Learning molecular energies using localized graph kernels

Grégoire Ferré, Terry Haut, and Kipton Barros

View Online   Export Citation   CrossMark

Learning molecular energies using localized graph kernels

Grégoire Ferré,[1,2] Terry Haut,[3] and Kipton Barros[2]

[1]*Université Paris-Est, CERMICS (ENPC), F-77455 Marne-la-Vallée, France*
[2]*Theoretical Division and CNLS, Los Alamos National Laboratory, Los Alamos, New Mexico 87545, USA*
[3]*Computational Physics and Methods Group (CCS-2), Los Alamos National Laboratory, Los Alamos, New Mexico 87545, USA*

Recent machine learning methods make it possible to model potential energy of atomic configurations with chemical-level accuracy (as calculated from *ab initio* calculations) and at speeds suitable for molecular dynamics simulation. Best performance is achieved when the known physical constraints are encoded in the machine learning models. For example, the atomic energy is invariant under global translations and rotations; it is also invariant to permutations of same-species atoms. Although simple to state, these symmetries are complicated to encode into machine learning algorithms. In this paper, we present a machine learning approach based on graph theory that naturally incorporates translation, rotation, and permutation symmetries. Specifically, we use a *random walk graph kernel* to measure the similarity of two adjacency matrices, each of which represents a local atomic environment. This Graph Approximated Energy (GRAPE) approach is flexible and admits many possible extensions. We benchmark a simple version of GRAPE by predicting atomization energies on a standard dataset of organic molecules. *Published by AIP Publishing.* [http://dx.doi.org/10.1063/1.4978623]

I. INTRODUCTION

There is a large and growing interest in applying machine learning methods to infer molecular properties and potential energies[1–6] based upon data generated from *ab initio* quantum calculations. Steady improvement in the accuracy, transferability, and computational efficiency of the machine learned models has been achieved. A crucial enabler, and ongoing challenge, is incorporating prior physical knowledge into the methodology. In particular, we know that potential energies should be invariant with respect to translations and rotations of atomic positions, and also with respect to permutations of identical atoms. The search for new techniques that naturally encode physical invariances is of continuing interest.

A common modeling approach is to decompose the energy of an atomic configuration as a sum of local contributions, and to represent each local atomic environment in terms of descriptors (i.e., features) that are inherently rotation and permutation invariant.[7,8] Such descriptors may be employed in, e.g., linear regression[9] or neural network models.[10–14] Recently, sophisticated descriptors have been proposed based on an expansion of invariant polynomials[15] and, inspired by convolutional neural networks, on a cascade of multiscale wavelet transformations.[16,17]

Kernel learning is an alternative approach that can avoid direct use of descriptors.[18,19] Instead, one specifies a kernel that measures similarity between two inputs and, ideally, directly satisfies known invariances. An advantage of kernel methods is that the accuracy can systematically improve by adding new configurations to the dataset. A disadvantage is that the computational cost to build a kernel regression model typically scales cubically with dataset size. Kernel methods that model potential energies include Gaussian Approximation Potentials (GAP),[20] Smooth Overlap of Atomic Potentials

(SOAP),[6,8] methods based upon eigenvalues of the Coulomb matrix,[21] Bag-of-Bonds,[22] and others.[4,23–25] However, encoding the physical invariances may impose challenges. For example, to achieve rotational invariance in GAP and SOAP, one requires explicit integration over the relative rotation of two configurations. To achieve permutation invariance in methods based upon the Coulomb matrix, one commonly restricts attention to sorted descriptors (e.g., matrix eigenvalues or Bag-of-Bonds), thus introducing singularities in the regression model, and potentially also sacrificing discriminative power.

Here, we present the Graph Approximated Energy (GRAPE) framework. GRAPE is based on two central ideas: we represent local atomic environments as weighted graphs based upon pairwise distances, for which rotation invariance is inherent, and we compare local atomic environments by leveraging well known graph kernel methods invariant with respect to node permutation. There is much flexibility in implementing these ideas. In this paper, we define the weighted adjacency matrix elements (i.e., the edge weights) through an analogy with SOAP,[8] and we use generalized random walk graph kernels[26–28] to define similarity between graphs. Graph kernels have commonly been employed to compare molecules and predict their properties,[28–36] but it appears that prior applications to energy regression are limited.[37] We demonstrate with benchmarks that GRAPE shows promise for energy regression tasks.

The rest of the paper is organized as follows. Section II reviews necessary background techniques: kernel ridge regression of energies, the SOAP kernel, and random walk graph kernels. We merge these tools to obtain our GRAPE kernel, presented in Sec. III. Finally, we demonstrate the performance of GRAPE on a standard molecular database in Sec. IV and summarize our results in Sec. V.

II. REVIEW OF RELEVANT METHODS

A. Kernel modeling of potential energy

Here we review the machine learning technique of kernel ridge regression and its application to modeling potential energy landscapes. We assume a dataset $(x_i, y_i)_{i=1...N}$, where x_i represents an atomic configuration and $y_i = E(x_i)$ its corresponding energy. The goal is to build a statistical approximation $\hat{E}$ that accurately estimates the energy of new configurations.

A common approach begins with *descriptors* (also called features or order parameters) $(h_m)_{m=1...M}$, designed to capture various aspects of a configuration x. Often $h_m(x)$ describes the geometry of a local atomic environment. Many descriptors suitable for modeling potential energy have been developed.[8] The descriptors can be used, for example, with simple linear regression,[9]

$$\hat{E}(x) = \sum_{m=1}^{M} \beta_m h_m(x), \qquad (1)$$

with learned regression coefficients β_m, or as inputs to a more complicated model such as a neural network.[13]

Kernel methods are an alternative machine learning approach and avoid direct use of descriptors. Instead, the starting point is a kernel $K(x, x')$ that measures the similarity between configurations x and x'. We use kernel ridge regression, for which the approximated energy reads

$$\begin{cases} \hat{E}(x) = \sum_{i=1}^{N} \hat{\alpha}_i K(x_i, x), \\ \hat{\alpha} = (K + \lambda I)^{-1} y, \end{cases} \qquad (2)$$

where $I \in \mathbb{R}^{N \times N}$ is the identity matrix, $K \in \mathbb{R}^{N \times N}$ denotes a matrix with elements $K(x_i, x_j)$, and $y = (y_1, y_2, \ldots, y_N)$ are the energies in the dataset. The parameter $\lambda > 0$ serves to regularize the model, and prevents overfitting by penalizing high frequencies.[18]

Equation (2) can be readily derived from the linear model, (1), using least squares error minimization with a ridge penalty; see Appendix A for details. The kernel becomes $K(x, x') = \sum_{m=1}^{M} h_m(x) h_m(x')$, and thus satisfies the criteria of symmetry and positive semi-definiteness. Conversely, any function K that is symmetric and positive semi-definite can be decomposed as above, where M may be infinite.[38] Thus, $(h_m)_{m=1...M}$ and K are formally interchangeable. An advantage of specifying the kernel directly is that the corresponding, potentially infinite set of descriptors may remain fully implicit.

