



ARTICLE



<https://doi.org/10.1038/s41467-021-27241-4>

OPEN

Learning neural network potentials from experimental data via Differentiable Trajectory Reweighting

Stephan Thaler ¹✉ & Julija Zavadlav ^{1,2}✉

In molecular dynamics (MD), neural network (NN) potentials trained bottom-up on quantum mechanical data have seen tremendous success recently. Top-down approaches that learn NN potentials directly from experimental data have received less attention, typically facing numerical and computational challenges when backpropagating through MD simulations. **We present the Differentiable Trajectory Reweighting (DiffTRe) method, which bypasses differentiation through the MD simulation for time-independent observables.** Leveraging thermodynamic perturbation theory, we avoid exploding gradients and achieve around 2 orders of magnitude speed-up in gradient computation for top-down learning. We show effectiveness of DiffTRe in learning NN potentials for an atomistic model of diamond and a coarse-grained model of water based on diverse experimental observables including thermodynamic, structural and mechanical properties. Importantly, DiffTRe also generalizes bottom-up structural coarse-graining methods such as iterative Boltzmann inversion to arbitrary potentials. The presented method constitutes an important milestone towards enriching NN potentials with experimental data, particularly when accurate bottom-up data is unavailable.

¹Professorship of Multiscale Modeling of Fluid Materials, TUM School of Engineering and Design, Technical University of Munich, Munich, Germany.

²Munich Data Science Institute, Technical University of Munich, Munich, Germany. ✉email: stephan.thaler@tum.de; julija.zavadlav@tum.de

Molecular modeling has become a cornerstone of many disciplines, including computational chemistry, soft matter physics, and material science. However, simulation quality critically depends on the employed potential energy model that defines particle interactions. There are two distinct approaches for model parametrization^{1,2}: Bottom-up approaches aim at matching data from high-fidelity simulations, providing labeled data of atomistic configurations with corresponding target outputs. Labeled data allow straightforward differentiation for gradient-based optimization, at the expense of inherently limiting model accuracy to the quality imposed by the underlying data-generating simulation. **On the other hand, top-down approaches optimize the potential energy model such that simulations match experimental data.** From experiments, however, labeled data on the atomistic scale are not available. Experimental observables are linked only indirectly to the potential model via an expensive molecular mechanics simulation, complicating optimization.

A class of potentials with tremendous success in recent years are neural network (NN) potentials due to their flexibility and capacity of learning many-body interactions^{3,4}. The vast majority of NN potentials are trained via bottom-up methods^{5–16}. The objective is to match energies and/or forces from a data set, most commonly generated via density functional theory (DFT) for small molecules in vacuum¹⁷. Within the data set distribution, state-of-the-art NN potentials have already reached the accuracy limit imposed by DFT, with the test error in predicting potential energy being around two orders of magnitude smaller than the corresponding expected DFT accuracy^{11,18}. In the limit of a sufficiently large data set without a distribution shift^{19,20} with respect to the application domain (potentially generated via active learning approaches²¹), remaining deviations of predicted observables from experiments are attributable to uncertainty in DFT simulations¹¹—in line with literature reporting DFT being sensitive to employed functionals²². More precise computational quantum mechanics models, e.g., the coupled cluster CCSD(T) method, improve DFT accuracy at the expense of significantly increased computational effort for data set generation^{23,24}. However, for larger systems such as macromolecules, quantum mechanics computations will remain intractable in the foreseeable future, preventing ab initio dataset generation altogether. Thus, the main obstacle in bottom-up learning of NN potentials is the currently limited availability of highly precise and sufficiently broad data sets. **自顶向下的方法不再需要生成可靠的模拟数据**

