Algorithmic Differentiation for Automated Modeling of Machine Learned Force Fields

Niklas Frederik Schmitz, Klaus-Robert Müller,* and Stefan Chmiela*

Cite This: J. Phys. Chem. Lett. 2022, 13, 10183−10189

ABSTRACT: Reconstructing force fields (FFs) from atomistic simulation data is a challenge since accurate data can be highly expensive. Here, machine learning (ML) models can help to be data economic as they can be successfully constrained using the underlying symmetry and conservation laws of physics. However, so far, every descriptor newly proposed for an ML model has required a cumbersome and mathematically tedious remodeling. We therefore propose using modern techniques from algorithmic differentiation within the ML modeling process, effectively enabling the usage of novel descriptors or models fully automatically at an order of magnitude higher computational efficiency. This paradigmatic approach enables not only a versatile usage of novel representations and the efficient computation of larger systems but also the simple inclusion of further physical knowledge, such as higher-order information (e.g., Hessians, more complex partial differential equations constraints etc.), even beyond the presented FF domain.

Most physical quantities are represented by differential equations (DEs)

$$L u(x) = f(x)$$

(1)

where \( L = \sum_{i<j} a_i D^j \) is a finite linear combination of differential operators \( D^j \) of order \( n \).

Examples of eq 1 are found in electrodynamics, fluid dynamics, quantum mechanics, etc., always imposing strong constraints on the form of a physical solution \( f(x) \). For several years, machine learning (ML) has started to broadly contribute to physical modeling across its disciplines, for instance in particle physics,1 atomistic simulations and force fields (e.g., see refs 2−6), fluid dynamics (e.g., see refs 7 and 8) or lattice gauge theories.9−12 Note, however, that the standard procedure of ML models has been so far to learn from empirical data only. Recently, ML modeling has also started to incorporate physical constraints, be it as regularization terms (e.g., see refs 13 and 14) or by explicitly taking into account conservation laws, DEs such as eq 1, symmetries (e.g., see refs 3, 15, and 16), various operator response properties17−18 or empirical correction terms, e.g., for long-ranged electrostatics.19−22 Notably, the direct usage of DEs allows the method to greatly simplify modeling and to be highly data efficient since the known laws of physics do not need to be learned from empirical data anymore (e.g., see refs 3, 15, and 16).

Let us study the popular example of eq 1 force fields (FFs), namely, the negative gradient \( L = -\nabla_x \) where \( u(x) \) is a scalar energy potential mapped to a vector field of forces \( f(x) \). The unknown force field (FF) is sampled at several locations \( \{x_i\}_{i=1}^M \) and \( f(x) \) are obtained from \textit{ab initio} atomistic reference data.5,15,23 The constraint

$$L = -\nabla_x$$

(2)

which is equivalent to both energy conservation and curl-free vector fields, enables the estimation of a faithful representation of true energy-conserving FFs as opposed to the estimation of general (unphysical) vector fields.5,24,25

This approach unfolds its usefulness well beyond the FF domain for higher-order differential operators and compositions thereof,26−30 which characterize complex properties across all physics domains, e.g., Hessians or Laplacian operators.31−33 Prior studies15−17,34−46 have demonstrated the effectiveness of explicit DE constraints, but their adoption still remains low due to an inefficiency that hampered model development and training so far: (1) Differential operator transforms inflate model complexity, making their full algebraic derivation mathematically tedious and error-prone. (2) Training and evaluation of a model using DE constraints is often associated with significant time and memory costs. While there exist prior works with focus on efficiency improvement, they rely on manual derivations and are limited to first-order gradients and primitive kernels without physical descrip-

Received: August 25, 2022
Accepted: October 20, 2022
Published: October 24, 2022
On the other hand, recent existing work that focuses on automating the derivation of (first-order) constraints is not able to avoid a costly dense instantiation of the full model.\textsuperscript{46}

In this work, we address both issues simultaneously using methods from algorithmic differentiation (AD) (e.g., refs \textsuperscript{47}−\textsuperscript{50}). While AD is now routinely used to automate the computation of derivatives, another benefit, which turns out to be essential for this work, is that it can simplify models that have a DE structure. As we will see, having a DE structure allows the collapse of certain portions of the computational graph early during evaluation, by preaccumulating derivatives according to the chain rule. In summary, there are three key aspects that make this work useful for a broad set of applications in physical modeling:

- It enables the systematic construction of empirical models subject to complex differential constraints.
- It is easy to use and versatile, even for elaborate neural network descriptors.
- It affords orders of magnitude gains in computational speed.