In the context of modeling potential energies and forces, physical locality is often a good approximation.[6] We assume that the energy of an atomic configuration x_i with n_i atoms can be decomposed as a sum of local contributions,

$$E(x_i) = \sum_{\ell=1}^{n_i} \varepsilon(\tilde{x}_{i,\ell}), \qquad (3)$$

where $\tilde{x}_{i,\ell}$ denotes a local view of environment x_i centered on atom ℓ. The index ℓ iterates through all n_i atoms of the atomic configuration x_i, and we denote by $Q = \sum_{i=1}^{N} n_i$ the total number of local environments in the database. If long-range interactions exist, e.g., induced by electrostatics, they should be treated separately from (3).

The key idea is that statistical regression should begin with the local energy ε, which we formally decompose as in (1). Then, after a series of straightforward steps (Appendix B), we find[6]

$$K(x_i, x_{i'}) = \sum_{\ell=1}^{n_i} \sum_{\ell'=1}^{n_{i'}} \tilde{K}(\tilde{x}_{i,\ell}, \tilde{x}'_{i',\ell'}), \qquad (4)$$

and the localized kernel regression finally reads for a configuration x,

$$\begin{cases} \hat{E}(x) = \sum_{i=1}^{N} \hat{\alpha}_i \sum_{\ell=1}^{n_i} \sum_{\ell'=1}^{n} \tilde{K}(\tilde{x}_{i,\ell}, \tilde{x}_{\ell'}), \\ \hat{\alpha} = (L\tilde{K}L^T + \lambda I)^{-1} y, \end{cases} \qquad (5)$$

where the entries of the matrix $\tilde{K} \in \mathbb{R}^{Q \times Q}$ are $\tilde{K}(x_{i,\ell}, x_{i',\ell'})$ (i.e., the similarity between local environments ℓ and ℓ' in configurations i and i'), and $L \in \mathbb{R}^{N \times Q}$ is such that $L_{i,\ell} = 1$ if the local environment ℓ belongs to configuration i and 0 otherwise.

The important conclusion is that the kernel $\tilde{K}$ for *local* configurations confers a kernel K for total configurations. This is a mathematical consequence of the energy decomposition, which encodes locality into our statistical model of global energies. Note that the energies ε of the local atomic configurations $\tilde{x}_\ell$ need not be explicitly learned. The relatively small size of $\tilde{x}_\ell$ improves computational efficiency, and generally enhances transferability of the model.

To summarize, Eqs. (4) and (5) provide the recipe for kernel ridge regression of energy. The rest of the paper discusses approaches to designing the local kernel $\tilde{K}$.

B. SOAP kernel

Here we review the powerful SOAP method[6,8] for regression of potential energy landscapes. The structure of the SOAP kernel will inspire our design of GRAPE, which we present in Sec. III.

We begin by representing the local configuration $\tilde{x}_\ell$ with $\tilde{n}_\ell$ neighbors as a smooth atomic density centered on atom ℓ,

$$\rho_{\tilde{x}_\ell}(\mathbf{r}) = \sum_{i=1}^{\tilde{n}_\ell} \omega(z_i) f_c(|\mathbf{r}_{i\ell}|) \varphi_\sigma(|\mathbf{r} - \mathbf{r}_{i\ell}|). \qquad (6)$$

Here $\mathbf{r}_{i\ell} = \mathbf{r}_i - \mathbf{r}_\ell$, with $\mathbf{r}_i$ the position of the ith atom in $\tilde{x}_\ell$ and z_i its atomic number with associated weight $\omega(z_i)$. There is flexibility is choosing the coefficient $\omega(z_i)$, provided that it is an injective function of the atomic number z_i for atom i. We also define a Gaussian function to smooth the atomic densities,

$$\varphi_\sigma(r) = \exp(-r^2/2\sigma^2), \qquad (7)$$

and a cutoff function,

$$f_c(r) = \begin{cases} \frac{1}{2} \left[1 + \cos(r\pi/R_{cut}) \right] & r \leq R_{cut}, \\ 0 & r \geq R_{cut}. \end{cases} \qquad (8)$$

The parameter $\sigma > 0$ sets the smoothing length scale, and $R_{cut} > 0$ the cutoff distance. Atom i contributes to $\rho_{\tilde{x}_\ell}(\mathbf{r})$ only if sufficiently close to atom ℓ, namely, if $|\mathbf{r}_{i\ell}| < R_{cut}$. We define $\tilde{x}_\ell$ to include only those atoms in x which satisfy this condition, yielding a substantial numerical speedup for large atomic configurations. In the following, for notational convenience, we

suppress ℓ indices without loss of generality. Namely, we consider two local atomic environments, $\tilde{x}$ and $\tilde{x}'$, centered on atomic positions $\mathbf{r}_\ell$ and $\mathbf{r}'_{\ell'}$.

The next step in building the SOAP kernel is to define a scalar product between local atomic densities,

$$S(\tilde{x}, \tilde{x}') = \int_{\mathbb{R}^3} \rho_{\tilde{x}}(\mathbf{r}) \rho_{\tilde{x}'}(\mathbf{r}) \, d\mathbf{r}. \tag{9}$$

The density fields $\rho_{\tilde{x}}$ and $\rho_{\tilde{x}'}$ are naturally invariant to permutations of atomic indices, and S inherits this symmetry. Translation invariance is a consequence of comparing local environments. However, S is not invariant with respect to rotations $\mathbf{r} \mapsto R\mathbf{r}$. To include this symmetry, one may integrate over rotations,

$$k(\tilde{x}, \tilde{x}') = \int_{SO(3)} S(R\tilde{x}, \tilde{x}')^p \, dR, \tag{10}$$

where dR is the Haar measure and $R\tilde{x}$ generates the configuration with density $\rho_{\tilde{x}}(R\mathbf{r})$. Numerical evaluation of (10) is non-obvious, and the SOAP approach involves expanding $\rho_{\tilde{x}}(\mathbf{r})$ in terms of spherical harmonics and applying orthogonality of the Wigner matrices.[8] In principle one could select integer $p > 1$ as an arbitrary parameter, but in practice one typically fixes $p = 2$ to simplify the expansion of spherical harmonics.

The final local SOAP kernel is defined by rescaling,

$$\tilde{K}(\tilde{x}, \tilde{x}') = \left[\frac{k(\tilde{x}, \tilde{x}')}{\sqrt{k(\tilde{x}, \tilde{x})k(\tilde{x}', \tilde{x}')}} \right]^\zeta, \tag{11}$$

for $\zeta > 0$, which is possible because the unscaled kernel is strictly positive for nonempty environments. After this rescaling, we have $0 \leq \tilde{K}(\tilde{x}, \tilde{x}') \leq 1$ and $\tilde{K}(\tilde{x}, \tilde{x}) = 1$. One commonly selects $\zeta > 1$ to effectively amplify the kernel in regions where $k(\tilde{x}, \tilde{x}')$ is largest.