Top-down approaches circumvent the need for reliable data-generating simulations. Leveraging experimental data in the potential optimization process has contributed greatly to the success of classical atomistic^{25,26} and coarse-grained²⁷ (CG) force fields¹. Training difficulties have so far impeded a similar approach for NN potentials: Only recent advances in **automatic differentiation (AD)**²⁸ software have enabled end-to-end differentiation of molecular dynamics (MD) observables with respect to potential energy parameters^{29,30}, by applying AD through the dynamics of a MD simulation^{29–32}. This direct reverse-mode AD approach saves all simulator operations on the forward pass to be used during gradient computation on the backward pass, resulting in excessive memory usage. Thus, direct reverse-mode AD for systems with more than hundred particles and a few hundred time steps is typically intractable^{29–32}. **Numerical integration of the adjoint equations^{33,34} represents a memory-efficient alternative that requires to save only those atomic configurations that directly contribute to the loss.** However, both approaches back-propagate the gradient through the entire simulation, which dominates computational effort and is prone to exploding gradients, as stated by Ingraham et al.³¹ and shown below.

Addressing the call for NN potentials trained on experimental data¹, we propose the Differentiable Trajectory Reweighting

(DiffTRe) method. DiffTRe offers end-to-end gradient computation and circumvents the need to differentiate through the simulation **by combining AD with previous work on MD reweighting schemes^{35–38}**. For the common use case of time-independent observables, DiffTRe avoids exploding gradients and **reduces the computational effort of gradient computations by around two orders of magnitude compared to backpropagation through the simulation.** Memory requirements are comparable to the adjoint method. We showcase the broad applicability of DiffTRe on three numerical test cases: First, we provide insight into the training process on a toy example of ideal gas particles inside a double-well potential. Second, we train the state-of-the-art graph neural network potential DimeNet++^{11,12} for an atomistic model of the diamond from its experimental stiffness tensor. Finally, we learn a DimeNet++ model for CG water based on pressure, as well as radial and angular distribution functions. The last example shows how DiffTRe also generalizes bottom-up structural coarse-graining methods such as the iterative Boltzmann inversion³⁹ or inverse Monte Carlo⁴⁰ to many-body correlation functions and arbitrary potentials. **DiffTRe allows to enhance NN potentials with experimental data,** which is particularly relevant for systems where bottom-up data are unavailable or not sufficiently accurate.

Results

Differentiable Trajectory Reweighting. Top-down potential optimization aims to match the K outputs of a molecular mechanics simulation \mathbf{O} to experimental observables $\tilde{\mathbf{O}}$. Therefore, the objective is to minimize a loss function $L(\theta)$, e.g., a mean-squared error (MSE)

$$L(\theta) = \frac{1}{K} \sum_{k=1}^K [\langle O_k(U_\theta) \rangle - \tilde{O}_k]^2, \quad (1)$$

where $\langle \rangle$ denotes the ensemble average, and $\langle O_k(U_\theta) \rangle$ depends on the potential energy U_θ parametrized by θ . We will focus on the case where a MD simulation approximates $\langle O_k(U_\theta) \rangle$ —with Monte Carlo⁴¹ being a usable alternative. With standard assumptions on ergodicity and thermodynamic equilibrium, the ensemble average $\langle O_k(U_\theta) \rangle$ is approximated via a time average

$$\langle O_k(U_\theta) \rangle \simeq \frac{1}{N} \sum_{i=1}^N O_k(\mathbf{S}_i, U_\theta), \quad (2)$$

where $\{\mathbf{S}_i\}_{i=1}^N$ is the trajectory of the system, i.e., a sequence of N states consisting of particle positions and momenta. Due to the small time step size necessary to maintain numerical stability in MD simulations, states are highly correlated. Subsampling, i.e., only averaging over every 100th or 1000th state, reduces this correlation in Eq. (2).