We demonstrate this for a number of popular FF models\textsuperscript{3} based on Gaussian process (GP) estimators under linear operator transformations. Such models can then be decomposed into tensor products of operators due to unique properties of the kernel function (see Figure 1).

As we will show, AD can yield efficient and automatic construction of GPs. Previously, a full instantiation of the DE constrained models was necessary, which due to the laborious manual derivations required, warranted a separate publication for each new constrained model (e.g. see refs \textsuperscript{15}−\textsuperscript{17}, \textsuperscript{25}, and \textsuperscript{51}). Using AD, we show that GPs can be created ad hoc and automatically (avoiding the above-discussed manual derivation steps) for arbitrary DE constraints, as we demonstrate by reimplementing a broad selection of popular classic as well as new ML-FFs; all can now be studied within our novel framework.

This newfound efficiency also enables us to easily recombine promising concepts from existing models into new, even more powerful physics models.

Specifically, AD enables us to break down differential expressions into sets of linear primitive operations, which are evaluated one-by-one to avoid a full instantiation of all intermediates of $Lu(x)$ whenever the response is needed only for a specific transformation of $x$. Due to the significant internal structure of derivatives, the individual terms that comprise $Lu$ are often low-dimensional and can thus be evaluated more...
efficiently by optimally managing the contraction order without having to undergo an unnecessary inflation.

Following this general idea, it turns out that most operator constraints can be applied with a surprisingly low overhead, often without even increasing the asymptotic computational complexity class of the original model. This fact is rather unexpected as it is intuitively counterintuitive when considering the outset. AD draws its efficiency from just two fundamental operations: 1. Given a differentiable function \( u: \mathbb{R}^N \rightarrow \mathbb{R}^N \) with total evaluation time cost \( O(N) \), AD guarantees that for \( x \in \mathbb{R}^N, v \in \mathbb{R}^N \), and \( w \in \mathbb{R}^N \):

\[
1. \text{Jacobian-vector product } J_u(x)v \text{ has cost } O(N) \\
2. \text{Jacobian-transpose-vector product } J_u^T(x)w \text{ has cost } O(N^2).
\]

Since any linear differential operator can be composed using these two simple rules, such operators are often at similar low cost (see Figure 1 and Tables S1 and SII), e.g., the Hessian-vector-products in \( O(N^2) \) instead of quadratic complexity. These improvements are possible because the full Jacobian never has to be calculated or stored.

Such inexpensive Jacobian products can be leveraged in any differentiable model, but they are particularly useful when applying DE constraints to GPs (for FFs) as we will now discuss

\[ u(x) \sim GP(\mu(x), k(x, x')) \]

is fully defined by a mean \( \mu: \mathbb{R}^N \rightarrow \mathbb{R} \) and a covariance function \( k(\cdot, \cdot): \mathbb{R}^N \times \mathbb{R}^N \rightarrow \mathbb{R} \) and is readily learned from data.24,52–54 For instance, many FFs, with \( n = 3N \) atomic degrees of freedom, are modeled by GPs from \( ab \) initio reference calculations (cf. refs 3–5 and 55). A natural benefit of GPs for DE-based regression is that they are closed under linear operators, leading to the form

\[ f(x) \sim GP(L\mu(x), L_{k(x)} \otimes L_{k(x)}^T(x, x')) \]

where subscripts indicate the argument of the kernel for each operator to act on.

