may be solved self-consistently in the same sense of solving the standard Euler equation. However, the $g[n]$ that exactly gives the density manifold is unknown. In the next section, we develop an approximation which attempts to reconstruct the density manifold from the training densities.

Manifold reconstruction using PCA

Our aim is to reconstruct $\mathcal{M}_N$ locally around a given density *n*(*x*), which is assumed to be on the density manifold. A simple approach is to approximate $\mathcal{M}_N$ as locally linear, using PCA to determine the tangent space empirically from the training densities. This will work as long as there are enough training densities covering the density manifold. First, we define a weighted average density around density *n*:

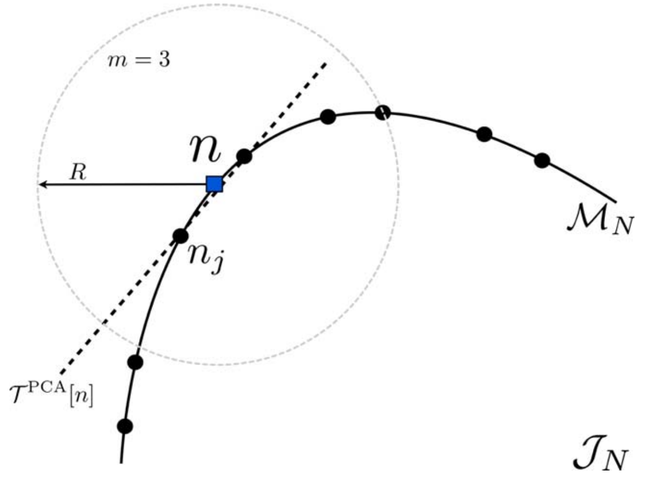

Figure 11. Cartoon showing the density manifold $\mathcal{M}_{\mathrm{N}}$ (curved line) that is contained in $\mathcal{J}_{\mathrm{N}}$, the training densities n_j for $j=1,\ldots,N_t$ (black circles). Also shown are the density $n \in \mathcal{M}$ (blue square) and the PCA approximation to tangent space of $\mathcal{M}_{\mathrm{N}}$ at n, $\boldsymbol{T}^{\mathrm{PCA}}(n)$ (dashed line). This tangent plane is a local approximation to $\mathcal{M}_{\mathrm{N}}$.

$$\bar{n}(x)=\frac{1}{\Omega}\sum_{j=1}^{N_T}\omega_j n_j(x) \tag{26}$$

This generalized average is weighted by the function $\omega(\|n-n'\|)$ that only depends on the distance from $n(x)$ to $n'(x)$, $\omega_j=\omega(\|n-n_j\|)$, and $\Omega=\sum_{j=1}^{N_T}\omega_j$. Note that $n'(x)$ refers to the density n' evaluated at x and not the derivative of n with respect to x.

The locality of the method comes from the choice of ω. For standard PCA, the choice is $\omega(r)=\theta(R-r)$, where θ is the Heaviside function, and R is the distance from n' to the m-th nearest training density. This equally weights the nearest m training densities, and ignores all other training densities. This choice was used in Ref. [33]. Here, we choose a slightly smoother weighting function:

$$\omega(r)=(1-r/R)\theta(R-r) \tag{27}$$

Next, PCA is performed by spectral analysis of the empirical covariance operator,[71] based on the weighted average value around $n(x)$. We define the centered neighborhood by $\tilde{n}_j(x)=n_j(x)-\bar{n}(x)$. In this problem, densities are represented on a grid with $N_G = 500$ points, so let $\boldsymbol{n}=(n(x_1),\ldots,n(x_{N_G}))^\top$ be the vector representation of $n(x)$. The covariance matrix $\Gamma \in \mathbb{R}^{N_G\times N_G}$ is

$$\Gamma=\frac{1}{\Omega}\sum_{j=1}^{N_T}\omega_j \boldsymbol{n}_j \boldsymbol{n}_j^\top, \tag{28}$$

with eigen decomposition

$$\Gamma \boldsymbol{u}_j=\lambda_j \boldsymbol{u}_j. \tag{29}$$

The eigenvalues are ordered such that $\lambda_j > \lambda_{j+1}$. The eigenvectors $\boldsymbol{u}_j$ are called principal components (PCs), and give the directions of maximum variance in the data. We define the variance lost in keeping d PCs as $\eta=1-\sum_{j=1}^{d}\lambda_j/\sum_{j=1}^{N_G}\lambda_j$. In this case, there is little to no variance in directions orthogonal to the tangent space of $\mathcal{M}_N$, and maximum variance in directions aligned with the tangent space. Thus, the first d PCs form a basis for the tangent space, where d is the dimensionality of the density manifold (and tangent space). The projection operator onto this basis is:

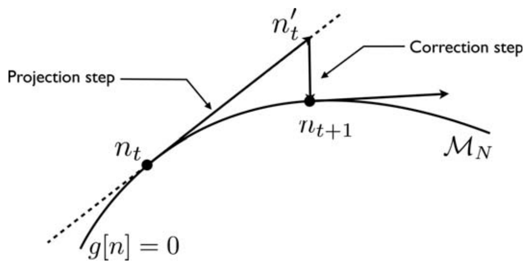

Figure 12. Schematic of the projected gradient descent. The functional derivative is projected onto the tangent space of the data manifold $\mathcal{M}_{\mathrm{N}}$ at n_{t} (dashed line). Next, a step is taken along the projected functional derivative to $n_t^{'}$ in the direction of lower energy. Finally, $g[n]$ is minimized orthogonal to the tangent space to ensure the minimization stays on $\mathcal{M}_{\mathrm{N}}$.