As the generated trajectory depends on θ , every update of θ during training would require a re-computation of the entire trajectory. However, by leveraging thermodynamic perturbation theory⁴², it is possible to re-use decorrelated states obtained via a reference potential $\hat{\theta}$. Specifically, the time average is reweighted to account for the altered state probabilities $p_\theta(\mathbf{S}_i)$ from the perturbed potential θ ^{35,36,42}:

$$\langle O_k(U_\theta) \rangle \simeq \sum_{i=1}^N w_i O_k(\mathbf{S}_i, U_\theta) \quad \text{with} \quad w_i = \frac{p_\theta(\mathbf{S}_i)/p_{\hat{\theta}}(\mathbf{S}_i)}{\sum_{j=1}^N p_\theta(\mathbf{S}_j)/p_{\hat{\theta}}(\mathbf{S}_j)}. \quad (3)$$

Assuming a canonical ensemble, state probabilities follow the Boltzmann distribution $p_\theta(\mathbf{S}_i) \sim e^{-\beta H(\mathbf{S}_i)}$, where $H(\mathbf{S}_i)$ is the Hamiltonian of the state (sum of potential and kinetic energy), $\beta = 1/(k_B T)$, k_B Boltzmann constant, T temperature. Inserting $p_\theta(\mathbf{S}_i)$ into Eq. (3) allows computing weights as a function of θ

(the kinetic energy cancels)

$$w_i = \frac{e^{-\beta(U_\theta(\mathbf{S}_i) - U_\theta(\mathbf{S}_i))}}{\sum_{i=j}^N e^{-\beta(U_\theta(\mathbf{S}_j) - U_\theta(\mathbf{S}_j))}}. \quad (4)$$

For the special case of $\theta = \hat{\theta}$, $w_i = 1/N$, recovering Eq. (2). Note that similar expressions to Eq. (4) could be derived for other ensembles, e.g., the isothermal–isobaric ensemble, via respective state probabilities $p_\theta(\mathbf{S}_i)$. In practice, the reweighting ansatz is only applicable given small potential energy differences. For large differences between θ and $\hat{\theta}$, by contrast, few states dominate the average. In this case, the effective sample size³⁷

$$N_{\text{eff}} \approx e^{-\sum_{i=1}^N w_i \ln(w_i)} \quad (5)$$

is reduced and the statistical error in $\langle O_k(U_\theta) \rangle$ increases (Eq. (3)).

Reweighting can be exploited for two purposes that are linked to speedups in the forward and backward pass, respectively: first, reweighting reduces computational effort as decorrelated states from previous trajectories can often be re-used³⁷. Second, and most importantly, reweighting establishes a direct functional relation between $\langle O_k(U_\theta) \rangle$ and θ . This relation via \mathbf{w} provides an alternative end-to-end differentiable path for computing the gradient of the loss $\nabla_\theta L$: differentiating through the reweighting scheme replaces the backward pass through the simulation. Leveraging this alternative differentiation path, while managing the effective sample size N_{eff} , are the central ideas behind the DiffTRe method.

The workflow of the DiffTRe algorithm consists of the following steps: first, an initial reference trajectory is generated from the canonical ensemble, e.g., via a stochastic or deterministic thermostat, from an initial state \mathbf{S}_{init} and reference potential $\hat{\theta}$ (Fig. 1a). Initial equilibration states are disregarded and the following states are subsampled yielding decorrelated states $\{\mathbf{S}_i\}_{i=1}^N$. Together with their reference potential energies $\{U_{\hat{\theta}}(\mathbf{S}_i)\}_{i=1}^N$, these states are saved for re-use during reweighting. In the next step, the reweighting scheme is employed to compute $\nabla_\theta L$ with respect to current parameters θ , where initially $\theta = \hat{\theta}$. An optimizer subsequently uses $\nabla_\theta L$ to improve θ . This procedure of reweighting, gradient computation and updating is

repeated as long as the statistical error from reweighting is acceptably small, i.e., N_{eff} is larger than a predefined \bar{N}_{eff} . As soon as $N_{\text{eff}} < \bar{N}_{\text{eff}}$, a new reference trajectory needs to be sampled using the current θ as the new $\hat{\theta}$. At least one θ update per reference trajectory is ensured because initially $N_{\text{eff}} = N$. Using the last generated state \mathbf{S}_N as \mathbf{S}_{init} for the next trajectory counteracts overfitting to a specific initial configuration. In addition, $p_{\hat{\theta}}(\mathbf{S}_{\text{init}})$ is reasonably high when assuming small update steps, reducing necessary equilibration time for trajectory generation. Saving only $\{\mathbf{S}_i\}_{i=1}^N$ and $\{U_{\hat{\theta}}(\mathbf{S}_i)\}_{i=1}^N$ from the simulation entails low-memory requirements similar to the adjoint method. DiffTRe assumes that deviations in predicted observables are attributable to an inaccurate potential U_θ rather than a statistical sampling error. Accordingly, N and the subsampling ratio n need to be chosen to yield a sufficiently small statistical error. Optimal values for N and n depend on the specific system, target observables, and the thermodynamic-state point.