As a concrete example, consider observing function values, gradients, and Hessians, i.e. \( L = (1, V, V^2) \) within the same model (cf. Figure 1). The corresponding kernel will take the form

\[ L_{k(x)} \otimes L_{k(x)}^T = \begin{bmatrix} k & \nabla k \\ \nabla k & \nabla k^T \\ \nabla k^T & \nabla k^T \\ \end{bmatrix}, \]

where each differential constraint appears as a cross-covariance combination with all other terms (e.g., the Hessian-Hessian covariance part being a fourth-order derivative). We leverage AD to construct this exceedingly complex matrix without the need to instantiate the corresponding analytical expressions. Rather, each term is constructed efficiently by contracting local Jacobian-vector products on-the-fly (cf. Figure 1).

Likewise, for third-order operators, the kernel uses a sixth-order derivative, and so on. At this complexity growth rate, a manual algebraic derivation becomes increasingly hopeless. However, with the help of AD, a mere definition of \( k(x, x') \) is sufficient, avoiding taking manual derivatives. Moreover, computationally the action of the operator avoids an intermediate generation of the full matrix expression, which gives rise to high computational efficiency.

We would like to discuss further the efficiency aspect of our AD framework, which also goes beyond standard AD usage. While the standard computation of the GP equations, for example, for the predictive mean are \( f(x) = \sum_i L_{k(x)} \otimes L_{k(x)}^T(x, x')(x, x') \), where coefficients \( \alpha \) are obtained by solving a linear system (see eqs S2 and S3 and refs 2 and 54). We note that this computation is usually dense and therefore slow.

Due to linearity, we can alternatively compute the same at roughly an order more efficiently (see Figure S1 and Table S1) by

\[ f(x) = \sum_i L_{k(x)} \otimes L_{k(x)}(x, x')(x, x') \]

as the operator tensor product is resolved at first into a scalar operator \( L_{k(x)} \), which can take full advantage of the automatic contraction rules inherent to AD. For an intuitive explanation see Figure 1, and for a more detailed derivation see the Supporting Information.

We will now apply the AD framework discussed above to the atomistic simulations domain. For this we proceed with the construction of potential energy surfaces (PESs) for small molecules from the well-established MD17 benchmark data set,15 using various GP-based ML-FFs that we recreate within our novel framework. First, we consider the Symmetric Gradient-domain Machine Learning (sGDML)16,58 model, which uses derivative constraints (eq 2) to simultaneously reconstruct conservative molecular FFs and their corresponding PESs. This model employs a twice differentiable kernel function from the parametric Matérn family,57–59 which is symmetrized to be invariant with respect to the relevant rigid space group, as well as dynamic nonrigid symmetries of the system at hand. It is then combined with a descriptor that enumerates all unique pairwise inverse distances between atoms. This composition of functions, if implemented in the standard manner,60 leads to an increased cost since all atomic degrees of freedom of the molecule enter the model as separate constraints.

Our AD framework resolves this complexity and avoids unnecessary instantiation of operator tensor products, yielding an order 3N improvement (where \( N \) is the number of atoms) in the case of gradient operators.

Furthermore, we have reimplemented other FF GP-based models, including FCHL19.25,61 Within our AD framework, we could do so by specifying the respective kernel functions without needing to manually implement derivatives. These models are compared to (s)GDML models trained on the exact same data set splits. To verify that our own implementations are correct, we have compared our test errors with the respective original publications.

Analyzing running times of the highly optimized FCHL19 reference implementation with our own AD-based one, we can already see the high intrinsic optimization abilities and computational advantages for our AD framework, which is able to generate predictions up to 2 orders of magnitude faster, while using only a single GPU instead of a CPU cluster node (cf. Table 2 in ref 25 and Table 3). We note, however, as a word of caution to this comparison that different compute architectures and programming languages are being used.

Table 3 contains a running time comparison of all considered models within our AD framework for the largest
We can even go a step further and use the representations learned by modern deep architectures\textsuperscript{62–64} as pretrained descriptors $D(x)$ within GDML-type models. Each descriptor then yields a new composite kernel

$$k(x, x') = k(D(x), D(x'))$$

(7)

Our numerical results show that this construction interestingly yields models that are more accurate than their (pretrained) ingredients. Tables 2 and 3 summarize the force prediction performance, showing that significant improvements are possible following this simple strategy. We reiterate that this advance was enabled by the unprecedented flexibility to remix different models enabled by our framework.