By Eq. (4), the local kernel $\tilde{K}$ produces a total kernel K that satisfies permutation, translation, and rotation invariance. The hyperparameters are σ, R_{cut}, p, and ζ.

C. Random walk graph kernels

Here we present random walk graph kernels.[26,28] In particular, the exponential graph kernel reviewed below will be the basis of GRAPE, our method for kernel regression of potential energies (cf. Sec. II A). Our use of graphs to represent local atomic environments is inspired by applications in chemical informatics.[32,34]

A graph $G = (V, E)$ is defined as a set of vertices $V = (v_i)_{i=1\ldots n}$ and edges $E \subseteq V \times V$ that connect pairs of vertices. An unweighted graph is equivalently represented by its adjacency matrix $A \in \mathbb{R}^{n \times n}$, where $A_{ij} = 1$ if vertices i and j are connected by an edge [i.e., if $(v_i, v_j) \in E$] and $A_{ij} = 0$ otherwise.

We work with weighted graphs, such that each edge $(v_i, v_j) \in E$ is assigned a nonzero weight w_{ij}. We represent such graphs via the weighted adjacency matrix,

$$A_{ij} = \begin{cases} w_{ij} & \text{if } (v_i, v_j) \in E, \\ 0 & \text{otherwise.} \end{cases} \tag{12}$$

We consider undirected and unlabeled graphs, which means that $A_{ij} = A_{ji}$ and the nodes and edges have no additional structure.

Graph kernels measure similarity between two graphs. Many graph kernels exist, including Laplacian kernels,[30] shortest path kernels,[31] skew spectrum kernels,[39] graphlet kernels,[40] and functional graph kernels.[41,42]

Here we focus on random walk graph kernels because they are simple, computationally efficient,[28] and suitable for learning potential energy landscapes, as will be made clear later. The motivating idea is to consider a random walk over graph vertices (i.e., states). For this purpose, we assume for now that A represents the transition matrix of a Markov process, such that $0 \leq A_{ij} \leq 1$ is the probability for a random walker to transition from state j to state i, and the columns of A conserve probability, $\sum_{i=1}^n A_{ij} = 1$. Later we will relax these probabilistic constraints and allow arbitrary $A_{ij} > 0$. We select the initial state according to a distribution $p \in \mathbb{R}^n$, i.e., the walker starts at state i with probability $0 \leq p_i \leq 1$. Then $(A^k p)_i$ is the probability that the walker reaches state i after k successful steps in the Markov chain. However, the walker may not reach k steps. Analogous to p, we assume a stopping distribution $q \in \mathbb{R}^n$. That is, before each step, a walker in state i stops with probability $0 \leq q_i \leq 1$. Then

$$s_k = \sum_{i=1}^n q_i (A^k p)_i = q^T A^k p \tag{13}$$

represents the probability that a random walker stops after k steps.

Our goal is to compare two graphs represented by adjacency matrices $A \in \mathbb{R}^{n \times n}$ and $A' \in \mathbb{R}^{n' \times n'}$. For this, we consider simultaneous random walks, one on each graph. The joint probability that the first walker transitions from $j \to i$ and the second walker transitions from $j' \to i'$ is the product of individual transition probabilities. These joint probabilities appear as the elements of $A_\times \in \mathbb{R}^{nn' \times nn'}$, the direct (Kronecker) product of adjacency matrices, defined by

$$A_\times = A \otimes A' = \begin{pmatrix} A_{1,1}A' & \cdots & A_{1,n}A' \\ \vdots & & \vdots \\ A_{n,1}A' & \cdots & A_{n,n}A' \end{pmatrix}. \tag{14}$$

Note that $A_\times$ can itself be interpreted as an adjacency matrix for the so-called direct product graph,[43] see Fig. 3.

Given starting p, p' and stopping q, q' distributions, the probability that both walkers simultaneously stop after k steps (13) is

$$s_k s'_k = (q^T \otimes q'^T)(A \otimes A')^k (p \otimes p') = q_\times^T A_\times^k p_\times, \tag{15}$$

where $p_\times = p \otimes p'$, $q_\times = q \otimes q'$, and we have used the matrix product identity $(A \otimes B)(C \otimes D) = (AC) \otimes (BD)$.

Two graphs can be compared by forming a weighted sum over paths of all lengths of the Markov chain. The random walk graph kernel finally reads

$$K(A, A') = \sum_{k \geq 0} \mu_k s_k s'_k = q_\times^T \left(\sum_{k \geq 0} \mu_k A_\times^k \right) p_\times. \tag{16}$$

The choice of $(\mu_k)_{k \geq 0}$ is left to the user as long as the series converges, and in this case (16) is known to define a positive semi-definite kernel.[28]

One must also select the starting and stopping distributions. Choosing p, p', q, q' to be uniform guarantees invariance with respect to permutation of vertices. To see this,

consider permutation of the nodes, represented by permutation matrices T and T', which transform the adjacency matrices as $A \to TAT^{-1}$ and $A' \to T'A'T'^{-1}$. The product matrix transforms as $A_\times \to (T \otimes T')A_\times(T \otimes T')^{-1}$, where $(T \otimes T')^{-1} = (T^{-1} \otimes T'^{-1})$. Consequently, the kernel (16) becomes $\hat{q}_\times^T \left(\sum_{\ell \geq 0} \mu_\ell A_\times^\ell \right) \hat{p}_\times$, where $\hat{p}_\times = (T^{-1}p) \otimes (T'^{-1}p')$ and $\hat{q}_\times = (qT) \otimes (q'T')$. If the starting and stopping distributions are uniform, we observe that $\hat{p}_\times = p_\times$ and $\hat{q}_\times = q_\times$, and the kernel exhibits permutation invariance.

Note, however, that with uniform starting and stopping distributions, it is crucial that the columns of adjacency matrices A and A' *not* be normalized. If they were, then we would have $q^T A^k p = 1$ independently of A, making it impossible to compare graphs. Consequently, although we continue to use (16), we abandon its probabilistic interpretation.

In our work, we select $\mu_k = \gamma^k/k!$ in (16) to get the exponential graph kernel,[26]

$$K(A, A') = q_\times^T e^{\gamma A_\times} p_\times, \tag{17}$$

where γ is a parameter controlling how fast the powers of $A_\times$ go to zero. It can be shown to reweight the eigenvalues in the comparison process.[28,41]

The direct product matrix $A_\times$ contains $n^2 n'^2$ matrix elements, a potentially large number, but the numerical cost of evaluating the kernel can be reduced by diagonalizing the factor matrices. Indeed, given $A = PDP^{-1}$ and $A' = P'D'P'^{-1}$, with D and D' diagonal, we write

$$A \otimes A' = (PDP^{-1}) \otimes (P'D'P'^{-1}) \tag{18}$$

$$= (P \otimes P')(D \otimes D')(P \otimes P')^{-1}, \tag{19}$$

so that (17) becomes

$$K(A, A') = (q^T P \otimes q'^T P')e^{\gamma D \otimes D'}(P^{-1}p \otimes P'^{-1}p'). \tag{20}$$

Because $D \otimes D'$ is itself a diagonal matrix, we can evaluate $\exp(\gamma D \otimes D')$ by applying the exponential to each of the nn' diagonal elements.