Computation of $\nabla_\theta L$ via reverse-mode AD through the reweighting scheme comprises a forward pass starting with computation of the potential $U_\theta(\mathbf{S}_i)$ and weight w_i for each \mathbf{S}_i (Eq. (4); Fig. 1a). Afterward, reweighted observables $\langle O_k(U_\theta) \rangle$ (Eq. (3)) and the resulting loss $L(\theta)$ (Eq. (1)) are calculated. The corresponding backward pass starts at $L(\theta)$ and stops at parameters θ in the potential energy computation $U_\theta(\mathbf{S}_i)$. The differentiation path defined by the reweighting ansatz is therefore independent of the trajectory generation.

Evaluation of $U_\theta(\mathbf{S}_i)$ (Fig. 1b) involves computing the pairwise distance matrix \mathbf{D} from atom positions of \mathbf{S}_i , that are fed into a learnable potential U_θ^{model} and a prior potential U^{prior} . Both potential components are combined by adding the predicted potential energies

$$U_\theta(\mathbf{S}_i) = U_\theta^{\text{model}}(\mathbf{D}) + U^{\text{prior}}(\mathbf{D}). \quad (6)$$

In subsequent examples of diamond and CG water, U_θ^{model} is a graph neural network operating iteratively on the atomic graph defined by \mathbf{D} . U^{prior} is a constant potential approximating a priori-known properties of the system, such as the Pauli exclusion principle (e.g., Eq. (12)). Augmenting NN potentials with a prior