$$P[n]=\sum_{j=1}^{d}\boldsymbol{u}_j\boldsymbol{u}_j^\top. \tag{30}$$

The tangent space using PCA is given by

$$\boldsymbol{T}^{\mathrm{PCA}}[n]=\{\boldsymbol{n}\,|\,(1-P[n])(\boldsymbol{n}-\bar{\boldsymbol{n}})=0\}. \tag{31}$$

Finally, we choose the PCA approximation to the constraint $g[n]$ in eq. (25) as the squared distance from $\boldsymbol{n}$ to the tangent plane $\boldsymbol{T}^{\mathrm{PCA}}[\boldsymbol{n}]$:

$$g^{\mathrm{PCA}}[n]=\|(1-P[n])\tilde{\boldsymbol{n}}\|^2. \tag{32}$$

The PCA approximate density manifold $\mathcal{M}^{\mathrm{PCA}}$ is then defined implicitly by $g^{\mathrm{PCA}}[n]=0$. The process is illustrated in

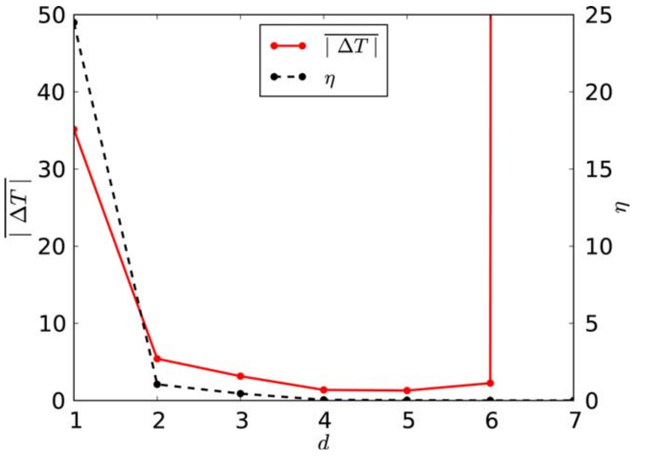

Figure 13. The MAE (in kcal/mol), $\overline{|\Delta T|}=\overline{|T^{\mathrm{ML}}[\tilde{n}]-T[n]|}$, evaluated on 100 constrained optimal densities compared with the variance lost η as a function of the number of PCs d in the PCA projection, with $m = 20$ nearest neighbors.

Table 4. The error in the KE in kcal/mol evaluated on constrained optimal densities using 100 densities for testing, with $N = 1$ for (a) $N_T = 40$ and (b) $N_T = 100$.

(a)

d/*m*	10	20	30
2	12 (98)	15 (100)	24 (100)
3	12 (100)	16 (100)	22 (100)
4	12 (98)	15 (100)	25 (100)
5	23,000 (18)	130 (27)	(0)

(b)

d/*m*	10	20	30	40
3	4.1 (99)	3.2 (100)	2.7 (99)	2.8 (100)
4	1.7 (100)	1.4 (100)	1.4 (100)	1.7 (100)
5	1.6 (100)	1.3 (100)	1.5 (100)	2.0 (100)
6	1.7 (93)	2.1 (100)	1.7 (100)	2.2 (100)

The percentage of converged optimal densities is given in parentheses. Here, *m* is the number of nearest neighbor densities used in PCA and *d* is number of PCs used in the projection.

Figure 11. In the next section, we develop a projected gradient descent method to solve eq. (25).

Projected gradient descent algorithm

For a given ML approximation to the KE functional,

$$E^{\mathrm{ML}}[n]=T^{\mathrm{ML}}[n]+V[n], \tag{33}$$

the algorithm to minimize the functional in eq. (25) to find a constrained optimal density is as follows (see Fig. 12). Choose an initial guess for the density, $n_0 \in \mathcal{M}_N$ (e.g., a training density):

1. Evaluate the functional derivative

$$\frac{\delta E^{\mathrm{ML}}[n]}{\delta n(x)}=\frac{\delta T_s^{\mathrm{ML}}[n]}{\delta n(x)}+v(x). \tag{34}$$

at $n = n_t$.

2. Compute the local PCA projection operator $P[n_t]$ from eq. (30).
3. Project the functional derivative onto the tangent space (see Fig. 12), and take a step:

$$n_t^{'}(x)=n_t(x)-\epsilon\hat{P}[n_t]\frac{\delta E^{\mathrm{ML}}[n]}{\delta n(x)}\bigg|_{n=n_t}, \tag{35}$$

where ϵ is a constant such that $0 < \epsilon \leq 1$. If convergence is unstable, reduce ϵ, trading stability for speed of convergence.

4. To ensure the constraint remains satisfied, we subtract the (weighted) mean of the training densities in the local neighborhood:

$$n_{t+1}(x)=n_t'(x)-(1-\hat{P}[n_t'])(n_t'-\bar{n}[n_t']). \tag{36}$$

We iterate these steps until convergence is achieved. We measure convergence by setting a maximum iteration step and tolerance threshold. If the total energy difference is smaller than the tolerance within the maximum number of iteration steps, the density is converged. If no solution is found, ϵ is reduced.