The computational cost to calculate $K(A, A')$ is dominated by matrix diagonalization, which scales like $\mathcal{O}(n^3)$ assuming $n \sim n'$. If the eigen decompositions of A and A' have been precomputed, the scaling reduces to $\mathcal{O}(n^2)$.

III. GRAPH APPROXIMATED ENERGY (GRAPE)

The main contribution of our work is GRAPE, a random walk graph kernel tailored to local energy regression. Since random walk graph kernels act on the direct product of adjacency matrices, our primary task is to select a suitable adjacency matrix associated with a local atomic environment $\tilde{x}$. To motivate our choice of adjacency matrix, we first present a graph interpretation of the SOAP kernel.

The SOAP kernel is constructed from the inner product, Eq. (9), between local atomic densities (6). Combining these equations, we obtain a double sum over atoms from different local environments,

$$S(\tilde{x}, \tilde{x}') = \sum_{i=1}^{\tilde{n}} \sum_{i'=1}^{\tilde{n}'} \omega(z_i)\omega(z_{i'})$$
$$\times f_c(|\mathbf{r}_i|)f_c(|\mathbf{r}'_{i'}|)\phi_\sigma(|\mathbf{r}_i - \mathbf{r}'_{i'}|), \tag{21}$$

where ϕ_σ is again Gaussian,

$$\phi_\sigma(|\mathbf{r}_i - \mathbf{r}'_{i'}|) = \int_{\mathbb{R}^3} \varphi_\sigma(|\mathbf{r} - \mathbf{r}_i|)\varphi_\sigma(|\mathbf{r} - \mathbf{r}'_{i'}|) \, d\mathbf{r}$$
$$= \sigma^3 \pi^{3/2} \varphi_{\sqrt{2}\sigma} \left(|\mathbf{r}_i - \mathbf{r}'_{i'}| \right). \tag{22}$$

Here, for notational convenience, the positions of the atoms $\mathbf{r}_i$ and $\mathbf{r}'_{i'}$ are given relative to the centers $\mathbf{r}_\ell$ and $\mathbf{r}'_{\ell'}$ of the local environments.

Seeking a graph interpretation of (21), we rewrite it in matrix form,

$$\begin{cases} S(\tilde{x}, \tilde{x}') = \sum_{i=1}^{\tilde{n}} \sum_{i'=1}^{\tilde{n}'} B_{ii'}^2 = \text{Tr}(B^T B), \\ B_{ii'} = \sqrt{\omega(z_i)\omega(z_i')f_c(|\mathbf{r}_i|)f_c(|\mathbf{r}'_{i'}|)\phi_\sigma(|\mathbf{r}_i - \mathbf{r}'_{i'}|)}. \end{cases} \tag{23}$$

The object $B \in \mathbb{R}^{\tilde{n} \times \tilde{n}'}$ is suggestive of an adjacency matrix. However, the entries $B_{ii'}$ of this matrix refer to atoms belonging to the distinct configurations $\tilde{x}$ and $\tilde{x}'$. Indeed, the graphical interpretation of B would be bipartite (i.e., containing only cross-links between the atoms of $\tilde{x}$ and $\tilde{x}'$) as illustrated in Fig. 1. Note that the elements $B_{ii'}$ vary with the relative rotation between $\tilde{x}$ and $\tilde{x}'$, and indeed SOAP requires nontrivial integration (10) to achieve rotational invariance.

In GRAPE, we achieve rotational invariance by representing local configurations as graphs. Guided by the form of (23), we introduce an analogous adjacency matrix $A \in \mathbb{R}^{\tilde{n} \times \tilde{n}}$ that operates on a *single* local environment $\tilde{x}$,

$$A_{ij} = \sqrt{\omega(z_i)\omega(z_j)f_c(|\mathbf{r}_i|)f_c(|\mathbf{r}_j|)\phi_\sigma(|\mathbf{r}_i - \mathbf{r}_j|)}. \tag{24}$$

For example, Fig. 2 illustrates the graph representation of a methane molecule. The elements A_{ij} of the adjacency matrix (24) link atoms i and j, both contained in $\tilde{x}$. Because these elements depend only on pairwise distances, they are manifestly rotation invariant.
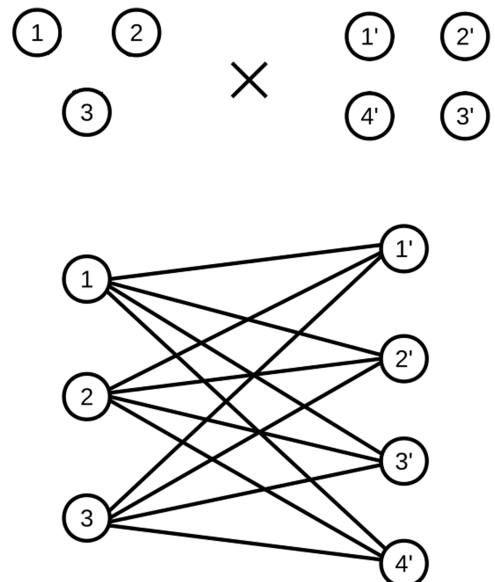

FIG. 1. Graph interpretation of the SOAP kernel. Two local environments comprised of 3 and 4 atoms are compared by forming a bipartite graph that links pairs of atoms.
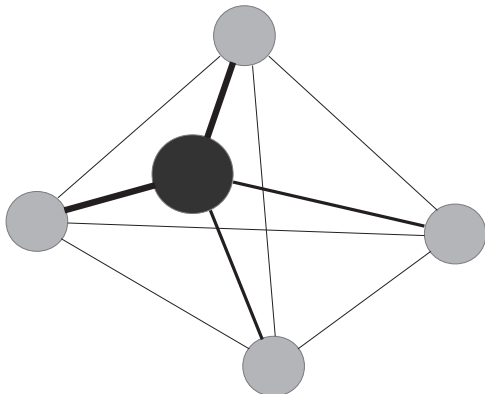
FIG. 2. Representation of methane CH_4 as a graph. Edge thickness between atoms i and j represents the adjacency matrix element A_{ij}.

To compare two local atomic environments, we apply the exponential graph kernel of Sec. II C,

$$k(\tilde{x}, \tilde{x}') = (q^T \otimes q'^T)e^{\gamma A \otimes A'}(p \otimes p'). \quad (25)$$

Figure 3 illustrates the graph represented by the direct product matrix, $A \otimes A'$, and should be contrasted with Fig. 1. We again use uniform starting and stopping probability distributions p, p', q, q' to ensure invariance with respect to permutation of indices, while rotational invariance is naturally inherited from the adjacency matrices A and A'.