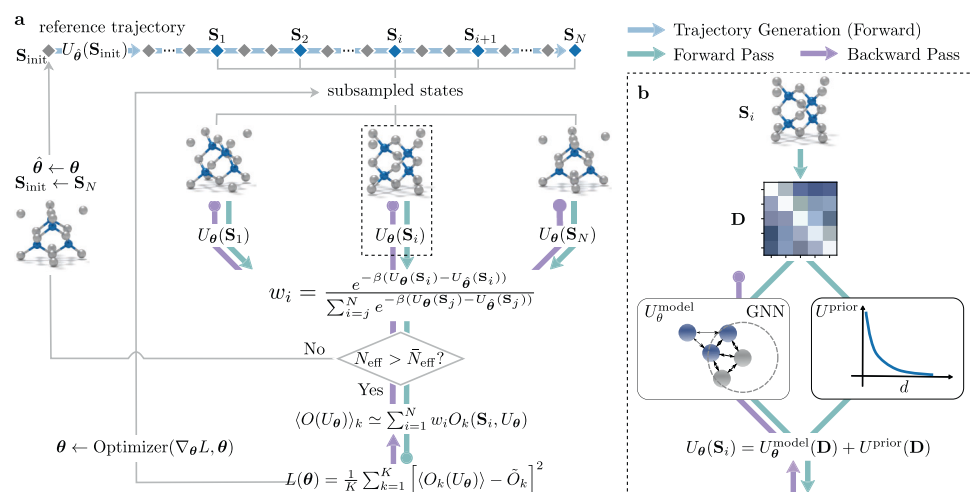


Fig. 1 Differentiable Trajectory Reweighting (DiffTRe). **a** Based on an initial state \mathbf{S}_{init} and reference potential parameters $\hat{\theta}$, a reference trajectory is generated, of which only subsampled states are retained (blue diamonds), while the majority of visited states are discarded (gray diamonds). For each retained state \mathbf{S}_i (represented by a generic molecular system), the potential energy $U_\theta(\mathbf{S}_i)$ and weight w_i are computed under the current potential parameters θ . w_i allow computation of reweighted observables $\langle O_k(U_\theta) \rangle$, the loss $L(\theta)$, its gradient $\nabla_\theta L$ and subsequently, updating θ via the optimizer. The updating procedure is repeated until the effective sample size $N_{\text{eff}} < \bar{N}_{\text{eff}}$, at which point a new reference trajectory needs to be generated starting from the last sampled state \mathbf{S}_N . **b** Computation of $U_\theta(\mathbf{S}_i)$ from the pairwise distance matrix \mathbf{D} , which is fed into the learnable potential U_θ^{model} (e.g., a graph neural network—GNN) and U^{prior} (e.g., a pairwise repulsive potential).

is common in the bottom-up coarse-graining literature^{8,10} to provide qualitatively correct behavior in regions of the potential energy surface (PES) not contained in the dataset, but reachable by the CG model. By contrast, DiffTRe does not rely on pre-computed data sets. Rather, the prior serves to control the data (trajectory) generation in the beginning of the optimization. In addition, U^{prior} reformulates the problem from learning $U_{\theta}^{\text{model}}$ directly to learning the difference between U^{prior} and the optimal potential given the data¹⁰. A well-chosen U^{prior} therefore represents a physics-informed initialization accelerating training convergence. Suitable U^{prior} can often be found in the literature: Classical force fields such as AMBER²⁵ and MARTINI²⁷ define reasonable interactions for bio-molecules and variants of the Embedded Atom Model⁴³ (EAM) provide potentials for metals and alloys. Note that U^{prior} is not a prior in the Bayesian sense providing a pervasive bias on learnable parameters in the small data regime. If U^{prior} is in contradiction with the data, $U_{\theta}^{\text{model}}$ will correct for U^{prior} as a result of the optimization. In the next section, we further illustrate for a toy problem the interplay between prior, gradients and the learning process in DiffTRe, and provide a comparison to direct reverse-mode AD through the simulation.

Double-well toy example. We consider ideal gas particles at a temperature $k_B T = 1$ trapped inside a one-dimensional double-well potential (Fig. 2a) parametrized by

$$U(x) = k_B T * [2500(x - 0.5)^6 - 10(x - 0.55)^2]. \quad (7)$$

The goal is to learn θ such that $U_{\theta}(x) = U_{\theta}^{\text{model}}(x) + U^{\text{prior}}(x)$ matches $U(x)$. We select a cubic spline as $U_{\theta}^{\text{model}}$, which acts as a flexible approximator for twice continuously differentiable functions. The cubic spline is parametrized via the potential energy

values of 50 control points $\{x_j, U_j\}_{j=1}^{50}$ evenly distributed over $x \in [0, 1]$. Analogous to NN potentials in subsequent problems, we randomly initialize $U_j \sim \mathcal{N}(0, 0.01^2 k_B T)$. Initializing $U_j = 0$ leads to largely identical results in this toy problem. The harmonic single-well potential $U^{\text{prior}}(x) = \lambda(x - 0.5)^2$, with scale $\lambda = 75$, encodes the prior knowledge that particles cannot escape the double-well. We choose the normalized density profile $\rho(x)/\rho_0$ of ideal gas particles as the target observable. The resulting loss function is

$$L = \frac{1}{K} \sum_{k=1}^K \left(\frac{\langle \rho(x_k) \rangle}{\rho_0} - \frac{\tilde{\rho}(x_k)}{\rho_0} \right)^2, \quad (8)$$

where $\rho(x)$ is discretized via K bins. $\langle \rho(x_k) \rangle$ are approximated based on $N = 10,000$ states after skipping 1000 states for equilibration, where a state is retained every 100 time steps. We minimize Eq. (8) via an Adam⁴⁴ optimizer with learning rate decay. For additional DiffTRe and simulation parameters, see Supplementary Method 1.1.