In order to demonstrate that the AD framework can also be applied beyond gradient observations (constraint from eq 2), we will now illustrate the effect of employing more complex higher-order DE constraints, namely, Hessians for the learning model.

Using the Hessian-kernel in eq 5, we reconstruct a toy surface from just two gradient/Hessian observations (see Figure 1). Notably, this example shows that constraints with higher derivatives are more informative and aid regression performance. With that, even a small sample size becomes sufficient to identify the correct model. For further illustrative examples using other DE constraints and ML-based DE solving, see Table S1.

Table 1. Test Errors (MAE) for Force Learning on MD17 with 1000 Reference Points Using Various New GP-FF Variants\textsuperscript{4}

| model         | aspirin | benzene | ethanol | malonaldehyde | naphthalene | salicylic acid | toluene | uracil |
|---------------|---------|---------|---------|---------------|-------------|----------------|---------|--------|
| sGDML         | 0.702   | 0.163   | 0.341   | 0.410         | 0.115       | 0.286          | 0.145   | 0.242  |
| FCHL19GPR     | 0.628   | 0.179   | 0.180   | 0.302         | 0.185       | 0.277          | 0.246   | 0.147  |
| sGDML\{RBF\} | 0.506   | 0.147   | 0.186   | 0.237         | 0.068       | 0.167          | 0.096   | 0.140  |
| global-FCHL19 | 0.748   | 0.235   | 0.319   | 0.429         | 0.292       | 0.216          | 0.370   | 0.120  |
| sFCHL19       | 0.430   | 0.158   | 0.160   | 0.274         | 0.094       | 0.172          | 0.111   | 0.130  |

“All errors are in kcal mol\textsuperscript{−1} Å\textsuperscript{−1}. Best results are in bold. The automation of AD allowed us to recombine existing ideas from different ML-based FFs on a broad scale and find better-performing model variants for all MD17 datasets. The best result for each dataset is marked in bold.

Table 2. Using Molecular Representations Generated by Various Deep Neural Network Architectures as Descriptor for GDML-Type Models\textsuperscript{4}

| model         | descriptor  | aspirin | ethanol | malonaldehyde | naphthalene | salicylic acid | toluene | uracil |
|---------------|-------------|---------|---------|---------------|-------------|----------------|---------|--------|
| SchNet        | SchNet      | 0.824   | 0.225   | 0.428         | 0.342       | 0.490          | 0.325   | 0.307  |
| sGDML\{RBF\} | SchNet      | 0.930   | 0.176   | 0.359         | 0.294       | 0.459          | 0.281   | 0.248  |
| PaiNN         | PaiNN (scalar features) | 0.389   | 0.220   | 0.336         | 0.103       | 0.232          | 0.122   | 0.176  |
| sGDML\{RBF\} | PaiNN       | 0.422   | 0.186   | 0.321         | 0.102       | 0.241          | 0.118   | 0.182  |

“All models have been trained on the same 1000 reference points for each molecule in the MD17 dataset. All (force) test errors (MAE) are in kcal mol\textsuperscript{−1} Å\textsuperscript{−1}. The best result for each dataset is marked in bold.

Table 3. Benchmarked Force Prediction Times (s) for Different Kernels\textsuperscript{4}

| model         | dense (N = 9) | contraction | speedup | dense (N = 21) | contraction | speedup |
|---------------|---------------|-------------|---------|---------------|-------------|---------|
| sGDML         | 0.0780        | 0.0018      | ×43.3   | 0.2832        | 0.0087      | ×32.5   |
| FCHL19GPR     | 57.9439       | 0.9815      | ×59.0   | 414.0216      | 10.1438     | ×40.8   |
| sGDML\{RBF\} | 0.0800        | 0.0015      | ×53.3   | 0.2811        | 0.0074      | ×37.9   |
| global-FCHL19 | 57.9194       | 0.9653      | ×59.5   | 458.5465      | 10.0728     | ×45.5   |
| sFCHL19       | 60.3951       | 1.0704      | ×56.4   | 419.3778      | 10.3336     | ×40.5   |