Errors on constrained optimal densities

With this new constrained minimization procedure via a projected gradient descent, we solve for the constrained optimal density for each test sample. We report the errors in the total energy and KE relative to the exact density in Table 3. In general, we expect these errors to be worse than the MLA evaluated on exact densities—by roughly a factor of 10. However, errors on constrained optimal densities decrease at the same rate with more training data, so an accuracy of 1 kcal/mol in KE is achieved with 150 training samples for $N = 1$, now on constrained optimal densities. Additionally, errors are of similar magnitude for multiple particles. In the last row of Table 3, we combine the training data from each *N* (100 training densities per *N* value) into one model. This combined MLA gives roughly the same error as each individual model. This is because, due to the locality of the Gaussian kernel, the training densities from each *N* are well separated (orthogonal in feature space) and the individual models are unaffected.

In the projected gradient descent, there are two PCA parameters that must be chosen: *m*, the number of nearest neighbors and *d*, the number of PCs to form the projection operator. Figure 13 shows the MAE evaluated by the constrained optimal density and variance lost as a function of the number of PCs *d* with $m = 20$. The MAE decreases initially as *d* increases as more PCs capture the local structure of the density manifold. As can be seen, $d = 4$ or 5 gives an optimal reconstruction of the tangent space of the manifold. As *d* increases further, the noise that was removed is re-introduced into the projection, causing the gradient descent algorithm to fail. For $d = 7$, many of the constrained searches do not converge. Table 4 reports the errors of the model evaluated on constrained optimal densities for $N_T = 40$ and $N_T = 100$, giving a rough optimization of the PCA parameters. Although the potential, which generates $\mathcal{M}_N$ has 9 parameters in this case, we observe that the optimal choice of *d* is only 4. This is because the data used to build the model is only a small fraction of $\mathcal{M}_N$. If we do not sample all relevant directions on $\mathcal{M}_N$, then the model cannot learn the functional derivative in those directions. The PCA projection will compensate by removing those directions. Thus, the effectiveness of our method depends on the sampling on the manifold.

Conclusion

In this work, we have explored in much greater detail the methods presented in Ref. [33], in which ML methods were used to directly approximate the KE of a quantum system as a functional of the electron density, and used this functional in a modified orbital-free DFT to obtain highly accurate self-consistent densities and energies.

We used a simple model as a proof of principle, to investigate how physical information about a system can be incorporated into the MLA. In particular, we tested a variety of standard kernels used in ML, and have found that the Gaussian kernel gives the lowest errors (the Cauchy kernel also achieves similar performance), showing that these two kernels capture the physics of the KE functional for our prototype system. All cross validation schemes that were tested gave similar predictions of hyperparameters that achieved low generalization error on the test set. Thus the standard methods for finding these parameters appear robust for this problem. Our results highlight the importance of an appropriate choice of kernel, as some of the kernels tested gave strikingly bad performance. With the construction of the L^2 norm that was used in the kernels, the method is basis set independent (as long as a complete basis is used). However, the ML method is capable of learning accurate KEs using a sparse grid (i.e., an incomplete basis). Using a sparse representation for the density without losing accuracy further speeds up calculations.

Many types of MLA can produce highly accurate KE estimates for exact densities. However, this is not sufficient for use in self-consistent calculations. We explained the origin of the noise in the functional derivative and developed a constrained search over the density manifold via a modified Euler equation, effectively projecting out the noise. We also introduced a local approximation to the manifold using PCA, and solved for constrained optimal densities using a projected gradient descent algorithm. This shows that an effective search for ground state densities does not require complete information of the whole electronic density domain. As the global minimum (i.e., the ground-state density) lies on the training density manifold, it is sufficient to restrict the self-consistent procedure to this limited domain. This stabilizes the optimization and drastically reduces the dimensionality of the problem, making the entire procedure possible. This worked well for our prototype system, yielding highly accurate constrained optimal energies and densities.

These results suggest that it might be possible to further extend the HK theorem. That theorem proves a one-to-one correspondence (under well-defined conditions) between the real-space one-electron density and potential. Nevertheless, this work suggests that this might be the continuum limit of a statement for finite representations of each. Under limited circumstances, it might be possible to prove that a finite representation of the density is sufficient to yield the energy within certain limits and a finite representation of the one-body potential. Work exploring this possibility is currently underway.

This work is motivated by the goal of applying ML to orbital-free DFT for large molecules. The practical contribution of this paper is to document a systematic scheme of MLA. All the procedures described and tested above depend only on distances between densities in the feature space. Although the analysis in this paper is performed on a prototype system in 1d, as long as a representation of density is well-defined [eq. (13)], there is no reason in principle that the same conceptual scheme would fail for more complicated systems in 3d. Of course, for a large system in 3d, there are up to $3N-6$ degrees of freedom, and sampling such a manifold uniformly in each dimension is not possible. But the true underlying dimensionality is not that of the coordinate space, but the complexity of situations encountered in any one simulation, and the variation in the potentials. If an accurate representation can be found that, for example, recognizes that a water molecule in an MD simulation differs only slightly from one configuration to another, the computational cost should be very affordable. Work along these lines is ongoing.

Acknowledgment

KRM thanks the BK21 Plus Program by NRF Korea, DFG and the Einstein Foundation.