Finally, we use (11) to rescale (25) and (5) to build our model which is suitable for regression on total energies. The kernel is regular and its derivatives are calculable in closed form (Appendix D), so atomic forces are also available from the regression model.

GRAPE shares most of its hyperparameters with SOAP, namely, σ, σ_{cut}, and ζ. However, the SOAP hyperparameter p is replaced by GRAPE's γ. Note that γ enables effective
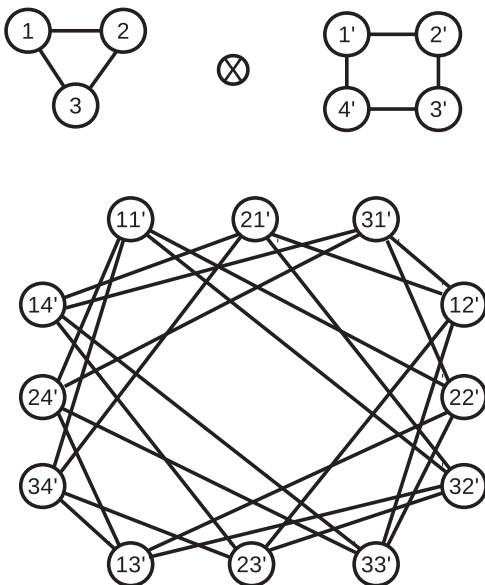

FIG. 3. Environments with 3 and 4 atoms are graphically represented by adjacency matrices A and A'. The direct product graph has 3×4 vertices and adjacency matrix elements $(A \otimes A')_{ij,kl} = A_{ik}A'_{jl}$. Random walk graph kernels operate on this direct product matrix.

reweighting of the eigenvalues, each of which corresponds to a different length scale.[28,41,42] For all methods considered in this work, we must also specify the regularization parameter λ used in kernel ridge regression (5).

The computational cost to directly evaluate the local GRAPE kernel $k(\tilde{x}, \tilde{x}')$ scales as $C_{loc} \sim \tilde{n}^3$ (see Sec. II C), where $\tilde{n}$ is the typical number of atoms in local environments $\tilde{x}$ and $\tilde{x}'$. Global energy regression (5) requires many local kernel evaluations, for both SOAP and GRAPE methods. Evaluating the total kernel (4) requires double summation over all $n \sim n'$ atoms in total configurations x and x', and thus scales as $\mathcal{O}(n^2 C_{loc})$. If there are N atomic configurations $(x_i)_{i=1...N}$ in the dataset, then evaluating the full kernel matrix K scales as $\mathcal{O}(N^2 n^2 C_{loc})$. Matrix inversion brings the total scaling to $\mathcal{O}(N^2 n^2 C_{loc} + N^3)$. An improvement is to precompute the diagonalization of every local adjacency matrix at cost $\mathcal{O}(Nn\tilde{n}^3)$. Then subsequent local kernel evaluations scale like $C_{loc} \sim \tilde{n}^2$, and the total cost of building the GRAPE energy regression model becomes $\mathcal{O}(Nn\tilde{n}^3 + N^2 n^2 \tilde{n}^2 + N^3)$. Once the model is built, computing the approximation energy (5) for a new configuration scales as $\mathcal{O}(n\tilde{n}^3 + Nn^2 \tilde{n}^2)$.

IV. BENCHMARK

We demonstrate the competitiveness of GRAPE by benchmarking it on a standard energy regression problem. We use the QM7 dataset of organic molecules used in Ref. 21 and freely available at http://quantum-machine.org/datasets/. This database contains 7165 molecules randomly selected from the GDB-13 database with associated atomization energies, typically between −2000 and −800 kcal/mol, obtained from hybrid DFT calculations.[44,45] GDB-13 contains approximately 10^9 organic molecules up to 23 atoms in size, and formed from elements H, C, N, O, and S. QM7 has been randomly partitioned into 5 sequences, each containing 1433 molecules. We use the first N molecules of partition 1 as training data (with $N = 100$, 300, 500, 1000), and partitions 2 and 3 as validation data, i.e., to select the hyperparameters necessary for the various regression methods (discussed below). Finally, after having locked the hyperparameters, we use partition 4 as our test data, from which we estimate the mean absolute error (MAE) and root mean square error (RMSE) of the energy regression models.

We compare the performance of GRAPE against that of the Coulomb matrix method[21] (Appendix C) and SOAP.[6] After

TABLE I. Mean Absolute Error (MAE) and Root Mean Square Error (RMSE), in units of kcal/mol, for Coulomb, SOAP and GRAPE regression methods, and various sizes N of the training dataset.

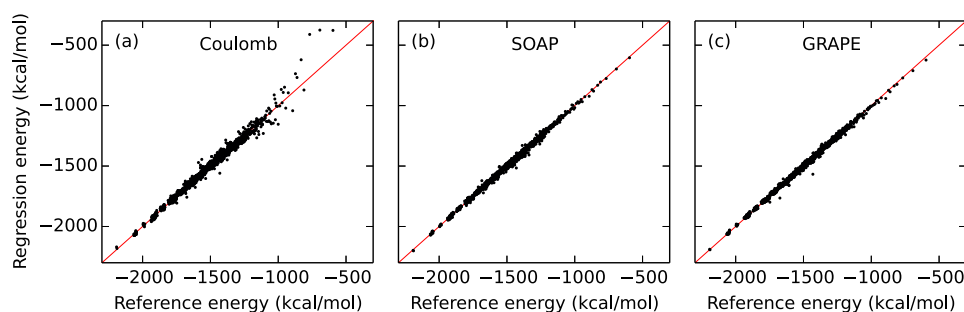|	N	100	300	500	1000
Coulomb	MAE	25.6	19.8	17.9	17.7
	RMSE	50.8	33.5	27.1	28.5
SOAP	MAE	15.6	11.3	10.4	9.7
	RMSE	21.0	15.6	14.5	13.3
GRAPE	MAE	11.2	10.1	9.6	9.0
	RMSE	14.9	13.9	13.3	12.7

FIG. 4. Atomization energies predicted by (a) Coulomb, (b) SOAP, and (c) GRAPE methods. The three regression models were built using a training dataset of $N = 1000$ organic molecules. Reference energies were calculated using density functional theory.

experimenting on the validation data, we selected the following hyperparameters for these methods. We set the Gaussian width parameter in (7) to be $\sigma = 1.0$ Å for both SOAP and GRAPE. This provides a small overlap of the Gaussian densities of bonded atoms, which are typically separated by a few Å. We select the cutoff radius in (8) to be $R_{cut} = 4.0$ Å for both SOAP and GRAPE, such that a typical local atomic neighborhood contains around 5 atoms. Interestingly, we find that GRAPE is especially sensitive to this hyperparameter, and that the energy regression error would nearly double if we instead selected $R_{cut} = 5.0$ Å for GRAPE. We must also select the weight $\omega(z_i)$ in (6) and (24) as a function of atomic number z_i. Again, after some experimentation, we select $\omega(z_i) = z_i$ for SOAP and $\omega(z_i) = 1/z_i$ for GRAPE. The latter implies that the GRAPE hyperparameter γ appearing in (25) should scale like the square of the typical atomic number in the data; we select $\gamma = 40.0$. Consistent with previous work,[8] we select $p = 2$ for SOAP (10) and $\zeta = 4.0$ in (11) for both SOAP and GRAPE. We select $\alpha = 0.05$ for the Coulomb hyperparameter appearing in Eq. (C4). The last hyperparameter λ, appearing in (5), regularizes the kernel regression. We select $\lambda = 10^{-7}$ for Coulomb, $\lambda = 10^{-5}$ for SOAP, and $\lambda = 10^{-3}$ for GRAPE. In selecting the above hyperparameters, we put approximately equal weight on the MAE and RMSE. We caution that all three models (Coulomb, SOAP, GRAPE) would likely benefit from a more exhaustive search over the hyperparameters. Our results should thus be interpreted as a *qualitative* comparison of GRAPE's performance relative to Coulomb and SOAP.