“Each model was trained using 1000 points and evaluated for a batch of 10 points. All timings are averaged over 10 runs (excluding an initial run for just-in-time compilation). Our approach (contraction) yields consistent speedups by a up to two orders of magnitude over the direct (dense) implementation of the constrained models. All measurements are done on a single Nvidia Titan RTX 24 GB GPU.
In summary, an overwhelming number of physical phenomena are governed by linear DEs that can be used as highly effective physical knowledge in data-driven estimation problems. By excluding physically infeasible solutions, such constraints play a crucial role in obtaining data-efficient and robust models. We have used methods from AD to address two key challenges that have so far hindered widespread adoption of this approach: (1) the full algebraic instantiation of the constrained model is expensive (or even unfeasible) to train and evaluate, and (2) the application of DEs to GPs by hand is tedious and error-prone for a modeler. Due to these obstacles, the construction of DE-constrained models did not leave much room for explorative and swift model creation in the past. To change that, we have contributed how our AD framework can be used to automate the model construction process and how regularities in the differential structure can be leveraged to gain efficiency in practice. This general framework has enabled us to construct and integrate various differential operator constraints that constitute the basic building blocks of most physical laws into ML-FFs. Following the same principle, more complex operators can be constructed and turned into constrained ML models; we used mainly GPs to show this point. For further examples, see the Supporting Information.

Finally, we have demonstrated in a series of numerical experiments how our framework can be used to readily replicate some state-of-the-art DE-constrained GP-based FFs by simply changing a few lines of code without the need to go through tedious manual derivations. Going one step further, we were able to demonstrate how competitive new kernelized variants of existing deep learning-based FFs can be developed and combined.

ASSOCIATED CONTENT
Supporting Information
The Supporting Information is available free of charge at https://pubs.acs.org/doi/10.1021/acs.jpcl.2c02632.

Further details about computational settings and benchmarks, toy examples, derivations of computational complexity considerations, Tables S1–SIV and Figures S1 and S2, as well as source code for all experiments (PDF) are available.

Transparent Peer Review report available (PDF)

AUTHOR INFORMATION
Corresponding Authors
Klaus-Robert Müller — Machine Learning Group, Technische Universität Berlin, 10587 Berlin, Germany; BIFOLD - Berlin Institute for the Foundations of Learning and Data, 10587 Berlin, Germany; Department of Artificial Intelligence, Korea University, Seoul 02841, Korea; Max Planck Institute for Informatics, 66123 Saarbrücken, Germany; Google Research, 10117 Berlin, Germany; orcid.org/0000-0002-3861-7685; Email: klaus-robert.mueller@tu-berlin.de

Stefan Chmiela — Machine Learning Group, Technische Universität Berlin, 10587 Berlin, Germany; BIFOLD - Berlin Institute for the Foundations of Learning and Data, 10587 Berlin, Germany; orcid.org/0000-0003-0892-952X; Email: stefan@chmiela.com

Author
Niklas Frederik Schmitz — Machine Learning Group, Technische Universität Berlin, 10587 Berlin, Germany

Complete contact information is available at: https://pubs.acs.org/10.1021/acs.jpcl.2c02632

Notes
The authors declare no competing financial interest.

ACKNOWLEDGMENTS
This work was supported in part by the German Ministry for Education and Research (BMBF) under Grants 01IS14013A-E, 01GQ1115, 01GQ0850, 01IS18056A, 01IS18025A, and 01IS18037A, by the Information & Communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (No. 2017-0-001779, Artificial Intelligence Graduate School Program, Korea University). We thank OT Unke for very helpful comments on the manuscript. Furthermore, we thank IPAM for warm hospitality and inspiration while finishing the manuscript.

REFERENCES

1. Baldi, P.; Sadowski, P.; Whiteson, D. Searching for exotic particles in high-energy physics with deep learning. Nat. comm. 2014, 5, 4308.