Appendix A: Contour Plot of Functional-Driven Error

Figure A1 shows contour plots of the MAE as a function of the global parameters (noise level λ and kernel parameter θ or σ) for the wave and power kernels. Unlike the RBF kernels, the contours are not bell shaped around the minimum. The wave kernel requires a high noise level to regulate the fitting, indicating that the wave kernel cannot capture the structure of

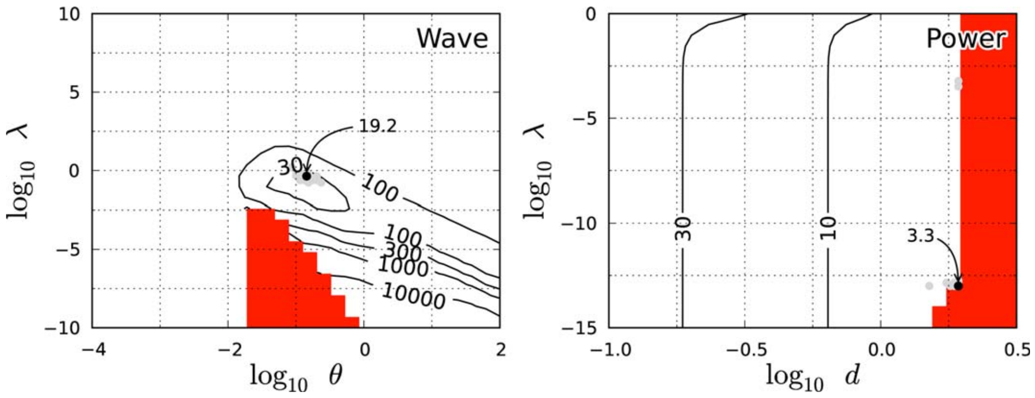

Figure A1. Contour plots of $\log_{10}$ of the mean of the absolute functional-driven error $|\Delta T_F|$ (in kcal/mol) over the test set as a function of the global parameters, λ and σ, for selected kernels with $N_T = 100$. The labels give the value of the contour in kcal/mol. Each gray dot gives the optimal choice of global parameters from a randomized 10-fold cross validation. The black dot denotes the median over 40 repetitions. The red shaded area in the contour plot shows when the inverse of the kernel matrix becomes ill-conditioned due to the limited numerical precision of our calculation.

Table 5. Optimal global parameters and functional-driven errors (mean absolute, standard deviation, and max absolute in kcal/mol) as a function of electron number N and number of training densities N_T with the Gaussian kernel.

N	N_T	$\lambda \cdot 10^{14}$	σ	$\overline{\lvert\Delta T_F\rvert}$	$\lvert\Delta T_F\rvert^{std}$	$\lvert\Delta T_F\rvert^{max}$
Centered						
1	40	37	1.8	1.6	2.9	24
1	60	0.14	0.9	0.51	0.8	9.5
1	80	0.99	1.2	0.27	0.43	3.6
1	100	0.12	0.8	0.21	0.36	4.2
1	150	0.05	1.2	0.06	0.11	0.90
Uncentered						
1	40	49	4.2	1.9	3.5	30
1	60	10	1.8	0.62	1.0	10
1	80	54	1.4	0.23	0.37	3.1
1	100	4.5	1.6	0.13	0.25	3.4
1	150	1.2	1.3	0.06	0.12	1.0
1	200	1.3	1.0	0.03	0.06	0.88
2	60	60	3.0	0.44	0.67	5.0
3	60	6.0	5.8	0.56	0.87	5.8
4	60	0.55	14	0.59	0.93	6.2
2	100	1.0	2.2	0.13	0.24	1.6
3	100	1.9	2.5	0.12	0.22	1.5
4	100	1.4	2.7	0.07	0.14	2.2
1−4	400	1.7	2.2	0.12	0.23	3.0

Gaussian $k[n,n']=\exp\left(-\frac{\|n-n'\|^2}{2\sigma^2}\right)$.

the data. The error is quite bad (19.2 kcal/mol with optimal choice of global parameters), even with 100 training densities. The power kernel gives a reasonable noise level and the MAE decreases as d increases. The red shaded area in the contour plot shows when the inverse of the kernel matrix becomes ill-conditioned due to the limited numerical precision of our calculation.

Appendix B: Model Performance with Kernels

Tables 5–10 give the performance of the Gaussian, Cauchy, Laplacian, wave, power, and linear kernels (including uncentered results for RBF kernels) for various training set sizes. The parameters are optimized via 40 repetitions of 10-fold cross validation. We report the mean, standard deviation, and max

Table 6. Optimal global parameters and functional-driven errors (mean absolute, standard deviation, and max absolute in kcal/mol) as a function of electron number N and number of training densities N_T with the Cauchy kernel.

Cauchy $k[n,n']=\frac{1}{1+\|n-n'\|^2/\sigma^2}$

N	N_T	$\lambda \cdot 10^{14}$	σ	$\overline{\lvert\Delta T_F\rvert}$	$\lvert\Delta T_F\rvert^{std}$	$\lvert\Delta T_F\rvert^{max}$
Uncentered						
1	40	3058	2.9	1.4	2.4	20
1	60	18	2.4	0.35	0.62	7.1
1	80	31	3.8	0.21	0.35	2.6
1	100	7.8	3.5	0.13	0.23	2.9
1	150	1.2	3.4	0.05	0.11	1.4
2	100	1.0	4.2	0.11	0.19	1.2
3	100	1.1	4.0	0.10	0.18	1.3
4	100	1.4	5.6	0.06	0.12	1.7
1−4	400	1.8	4.2	0.09	0.18	2.3

Table 7. Optimal global parameters and functional-driven errors (mean absolute, standard deviation, and max absolute in kcal/mol) as a function of electron number N and number of training densities N_T with the Laplacian kernel.