Table I displays the MAE and RMSE error estimates for all three methods and various sizes N of the training dataset. Figure 4 shows regression energies for the test data with $N = 1000$, and we observe that GRAPE is competitive with SOAP; given our limited tuning of the hyperparameters, we cannot conclude that one method outperforms the other. However, both SOAP and GRAPE significantly and consistently outperform the Coulomb method. We note that our MAE and RMSE estimates for the Coulomb method are consistent with those previously reported,[21] when holding N fixed.

For reference, the best known methods in the literature outperform all three kernels in Table I. Reference 4 achieves an MAE of ≈ 6 kcal/mol for $N = 1000$ using a Laplacian kernel applied to a random sorting of the Coulomb matrix.[46] The Bag-of-Bonds kernel method[22] reduces the MAE to ≈ 3.3 for $N = 1000$. Neural networks appear to be state of the art for this QM7 dataset. When training on $N = 5800$ molecules, the Deep Tensor Neural Network[14] achieves

an MAE of ≈ 1.0 kcal/mol—a significant improvement over the Bag-of-Bonds MAE, ≈ 1.5 kcal/mol.

V. DISCUSSION

We introduced GRAPE, a simple and flexible method for energy regression based on random walk graph kernels. The approach is invariant with respect to translations, rotations, and permutations of same-species atoms. Using a standard benchmark dataset of organic molecules, we demonstrated that GRAPE's energy predictions are competitive with commonly applied kernel methods. Because the GRAPE kernel is regular and admits simple closed-form derivatives, it is a candidate for fitting atomic forces although we did not benchmark the method on this application.

Like the Coulomb method,[21] GRAPE essentially consists of comparing matrices. These matrices, built upon interatomic distances, are inherently invariant with respect to translation and rotation, while carrying complete information of the system. However, the matrix representation lacks permutation invariance. To restore permutation invariance, the Coulomb method restricts its attention to the list of ordered eigenvalues. Unfortunately, due to this sorting, the associated kernel becomes non-differentiable, which can be a source of instability, as pointed out in Ref. 17. Unlike Coulomb, GRAPE is completely regular.

An alternative machine learning approach is to begin by representing the atomic configuration as a density, e.g., using Gaussians peaked on each atomic position. This density field provides a natural way to compare two environments,[16,20,25] and this is the approach taken by the SOAP method.[8] A disadvantage of this approach is that the density is not inherently rotational invariant, and density-based kernels typically require an explicit integration over rotations. An achievement of the SOAP kernel is that its rotational integral, Eq. (10), can be evaluated in closed form via a spherical harmonic expansion of the Gaussian densities. GRAPE obviates the need for such an integral in the first place.

An exciting aspect of the GRAPE approach is its flexibility for future extensions. For example, one can use multiscale analysis techniques, such as the multiscale Laplacian graph kernel,[47] or wavelet and scattering transforms over graphs.[35,48,49] Another promising research direction is to encode more physical knowledge into the graph, e.g., using labeled graphs.[28] The vertex labels could directly represent

the atomic species or could more subtly encode chemical information such as the number of electrons in the valence shell.

ACKNOWLEDGMENTS

APPENDIX A: KERNEL RIDGE REGRESSION

Here we review the machine learning technique of kernel ridge regression. We assume a dataset of pairs $(x_i, y_i)_{i=1\ldots N}$. In our application, each x_i represents an atomic configuration and y_i its corresponding energy. The goal is to build a model of energies $\hat{E}(x)$ for configurations x not contained in the dataset. Following Ref. 50 we begin with ordinary ridge regression.

We formally model the energy as a linear combination of descriptors $(h_m)_{m=1\ldots M}$,

$$\hat{E}(x) = \sum_{m=1}^{M} \beta_m h_m(x), \tag{A1}$$

where M is possibly infinite. Note that $\hat{E}(x)$ is linear in the regression coefficients β_m but potentially very nonlinear in x. We select regression coefficients $\hat{\beta}_m$ that minimize a loss function

$$\sum_{i=1}^{N} V(y_i - \hat{E}(x_i)) + \frac{\lambda}{2} \sum_{m=1}^{M} \beta_m^2. \tag{A2}$$

The simplest error measure, $V(r) = r^2$, yields a linear system of equations to be solved for β_m. The solution is the well-known ridge regression model,[51]

$$\hat{E}(x) = \sum_{m=1}^{M} \hat{\beta}_m h_m(x), \tag{A3}$$

$$\hat{\beta} = (HH^T + \lambda I)^{-1} Hy, \tag{A4}$$

with matrix elements $H_{mi} = h_m(x_i)$ and vector $y = (y_1, y_2, \ldots, y_N)$. The empirical regularization parameter $\lambda > 0$ effectively smooths the approximation $\hat{E}$, and consequently improves the conditioning of the linear inversion problem.

It turns out that we can generalize this model by introducing the inner product kernel,

$$K(x, x') = \sum_{m=1}^{M} h_m(x) h_m(x'), \tag{A5}$$

which measures similarity between inputs x and x'. Note that

$$K_{ij} = K(x_i, x_j) = (H^T H)_{i,j}, \tag{A6}$$

$$K(x, x_i) = (h^T H)_i, \tag{A7}$$

where $h = (h_1(x), h_2(x), \ldots, h_M(x))$. The matrix identity $(HH^T + \lambda I)^{-1} H = H(H^T H + \lambda I)^{-1}$ allows us to rewrite (A3) and (A4) in the suggestive form,

$$\hat{E}(x) = \sum_{i=1}^{N} \hat{\alpha}_i K(x, x_i), \tag{A8}$$

$$\hat{\alpha} = (K + \lambda I)^{-1} y. \tag{A9}$$

The key insight is that the descriptors h_m no longer appear explicitly. One may select the kernel K directly, and thus implicitly define the family $(h_m)_{m=1\ldots M}$.