2. Rupp, M.; Tkatchenko, A.; Müller, K.-R.; von Lilienfeld, O. A. Fast and Accurate Modeling of Molecular Atomization Energies With Machine Learning. Phys. Rev. Lett. 2012, 108, 053001.

3. Unke, O. T.; Chmiela, S.; Sauceda, H. E.; Gastegger, M.; Poltavsky, I.; Schütt, K. T.; Tkatchenko, A.; Müller, K.-R. Machine Learning Force Fields. Chem. Rev. 2021, 121, 10142–10186.

4. von Lilienfeld, O. A.; Müller, K.-R.; Tkatchenko, A. Exploring Chemical Compound Space With Quantum-Based Machine Learning. Nat. Rev. Chem. 2020, 4, 347–358.

5. Noe, F.; Tkatchenko, A.; Müller, K.-R.; Clementi, C. Machine learning for molecular simulation. Annu. Rev. Phys. Chem. 2020, 71, 361–390.

6. Behler, J. Atom-centered symmetry functions for constructing high-dimensional neural network potentials. J. Chem. Phys. 2011, 134, 074106.

7. Brunton, S. L.; Noack, B. R.; Koumoutsakos, P. Machine learning for fluid mechanics. Annu. Rev. Fluid Mech. 2020, 52, 477–508.

8. Kochkov, D.; Smith, J. A.; Alieva, A.; Wang, Q.; Brenner, M. P.; Hoyer, S. Machine learning–accelerated computational fluid dynamics. Proc. Natl. Acad. Sci. U. S. A. 2021, 118, 118.

9. Albergo, M. S.; Kanwar, G.; Shanahan, P. E. Flow-based generative models for Markov chain Monte Carlo in lattice field theory. Phys. Rev. D 2019, 100, 034515.

10. Kanwar, G.; Albergo, M. S.; Boyda, D.; Cranmer, K.; Hackett, D. C.; Racanière, S.; Rezende, D. J.; Shanahan, P. E. Equivariant Flow-Based Sampling for Lattice Gauge Theory. Phys. Rev. Lett. 2020, 125, 126101.

11. Nicolì, K. A.; Anders, C. J.; Funcke, L.; Hartung, T.; Jansen, K.; Kessel, P.; Nakajima, S.; Stornati, P. Estimation of Thermodynamic Observables in Lattice Field Theories with Deep Generative Models. Phys. Rev. Lett. 2021, 126, 032001.

12. Luo, D.; Carleo, G.; Clark, B. K.; Stokes, J. Gauge equivariant neural networks for quantum lattice gauge theories. Phys. Rev. Lett. 2021, 127, 276402.

13. Karniadakis, G. E.; Kevrekidis, I. G.; Lu, L.; Perdikaris, P.; Wang, S.; Yang, L. Physics-informed machine learning. Nature Reviews Physics 2021, 3, 422–440.

14. Pun, G.; Batra, R.; Ramprasad, R.; Mishin, Y. Physically informed artificial neural networks for atomistic modeling of materials. Nat. Commun. 2019, 10, 2339.

15. Chmiela, S.; Tkatchenko, A.; Sauceda, H. E.; Poltavsky, I.; Schütt, K. T.; Müller, K.-R. Machine learning of accurate energy-conserving molecular force fields. Sci. Adv. 2017, 3, No. e1603015.
(16) Chmiela, S.; Sauceda, H. E.; Müller, K.-R.; Tkatchenko, A. Towards exact molecular dynamics simulations with machine-learned force fields. Nat. Commun. 2018, 9, 3887.

(17) Christensen, A. S.; Faber, F. A.; von Lilienfeld, O. A. Operators in quantum machine learning: Response properties in chemical space. J. Phys. Chem. 2019, 150, 064105.

(18) Unke, O.; Bogojecki, M.; Geigert, M.; Smidt, T.; Müller, K.-R. SE (3)-equivariant prediction of molecular wavefunctions and electronic densities. Adv. Neural Inf. Process. Syst. 2021, 34, 1.