Laplacian $k[n,n']=\exp\left(-\frac{\|n-n'\|}{2\sigma}\right)$

N	N_T	$\lambda \cdot 10^{16}$	$\sigma \cdot 10^{-5}$	$\overline{\lvert\Delta T_F\rvert}$	$\lvert\Delta T_F\rvert^{std}$	$\lvert\Delta T_F\rvert^{max}$
Centered						
1	40	10	7.2	10	22	231
1	60	10	7.2	8.7	20	230
1	80	10	13	6.6	18	230
1	100	10	3.6	6.4	18	231
1	150	10	3.6	4.7	16	223
1	200	10	0.3	4.5	16	222
2	100	10	1.7	5.0	16	220
3	100	10	6.2	4.8	11	129
4	100	10	1.7	3.5	8.7	104
1−4	400	10	0.28	18	38	574
Uncentered						
1	40	5.9	276	10	22	237
1	60	6.9	233	11	21	231
1	80	6.4	168	7.4	18	232
1	100	6.9	144	6.9	18	229
1	150	8.8	3.9	4.7	16	222

Table 8. Optimal global parameters and functional-driven errors (mean absolute, standard deviation, and max absolute in kcal/mol) as a function of electron number N and number of training densities N_T with the wave kernel.

Wave $k[n,n']=\frac{\theta}{\|n-n'\|}\sin\frac{\|n-n'\|}{\theta}$

N	N_T	λ	$\theta \cdot 10^2$	$\overline{\lvert\Delta T_F\rvert}$	$\lvert\Delta T_F\rvert^{std}$	$\lvert\Delta T_F\rvert^{max}$
Centered						
1	40	0.18	15	18	39	622
1	60	0.45	15	17	36	650
1	80	0.45	15	21	32	255
1	100	0.45	15	19	31	252
1	150	0.45	15	17	28	232
1	200	0.45	15	16	26	228
2	100	0.27	13	9.8	19	208
3	100	0.45	9.1	10	18	168
4	100	0.45	5.7	7.7	12	111
1−4	400	0.27	129	308	554	7056
Uncentered						
1	40	0.37	9.6	28	48	433
1	60	0.50	9.4	26	44	385
1	80	0.63	8.8	26	45	412
1	100	0.80	9.0	25	45	414
1	150	0.95	8.7	24	42	411

Table 9. Optimal global parameters and functional-driven errors (mean absolute, standard deviation, and max absolute in kcal/mol) as a function of electron number N and number of training densities N_T with the linear kernel.

$k[n,n']=\langle n,n'\rangle+c$

N	N_T	λ	$c \cdot 10^{-4}$	$\overline{\lvert\Delta T_F\rvert}$	$\lvert\Delta T_F\rvert^{std}$	$\lvert\Delta T_F\rvert^{max}$
1	40	0.04	1.8	56	74	396
1	60	1.3	3.3	53	70	385
1	80	0.81	4.0	52	69	397
1	100	0.62	6.0	53	69	376
1	150	0.23	6.0	53	69	387

Table 10. Optimal global parameters and functional-driven errors (mean absolute, standard deviation, and max absolute in kcal/mol) as a function of electron number N and number of training densities N_T with the power kernel.

Power $k[n, n'] = -\|n - n'\|^d$

N	N_T	$\lambda \cdot 10^{14}$	d	$\overline{\lvert\Delta T_F\rvert}$	$\lvert\Delta T_F\rvert^{\text{std}}$	$\lvert\Delta T_F\rvert^{\max}$
Centered						
1	40	10	2.0	5.2	10	117
1	60	10	2.0	4.3	8.1	77
1	80	10	2.0	3.4	7.4	90
1	100	10	2.0	3.3	8.0	104
1	150	10	2.0	2.5	5.9	79
1	200	10	2.0	2.3	5.8	77
2	100	10	2.0	2.3	5.6	72
3	100	10	2.0	1.8	4.0	49
4	100	10	2.0	1.6	3.4	37
1−4	400	321	2.0	4.5	7.7	94
Uncentered						
N	N_T	Λ	d	$\overline{\lvert\Delta T_F\rvert}$	$\lvert\Delta T_F\rvert^{\text{std}}$	$\lvert\Delta T_F\rvert^{\max}$
1	40	6.9×10^{11}	6.3	3400	3400	4000
1	60	3.2×10^{17}	3.0	3400	3400	4000
1	80	4.6×10^{21}	7.0	3400	3400	4000
1	100	$8.3\ 10^{27}$	3.7	3400	3400	4000
1	150	$2.4\ 10^{14}$	2.3	3400	3400	4000
1	200	$2.6 \cdot 10^{15}$	2.3	3400	3400	4000
1	300	$8.2 \cdot 10^{27}$	14	3400	3400	4000
1	400	$1.3 \cdot 10^{8}$	1.0	3400	3400	4000

Table 11. Optimal global parameters and functional-driven errors (mean absolute, standard deviation, and max absolute in kcal/mol) as a function of grid points N_G and number of training densities N_T with the Gaussian kernel.