Equations (A8) and (A9) together with a suitable kernel constitute the method of kernel ridge regression. The kernel must be symmetric and positive, i.e., $K(x, x') = K(x', x)$ for all x, x', and $\sum_i \sum_{i'} \alpha_i \alpha_{i'} K(x_i, x_{i'}) \geq 0$ for any non-empty collection of points and coefficients $\{(x_i, \alpha_i)\}$. By Mercer's theorem,[38] these conditions are equivalent to the decomposition (A1).

There are multiple ways to interpret Eqs. (A8) and (A9). In statistical learning theory, we may derive $\hat{E}(x)$ as the function in some space $\mathcal{H}$ that best minimizes the squared error $\sum_{i=1}^{N} \left(y_i - \hat{E}(x_i)\right)^2$ subject to a regularization term $\lambda ||\hat{E}||_{\mathcal{H}}$. The kernel K generates both $\mathcal{H}$ (the so-called reproducing kernel Hilbert space) and its norm $|| \cdot ||_{\mathcal{H}}$.[19,52] Another interpretation is Bayesian. In this case, one views the coefficients β_m as independent random variables with Gaussian prior probabilities, then the kernel K specifies the covariance of y and y', and $\hat{E}(x)$ becomes the posterior expectation for configuration x. This approach is called Gaussian process regression or Kriging.[53]

We note that Eq. (A8) [but not (A9)] is independent of the specific choice of error measure V. For example, the method of support vector regression corresponds to the choice $V(r) = \max(0, |r| - \epsilon)$.

APPENDIX B: LOCAL ENERGY DECOMPOSITION

We assume that the energy of an atomic configuration x with n atoms can be decomposed as a sum over local atomic environments $\tilde{x}_\ell$,

$$E(x) = \sum_{\ell=1}^{n} \varepsilon(\tilde{x}_\ell). \tag{B1}$$

Following Eq. (A1), we model local energies $\varepsilon(\tilde{x}_\ell)$ as a linear combination of abstract descriptors $\tilde{h}_m(\tilde{x}_\ell)$,

$$\hat{\varepsilon}(\tilde{x}_\ell) = \sum_{m=1}^{M} \beta_m \tilde{h}_m(\tilde{x}_\ell). \tag{B2}$$

After inserting (B2) into (B1), we observe that the model for total energy,

$$\hat{E}(x) = \sum_{\ell=1}^{n} \hat{\varepsilon}(\tilde{x}_\ell) = \sum_{m=1}^{M} \beta_m h_m(x), \tag{B3}$$

involves the same coefficients β_m but new descriptors

$$h_m(x) = \sum_{\ell=1}^{n} \tilde{h}_m(\tilde{x}_\ell). \tag{B4}$$

We assume a dataset $(x_i, y_i)_{i=1\ldots N}$ containing configurations x_i and total energies

$$y_i = E(x_i). \tag{B5}$$

Again minimizing the cost function of Eq. (A2), the kernel ridge regression model of Eqs. (A8) and (A9) is unchanged.

However, the kernel between the full atomic configurations is now constrained. Equations (A5) and (B4) together imply that

$$K(x, x') = \sum_{\ell=1}^{n} \sum_{\ell'=1}^{n'} \tilde{K}(\tilde{x}_\ell, \tilde{x}'_{\ell'}), \tag{B6}$$

where $\tilde{K}(\tilde{x}_\ell, \tilde{x}'_{\ell'}) = \sum_m \tilde{h}_m(\tilde{x}_\ell)\tilde{h}_m(\tilde{x}'_{\ell'})$.

We conclude that the energy decomposition (B1) constrains the kernel K to a sum over terms $\tilde{K}$ involving *local* environments. In "kernelizing" this model, we only require specification of the local kernel $\tilde{K}$; the descriptors $\tilde{h}_m$ are implicit and typically infinite in number.

APPENDIX C: COULOMB MATRIX METHOD

We review the Coulomb matrix method of Ref. 21. Here, we elide localized kernel (5) and consider only global atomic configurations x as in Ref. 21. Given a configuration with positions and atomic numbers $(\mathbf{r}_i, z_i)_{i=1...N}$, the Coulomb matrix is

$$M_{i,j} = \begin{cases} \dfrac{z_i z_j}{\|r_i - r_j\|}, & i \neq j, \\[2mm] 0.5\, z_i^{2.4}, & i = j. \end{cases} \tag{C1}$$

Although the matrix is invariant with respect to translations and rotations of the atomic positions, it is not invariant with respect to permutations of indices. The matrix eigenvalues, however, are permutation invariant. To compare configurations x and x' (of size n and n', respectively), the Coulomb method compares the lists of eigenvalues (λ_i and λ'_i, respectively), sorted by decreasing magnitude and truncated at length $\min(n, n')$. We define normalized eigenvalues,

$$\hat{\lambda}_i = \lambda_i / \|\lambda\|, \tag{C2}$$

$$\hat{\lambda}'_i = \lambda'_i / \|\lambda'\|, \tag{C3}$$

where $\|\lambda\| = \sqrt{\sum_{i=1}^{n} \lambda_i^2}$. Then the Coulomb kernel reads

$$K(x, x') = \exp\left[-\alpha \sum_{i=1}^{\min(n,n')} (\hat{\lambda}_i - \hat{\lambda}'_i)^2\right], \tag{C4}$$

with α a hyperparameter.

APPENDIX D: COMPUTING FORCES

Given a regression model for energy, it is often desirable to compute its gradient to obtain a regression model for forces. One motivation is to use machine learned forces within molecular dynamics simulations. Another motivation is that typical datasets (e.g., as generated by *ab initio* calculations) often contain force information that can aid the energy regression.[6] In either case, the key step is to calculate the derivative $\partial_\mathbf{r} K(x, x')$ of the kernel $K(x, x')$ with respect to some atomic position $\mathbf{r}$ in configuration x. Here, we consider the exponential graph kernel (25) used in GRAPE. As before, A and A' denote the adjacency matrices of x and x', respectively. Assuming $\partial_\mathbf{r} A' = 0$, the gradient of the unscaled kernel is

$$\partial_\mathbf{r} k(x, x') = \gamma(q^T \otimes q'^T)(\partial_\mathbf{r} A \otimes A')e^{\gamma A \otimes A'}(p \otimes p'). \tag{D1}$$

Referring to Eq. (24) we see that the GRAPE adjacency matrix element A_{ij} has a simple functional dependence on positions $\mathbf{r}_i$ and $\mathbf{r}_j$ (and also an implicit dependence on the central atom position $\mathbf{r}_\ell$), and it is thus straightforward to express $\partial_\mathbf{r} A$ in a closed form. To compute the normalized kernel (11), one also needs

$$\partial_\mathbf{r} k(x, x) = 2\gamma(q^T \otimes q^T)(\partial_\mathbf{r} A \otimes A)e^{\gamma A \otimes A}(p \otimes p). \tag{D2}$$