(19) Yao, K.; Herr, J. E.; Toth, D. W.; Mckintyre, R.; Parkhill, J. The TensorMol-0.1 model chemistry: a neural network augmented with long-range physics. Chemical science 2018, 9, 2261–2269.

(20) Unke, O. T.; Meuwly, M. PhysNet: a neural network for predicting energies, forces, dipole moments, and partial charges. J. Chem. Theory Comput. 2019, 15, 3678–3693.

(21) Grisafi, A.; Ceriotti, M. Incorporating long-range physics in atomic-scale machine learning. J. Chem. Phys. 2019, 151, 204105.

(22) Unke, O. T.; Chmiela, S.; Geigert, M.; Schütt, K. T.; Sauceda, H. E.; Müller, K.-R. Spookynet: Learning force fields with electronic degrees of freedom and nonlocal effects. Nat. comm. 2021, 12, 7273.

(23) Bartók, A. P.; Csányi, G. Gaussian approximation potentials: A brief tutorial introduction. Int. J. Quantum Chem. 2015, 115, 1051–1057.

(24) Bartók, A. P.; Payne, M. C.; Kondor, R.; Csányi, G. Gaussian approximation potentials: The accuracy of quantum mechanics, without the electrons. Phys. Rev. Lett. 2010, 104, 136403.

(25) Christensen, A. S.; Bratholm, L. A.; Faber, F. A.; von Lilienfeld, O. A. FCHL revisited: faster and more accurate quantum machine learning. Phys. Rev. Lett. 2020, 125, 044107.

(26) Batz, P.; Ruttner, A.; Opper, M. Variational Estimation of the Drift for Stochastic Differential Equations from the Empirical Density. J. Stat. Mech. 2016, 2016, 083404.

(27) Sriperumbudur, B.; Fukumizu, K.; Gretton, A.; Hyvärinen, A.; Klus, S.; Lusch, R.; Mardt, A.; Pasquali, L.; Schölkopf, B.; Smidt, T.; Wang, Z.; Weinberger, K. Gaussian Processes: Application to Ordinary and Partial Differential Equations. Proceedings of the Twentieth International Conference on Machine Learning 2003, 234–241.

(28) Sriperumbudur, B.; Fukumizu, K.; Gretton, A.; Hyvärinen, A.; Zhang, L.; Lusch, R.; Schölkopf, B.; Smidt, T.; Wang, Z.; Weinberger, K. Gaussian Processes: Application to Ordinary and Partial Differential Equations. Proceedings of the Twentieth International Conference on Machine Learning 2003, 234–241.

(29) Lange-Hegermann, M. Linearly Constrained Gaussian Processes with Boundary Conditions. Proceedings of The 24th International Conference on Artificial Intelligence and Statistics; 2021. pp 1090–1098.

(30) Raissi, M.; Perdikaris, P.; Karniadakis, G. E. Machine Learning of Linear Differential Equations Using Gaussian Processes. J. Comput. Phys. 2017, 348, 683–693.

(31) Thomas, N.; Smidt, T.; Kearnes, S.; Yang, L.; Li, L.; Kohlhoff, K.; Riley, P. Tensor field networks: Rotation-and translation-equivariant neural networks for 3D point clouds. arXiv 2018, 1802.08219.

(32) Mardt, A.; Pasquali, L.; Wu, H.; Noé, F. VAMPnets for deep learning of molecular kinetics. Nat. Commun. 2018, 9, 5.

(33) Greidanus, S.; Dzamba, M.; Yosinski, J. Hamiltonian Neural Networks. Adv. Neural Inf. Process. Syst. 2019, 32.

(34) Crämer, M.; Greidanus, S.; Hoyer, S.; Battaglia, P.; Pürrer, D.; Ho, S. Lagrangian neural networks. arXiv 2020, 2003.04630.

(35) Lutter, M.; Ritter, C.; Peters, J. Deep Lagrangian Networks: Using Physics as Model Prior for Deep Learning. 7th International Conference on Learning Representations. 2019.