N_G	N_T	$\lambda \times 10^{14}$	σ	$\overline{\lvert\Delta T_F\rvert}$	$\lvert\Delta T_F\rvert^{\text{std}}$	$\lvert\Delta T_F\rvert^{\max}$
500	40	0.27	13	2.4	4.8	41
250	40	0.27	15	2.2	4.2	37
100	40	0.23	16	2.2	4.2	37
50	40	0.36	16	2.0	3.8	34
25	40	0.17	23	1.8	3.2	27
20	40	0.17	23	1.9	3.4	29
10	40	0.23	15	3.0	5.1	34
500	60	0.35	2.4	0.76	1.2	12
250	60	0.42	3.1	0.90	1.4	13
100	60	0.40	2.7	0.81	1.3	12
50	60	0.39	3.0	0.88	1.4	13
25	60	0.15	3.4	0.92	1.5	12
20	60	0.27	3.8	0.66	1.1	8.8
10	60	0.54	5.6	1.4	3.4	42
500	80	19	1.9	0.23	0.37	2.7
250	80	10	1.9	0.22	0.36	2.8
100	80	11	1.7	0.22	0.36	3.1
50	80	15	1.7	0.22	0.36	3.0
25	80	6.0	2.1	0.21	0.34	2.3
20	80	5.4	2.4	0.20	0.34	3.4
10	80	3.8	1.7	1.1	1.7	11
500	100	1.5	1.9	0.131 (1.39)	0.259 (2.65)	3.49 (15.7)
250	100	2.0	1.9	0.130 (1.41)	0.261 (2.67)	3.43 (15.9)
100	100	1.6	1.9	0.130 (1.39)	0.258 (2.64)	3.48 (15.7)
50	100	2.5	1.9	0.131 (1.42)	0.258 (2.69)	3.40 (16.0)
25	100	0.65	2.1	0.129 (1.42)	0.259 (2.71)	3.62 (16.2)
20	100	0.16	2.5	0.179 (1.14)	0.232 (1.71)	2.61 (9.32)
10	100	3.5	2.2	1.08 (8200)	1.86 (22,000)	12.0 (66,000)

Self-consistent results for $N_T = 100$ are shown in brackets.

of the absolute functional-driven errors $|\Delta T_F| = |T^{\text{ML}}[n] - T[n]|$, evaluated on exact densities. Self-consistent results are not included.

Appendix C: Sparse Grid

Table 11 shows the effect of using a sparse grid to represent the density on the ML approximation for the KE functional (using the Gaussian kernel), for various training set sizes $N_T = 40, 60, 80, 100$ and number of grid points $N_G = 10, 20, 25, 50, 100, 250, 500$. In all cases, the MAE barely fluctuates with N_G until about 10. When N_G is <100, the optimal length scale of the kernel adjusts to compensate for the inaccurate approximation to the integral in the L^2 inner product.

Keywords: density functional theory · machine learning · orbital free · kinetic energy functional · self-consistent calculation

How to cite this article: L. Li, J. C. Snyder, I. M. Pelaschier, J. Huang, U.-N. Niranjan, P. Duncan, M., Rupp, K.-R. Müller, K. Burke. *Int. J. Quantum Chem.* **2016**, *116*, 819–833. DOI: 10.1002/qua.25040

Additional Supporting Information may be found in the online version of this article.

[1] P. A. M. Dirac, *Proc. R. Soc. London Ser A* **1929**, *123*, 714.
[2] W. Kohn, *Rev. Mod. Phys.* **1999**, *71*, 1253.
[3] P. Hohenberg, W. Kohn, *Phys. Rev. B* **1964**, *136*, 864.
[4] R. M. Dreizler, E. K. U. Gross, *Density Functional Theory: An Approach to the Quantum Many-Body Problem;* Springer, Springer-Verlag Berlin Heidelberg, **1990**.
[5] L. H. Thomas, *Mathematical Proceedings of the Cambridge Philosophical Society*, Vol. *23*; Cambridge Univ Press, **1927**; pp. 542–548.
[6] E. Fermi, *Z Phys.* **1928**, *48*, 73.
[7] E. Teller, *Rev. Mod. Phys.* **1962**, *34*.
[8] W. Kohn, L. J. Sham, *Phys. Rev.* **1965**, *140*, A1133. URL http://link.aps.org/doi/10.1103/PhysRev.140.A1133.
[9] A. D. Becke, *Phys. Rev. A* **1988**, *38*, 3098. URL http://link.aps.org/doi/10.1103/PhysRevA.38.3098.
[10] C. Lee, W. Yang, R. G. Parr, *Phys. Rev. B* **1988**, *37*, 785. URL http://link.aps.org/doi/10.1103/PhysRevB.37.785.
[11] J. P. Perdew, K. Burke, M. Ernzerhof, *Phys. Rev. Lett.* **1996**, *77*, 3865. URL http://link.aps.org/doi/10.1103/PhysRevLett.77.3865.
[12] V. V. Karasiev, R. S. Jones, S. B. Trickey, F. E. Harris, In *New Developments in Quantum Chemistry*; J. L. Paz, A. J. Hernández, Eds.; Transworld Research Network: Kerala, India, **2009**; pp. 25–54.
[13] V. Karasiev, S. Trickey, *Comp. Phys. Commun.* **2012**, *183*, 2519. ISSN 0010-4655, URL http://www.sciencedirect.com/science/article/pii/S0010465512002287.
[14] F. Tran, T. A. Wesolowski, *Int. J. Quantum Chem.* **2002**, *89*, 441. ISSN 1097-461X, URL http://dx.doi.org/10.1002/qua.10306.
[15] S. Mohr, L. E. Ratcliff, L. Genovese, D. Caliste, P. Boulanger, S. Goedecker, T. Deutsch, *Phys. Chem. Chem. Phys.* (in press). URL http://dx.doi.org/10.1039/C5CP00437C.
[16] R. Z. Khaliullin, J. VandeVondele, J. Hutter, *J. Chem. Theory Comput.* **2013**, *9*, 4421. http://dx.doi.org/10.1021/ct400595k, URL http://dx.doi.org/10.1021/ct400595k.
[17] R. Baer, D. Neuhauser, E. Rabani, *Phys. Rev. Lett.* **2013**, *111*, 106402. URL http://link.aps.org/doi/10.1103/PhysRevLett.111.106402.
[18] L. Hung, E. A. Carter, *Chem. Phys. Lett.* **2009**, *475*, 163.