[1] T. Ho and H. Rabitz, J. Chem. Phys. **104**, 2584 (1996).
[2] R. P. A. Bettens and M. A. Collins, J. Chem. Phys. **111**, 816 (1999).
[3] B. J. Braams and J. M. Bowman, Int. Rev. Phys. Chem. **28**, 577 (2009).
[4] K. Hansen, G. Montavon, F. Biegler, S. Fazli, M. Rupp, M. Scheffler, O. A. von Lilienfeld, A. Tkatchenko, and K.-R. Müller, J. Chem. Theory Comput. **9**, 3404 (2013).
[5] M. Rupp, R. Ramakrishnan, and O. A. von Lilienfeld, J. Phys. Chem. Lett. **6**, 3309 (2015).
[6] A. P. Bartók and G. Csányi, Int. J. Quantum Chem. **115**, 1051 (2015).
[7] J. Behler, J. Chem. Phys. **134**, 074106 (2011).
[8] A. P. Bartók, R. Kondor, and G. Csányi, Phys. Rev. B **87**, 184115 (2013).
[9] A. Thompson, L. Swiler, C. Trott, S. Foiles, and G. Tucker, J. Comput. Phys. **285**, 316 (2015).
[10] J. Behler and M. Parrinello, Phys. Rev. Lett. **98**, 146401 (2007).
[11] C. M. Handley and J. Behler, Eur. Phys. J. B **87**, 152 (2014).
[12] J. Behler, J. Phys.: Condens. Matter **26**, 183001 (2014).
[13] J. Behler, Int. J. Quantum Chem. **115**, 1032 (2015).
[14] K. T. Schütt, F. Arbabzadah, S. Chmiela, K. R. Müller, and A. Tkatchenko, Nat. Commun. **8**, 13890 (2017).
[15] A. V. Shapeev, Multiscale Model. Simul. **14**(3), 1153–1173 (2016).
[16] M. Hirn, N. Poilvert, and S. Mallat, preprint arXiv:1502.02077 (2015).
[17] M. Hirn, S. Mallat, and N. Poilvert, preprint arXiv:1605.04654 (2016).
[18] T. Hofmann, B. Schölkopf, and A. J. Smola, Ann. Stat. **36**, 1171 (2008).
[19] B. Scholkopf and A. J. Smola, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond* (MIT Press, Cambridge, 2001).
[20] A. P. Bartók, M. C. Payne, R. Kondor, and G. Csányi, Phys. Rev. Lett. **104**, 136403 (2010).
[21] M. Rupp, A. Tkatchenko, K.-R. Müller, and O. A. von Lilienfeld, Phys. Rev. Lett. **108**, 058301 (2012).
[22] K. Hansen, F. Biegler, R. Ramakrishnan, W. Pronobis, O. A. von Lilienfeld, K.-R. Müller, and A. Tkatchenko, J. Phys. Chem. Lett. **6**, 2326 (2015).
[23] K. T. Schütt, H. Glawe, F. Brockherde, A. Sanna, K. R. Müller, and E. K. U. Gross, Phys. Rev. B **89**, 205118 (2014).
[24] Z. Li, J. R. Kermode, and A. De Vita, Phys. Rev. Lett. **114**, 096405 (2015).
[25] G. Ferré, J.-B. Maillet, and G. Stoltz, J. Chem. Phys. **143**, 104114 (2015).
[26] T. Gärtner, K. Driessens, and J. Ramon, in *NIPS Workshop on Unreal Data: Principles of Modeling Nonvectorial Data* (NIPS, 2002), Vol. 5, pp. 49–58.
[27] T. Gärtner, P. Flach, and S. Wrobel, *Learning Theory and Kernel Machines* (Springer, 2003), pp. 129–143.
[28] S. V. N. Vishwanathan, N. N. Schraudolph, R. Kondor, and K. M. Borgwardt, J. Mach. Learn. Res. **11**, 1201 (2010).
[29] J. Drews, Science **287**, 1960 (2000).
[30] A. J. Smola and R. Kondor, *Learning Theory and Kernel Machines* (Springer, 2003), pp. 144–158.
[31] K. M. Borgwardt and H.-P. Kriegel, in *Fifth IEEE International Conference on Data Mining* (IEEE, 2005), p. 8.
[32] L. Ralaivola, S. J. Swamidass, H. Saigo, and P. Baldi, Neural Networks **18**, 1093 (2005).
[33] R. Sharan and T. Ideker, Nat. Biotechnol. **24**, 427 (2006).
[34] L. Schietgat, J. Ramon, M. Bruynooghe, and H. Blockeel, in *International Conference on Discovery Science* (Springer, 2008), pp. 197–209.
[35] A. Smalter, J. Huan, and G. Lushington, J. Bioinf. Comput. Biol. **7**, 473 (2009).
[36] B. Gaüzere, L. Brun, and D. Villemin, Pattern Recogn. Lett. **33**, 2038 (2012).
[37] H. Y. Sun, "Learning over molecules: Representations and kernels," Bachelor's thesis, Harvard College, 2014.
[38] J. Mercer, Philos. Trans. R. Soc., A **209**, 415 (1909).
[39] R. Kondor and K. M. Borgwardt, in *Proceedings of the 25th International Conference on Machine Learning* (ACM, 2008), pp. 496–503.
[40] N. Shervashidze, S. Vishwanathan, T. Petri, K. Mehlhorn, and K. M. Borgwardt, in *AISTATS* (NIPS, 2009), Vol. 5, pp. 488–495.
[41] A. Shrivastava and P. Li, *Advances in Social Networks Analysis and Mining* (IEEE, 2014), pp. 62–71.
[42] A. Shrivastava and P. Li, preprint arXiv:1404.5214 (2014).
[43] W. Imrich and S. Klavzar, *Product Graphs* (Wiley, 2000).
[44] P. Hohenberg and W. Kohn, Phys. Rev. **136**, B864 (1964).

[45]W. Kohn and L. J. Sham, Phys. Rev. **140**, A1133 (1965).

[46]G. Montavon, K. Hansen, S. Fazli, M. Rupp, F. Biegler, A. Ziehe, A. Tkatchenko, A. V. Lilienfeld, and K.-R. Müller, in *Advances in Neural Information Processing Systems*, edited by F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger (Curran Associates, Inc., 2012), Vol. 25, pp. 440–448.

[47]R. Kondor and H. Pan, in *Advances in Neural Information Processing Systems* (NIPS, 2016), pp. 2982–2990.

[48]D. K. Hammond, P. Vandergheynst, and R. Gribonval, Appl. Comput. Harmonic Anal. **30**, 129 (2011).

[49]X. Chen, X. Cheng, and S. Mallat, in *Advances in Neural Information Processing Systems* (NIPS, 2014), pp. 1709–1717.

[50]T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed. (Springer, New York, 2009), Chap. 12.3.7, pp. 436–437.

[51]A. E. Hoerl and R. W. Kennard, Technometrics **12**, 55 (1970).

[52]A. J. Smola, "Learning with kernels," Ph.D. thesis, Technische Universität Berlin, 1998.

[53]C. E. R. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning* (MIT Press, Cambridge, 2006).