(36) Huang, D.; Teng, C.; Bao, J. L.; Tristan, J.-B. mad-GP: automatic differentiation of Gaussian processes for molecules and materials. J. Math. Chem. 2022, 60, 969–1000.

(37) Bryson, A. E.; Denham, W. F. A steepest-ascent method for solving optimum programming problems. J. Appl. Mech. 1962, 29, 247–257.

(38) Griewank, A.; Walther, A. Evaluating Derivatives: Principles and Techniques of Algorithmic Differentiation, 2nd ed.; SIAM, 2008.

(39) Goodrich, C. P.; King, E. M.; Schoenholtz, S. S.; Cubuk, E. D.; Brunner, M. Designing self-assembling kinetics with differentiable statistical physics models. Proc. Natl. Acad. Sci. U. S. A. 2021, 118, 118.

(40) Baydin, A. G.; Pearlmutter, B. A.; Radul, A. A.; Siskind, J. M. Automatic differentiation in machine learning: A survey. J. Machine Learn. Res. 2018, 18, 1–43.

(41) Klus, S.; Gell, P.; Nüske, F.; Noé, F. Symmetric and antisymmetric kernels for machine learning problems in quantum physics and chemistry. Machine Learning: Science and Technology 2021, 2, 045016.

(42) MacKay, D. J. C.Introduction to Gaussian Processes; NATO ASI Series f: Computer and Systems Sciences; Springer, 1998; Vol. 168, pp 133–166.

(43) Rasmussen, C. E.; Ghahramani, Z. Occam’s razor. Adv. Neural Inf. Process. Syst. 2001, 294–300.

(44) Rasmussen, C. E. Advanced Lectures on Machine Learning; Springer, 2004; pp 63–71.

(45) Keith, J. A.; Vassilev-Galindo, V.; Cheng, B.; Chmiela, S.; Geigert, M.; Müller, K.-R.; Tkatchenko, A. Combining Machine Learning and Computational Chemistry for Predictive Insights Into Chemical Systems. Chem. Rev. 2021, 121, 9816–9872.

(46) Chmiela, S.; Vassilev-Galindo, V.; Unke, O. T.; Kabylida, A.; Sauceda, H. E.; Tkatchenko, A.; Müller, K.-R. Accurate global machine learning force fields for molecules with hundreds of atoms. arXiv 2022, 2209.14865.

(47) Gneiting, T.; Kleiber, W.; Schlather, M. Matérn Cross-Covariance Functions for Multivariate Random Fields. J. Am. Stat. Assoc. 2010, 105, 1167–1177.

(48) Shichayev, I. S.; Ryzhik, I. M. In Table of Integrals, Series, and Products, Equation 8.468, 7th ed.; Jeffrey, A., Zwillinger, D., Eds.; 2007.

(49) Gneiting, T.; Kleiber, W.; Schlather, M. Matérn Cross-Covariance Functions for Multivariate Random Fields. J. Am. Stat. Assoc. 2010, 105, 1167–1177.

(50) Chmiela, S.; Sauceda, H. E.; Poltavsky, I.; Müller, K.-R.; Tkatchenko, A. sGDMC: Constructing Accurate and Data Efficient Molecular Force Fields Using Machine Learning. Comput. Phys. Commun. 2019, 240, 38–45.

(51) Faber, F. A.; Christensen, A. S.; Huang, B.; von Lilienfeld, O. A. Alchemical and structural distribution based representation for universal quantum machine learning. J. Chem. Phys. 2018, 148, 241717.
(62) Schütt, K. T.; Sauceda, H. E.; Kindermans, P.-J.; Tkatchenko, A.; Müller, K.-R. SchNet – A deep learning architecture for molecules and materials. J. Chem. Phys. 2018, 148, 241722.

(63) Schütt, K.; Unke, O.; Gastegger, M. Equivariant message passing for the prediction of tensorial properties and molecular spectra. Proc. Machine Learning Res. 2021, 9377–9388.

(64) Wilson, A. G.; Hu, Z.; Salakhutdinov, R.; Xing, E. P. Deep kernel learning. In Artificial intelligence and statistics; 2016; pp 370–378.