[19] M. Hodak, W. Lu, J. Bernholc, *J. Chem. Phys.* **2008**, *128*, 014101.

[20] J. C. Snyder, M. Rupp, K. Hansen, L. Blooston, K. R. Müller, K. Burke, *J. Chem. Phys.* **2013**, *139*, 224104. URL http://scitation.aip.org/content/aip/journal/jcp/139/22/10.1063/1.4834075.

[21] C. F. von Weizsäcker, *Z. Phys.* **1935**, *96*, 431. ISSN 0044-3328, URL http://dx.doi.org/10.1007/BF01337700.

[22] Y. Wang, E. Carter, In *Theoretical Methods in Condensed Phase Chemistry*; S. Schwartz, Ed.; Kluwer: NY, **2000**.

[23] V. Karasiev, D. Chakraborty, S. Trickey, In *Many-Electron Approaches in Physics, Chemistry, and Mathematics*; L. D. Site, V. Bach, Eds.; Springer Verlag: Kluwer, NY, **2014**.

[24] V. V. Karasiev, R. S. Jones, S. B. Trickey, F. E. Harris, *Phys. Rev. B* **2009**, *80*, 245120. URL http://link.aps.org/doi/10.1103/PhysRevB.80.245120.

[25] V. Karasiev, R. Jones, S. Trickey, F. E. Harris, *Phys. Rev. B* **2013**, *87*, 239903(E). URL http://link.aps.org/doi/10.1103/PhysRevB.87.239903.

[26] E. Chacón, J. E. Alvarellos, P. Tarazona, *Phys. Rev. B* **1985**, *32*, 7868. URL http://link.aps.org/doi/10.1103/PhysRevB.32.7868.

[27] P. García-González, J. E. Alvarellos, E. Chacón, *Phys. Rev. B* **1996**, *53*, 9509. URL http://link.aps.org/doi/10.1103/PhysRevB.53.9509.

[28] P. García-González, J. E. Alvarellos, E. Chacón, *Phys. Rev. B* **1998**, *57*, 4857. URL http://link.aps.org/doi/10.1103/PhysRevB.57.4857.

[29] L. W. Wang, M. P. Teter, *Phys. Rev. B* **1992**, *45*, 13196. URL http://link.aps.org/doi/10.1103/PhysRevB.45.13196.

[30] Y. A. Wang, N. Govind, E. A. Carter, *Phys. Rev. B* **1999**, *60*, 16350. URL http://link.aps.org/doi/10.1103/PhysRevB.60.16350.

[31] J. Xia, C. Huang, I. Shin, E. A. Carter, *J. Chem. Phys.* **2012**, *136*, 084102. pages 13) URL http://link.aip.org/link/?JCP/136/084102/1.

[32] K. Vu, J. C. Snyder, L. Li, M. Rupp, B. F. Chen, T. Khelif, K.–R. Müller, K. Burke, *Int. J. Quantum Chem*. **2015**, *115*, 1115.

[33] J. C. Snyder, M. Rupp, K. Hansen, K. R. Müller, K. Burke, *Phys. Rev. Lett.* **2012**, *108*, 253002. URL http://link.aps.org/doi/10.1103/PhysRevLett.108.253002.

[34] K. R. Müller, S. Mika, G. Rätsch, K. Tsuda, B. Schölkopf, *IEEE Trans. Neural Network* **2001**, *12*, 181.

[35] I. Kononenko, *Artif. Intell. Med.* **2001**, *23*, 89.

[36] S. Lemm, B. Blankertz, T. Dickhaus, K. R. Müller, *Neuroimage* **2011**, *56*, 387.

[37] T. Joachims, *Text categorization with support vector machines: Learning with many relevant features;* Springer, Springer Berlin Heidelberg, **1998**.

[38] S. Dumais, J. Platt, D. Heckerman, M. Sahami, In *Proceedings of the seventh international conference on Information and knowledge management*, ACM, **1998**; pp. 148–155.

[39] M. Rupp, A. Tkatchenko, K. R. Müller, O. von Lilienfeld, *Phys. Rev. Lett.* **2012**, *108*, 058301. URL http://link.aps.org/doi/10.1103/PhysRevLett.108.058301.

[40] K. Hansen, G. Montavon, F. Biegler, S. Fazli, M. Rupp, M. Scheffler, O. A. von Lilienfeld, A. Tkatchenko, K. R. Müller, *J. Chem. Theory Comput.* **2013**, *9*, 3404. http://pubs.acs.org/doi/pdf/10.1021/ct400195d, URL http://pubs.acs.org/doi/abs/10.1021/ct400195d.

[41] G. Montavon, M. Rupp, V. Gobre, A. Vazquez-Mayagoitia, K. Hansen, A. Tkatchenko, K. R. Müller, O. A. von Lilienfeld, *N. J. Phys.* **2013**, *15*, 095003. URL http://stacks.iop.org/1367-0000/15/i=9/a=095003.

[42] Z. D. Pozun, K. Hansen, D. Sheppard, M. Rupp, K. R. Müller, G. Henkelman, *J. Chem. Phys.* **2012**, *136*, 174101. pages 8) URL http://link.aip.org/link/?JCP/136/174101/1.

[43] A. P. Bartók, M. C. Payne, R. Kondor, G. Csányi, *Phys. Rev. Lett.* **2010**, *104*, 136403. URL http://link.aps.org/doi/10.1103/PhysRevLett.104.136403.

[44] T. L. Fletcher, S. J. Davie, P. L. Popelier, *J. Chem. Theory Comput.* **2014**, *10*, 3708.

[45] M. J. Mills, P. L. Popelier, *J. Chem. Theory Comput.* **2014**, *10*, 3840.

[46] R. T. McGibbon, V. S. Pande, *J. Chem. Theory Comput.* **2013**, *9*, 2900.

[47] R. Fournier, S. Orel, *J. Chem. Phys.* **2013**, *139*, 234110.

[48] J. Behler, M. Parrinello, *Phys. Rev. Lett.* **2007**, *98*, 146401. URL http://link.aps.org/doi/10.1103/PhysRevLett.98.146401.

[49] K. T. Schütt, H. Glawe, F. Brockherde, A. Sanna, K. R. Müller, E. K. U. Gross, *Phys. Rev. B* **2014**, *89*, 205118.

[50] K. Hansen, F. Biegler, R. Ramakrishnan, W. Pronobis, O. A. von Lilienfeld, K.-R. Müller, A. Tkatchenko, *J. Phys. Chem. Lett*. **2015**, *6*, 2326.

[51] E. Hairer, P. Nørsett, P. S. Paul, G. Wanner, *Solving ordinary differential equations I: Nonstiff problems;* Springer: New York, **1993**.

[52] M. C. Kim, E. Sim, K. Burke, *Phys. Rev. Lett.* **2013**, *111*, 073003. URL http://link.aps.org/doi/10.1103/PhysRevLett.111.073003.

[53] E. H. Lieb, Inequalities; Springer, Springer-Verlag Berlin Heidelberg, **2002**; pp. 269–303.

[54] J. A. Lee, M. Verleysen, *Nonlinear dimensionality reduction;* Springer Science & Business Media, Springer-Verlag New York, **2007**.

[55] V. Vapnik, *The nature of statistical learning theory;* Springer Verlag: New York, **1995**.

[56] B. Scholkopf, S. Mika, C. Burges, P. Knirsch, K. R. Müller, G. Ratsch, A. Smola, *IEEE Trans. Neural Netw.* **1999**, *10*, 1000. ISSN 1045-9227.

[57] G. Montavon, M. Braun, T. Krueger, K. R. Müller, *IEEE Signal Process. Mag.* **2013**, *30*, 62. ISSN 1053-5888.

[58] B. Schölkopf, A. Smola, *Learning with Kernels;* MIT Press, Cambridge, **2002**.

[59] T. Hofmann, B. Schölkopf, A. Smola, *Ann Stat* **2008**, *36*, 1171.

[60] N. Aronszajn, *Trans. Am. Math. Soc.* **1950**, *68*, 337.

[61] B. Schölkopf, A. Smola, K. Müller, *Neural Comput.* **1998**, *10*, 1299.

[62] T. Hastie, R. Tibshirani, J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed.; Springer: New York, **2009**.

[63] M. Rupp, E. Proschak, G. Schneider, *J. Chem. Inf. Model* **2007**, *47*, 2280.

[64] A. Smola, B. Schölkopf, K. Müller, *Neural Netw.* **1998**, *11*, 637.

[65] A. Zien, G. Rätsch, S. Mika, B. Schölkopf, T. Lengauer, K. R. Müller, *Bioinformatics* **2000**, *16*, 799.

[66] M. Braun, J. Buhmann, K. R. Müller, *J. Machine Learn. Res.* **2008**, *9*, 1875.

[67] T. Poggio, S. Mukherjee, R. Rifkin, A. Rakhlin, A. Verri, Tech. Rep. AI Memo 2001-011, CBCL Memo 198, Massachusetts Institute of Technology, **2001**.

[68] S. I. Amari, N. Murata, K. R. Müller, M. Finke, H. Yang, *IEEE Trans. Neural Netw.* **1997**, *8*, 985.

[69] J. C. Snyder, S. Mika, K. Burke, and K.-R. Müller, In *Empirical Inference - Festschrift in Honor of Vladimir N. Vapnik*; B. Schoelkopf, Z. Luo, V. Vovk, Eds.; Springer: Heidelberg, **2013**.

[70] J. C. Snyder, M. Rupp, K.-R. Müller, K. Burke *Int. J. Quantum Chem.* **2015**, *115*, 1102.

[71] C. Bregler, S. M. Omohundro, *Surface learning with applications to lipreading;* International Computer Science Institute, San Francisco, CA. Morgan Kaufmann, **1994**.

Received: 31 August 2015
Revised: 29 September 2015
Accepted: 15 October 2015
Published online 17 November 2015