Initially, ρ/ρ_0 resulting from $U^{\text{prior}}(x)$ deviates strongly from the target double-well density (Fig. 2b). The loss curve illustrates successful optimization over 200 update steps (Fig. 2c). The wall-clock time per parameter update Δt clearly shows two distinct levels: at the start of the optimization, update steps are rather large, significantly reducing N_{eff} . Hence a new reference trajectory generation is triggered with each update (average $\Delta t \approx 39.2$ s). Over the course of the simulation, updates of $U_{\theta}^{\text{model}}(x)$ become smaller and reference trajectories are occasionally re-used (average $\Delta t \approx 2.76$ s). After optimization, the target density is matched well. The learned potential energy function $U_{\theta}(x)$ recovers the data-generating potential $U(x)$ (Supplementary Fig. 1a); thus, other thermodynamic and kinetic observables will match reference values closely. However, this conclusion does not

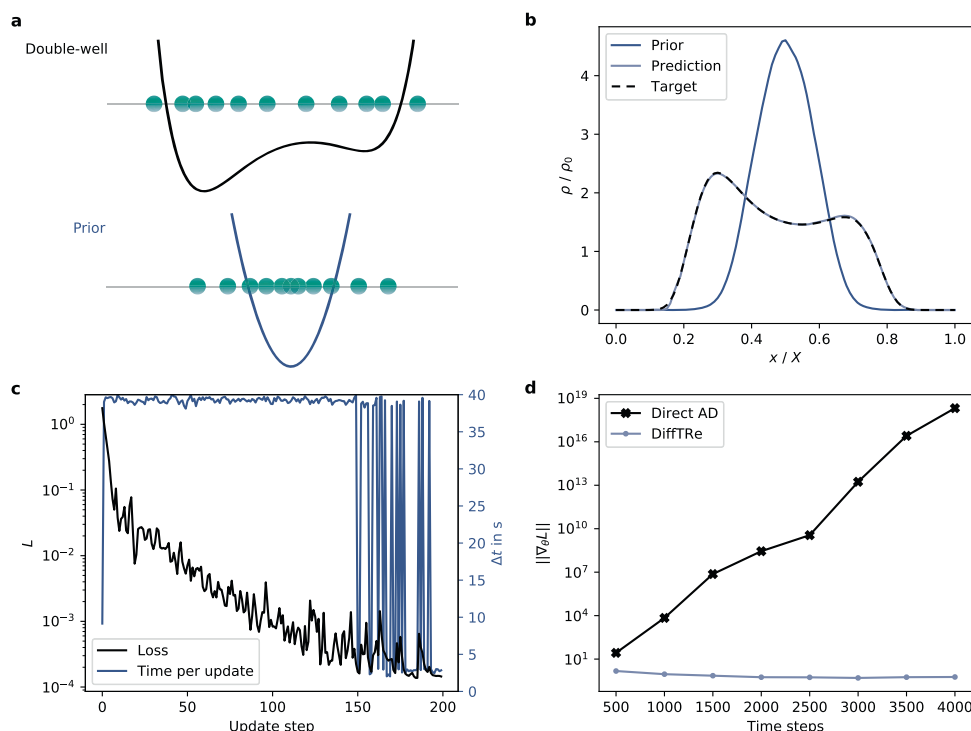


Fig. 2 Double-well toy example. **a** Sketch of the double-well and prior potential with corresponding example states of ideal gas particles (green circles). The learned potential results in a normalized density ρ/ρ_0 (over the normalized position x/X) that matches the target closely (**b**). Successful learning is reflected in the loss curve L , where a significant reduction in wall-clock time per parameter update Δt towards the end of the optimization is achieved through re-using previously generated trajectories (**c**). Gradients computed via DiffTRe have constant magnitudes while gradients obtained from direct reverse-mode automatic differentiation through the simulation suffer from exploding gradients for longer trajectories (**d**).

apply in realistic applications, where learned potentials are in general not unique² due to the limited number of target observables that can be considered in practice.

The effect of U^{prior} on the training process is twofold: First, by encoding prior knowledge, it simplifies convergence, as $U_{\theta}^{\text{model}}(x)$ only needs to adapt the single-well prior instead of learning large energy barriers from scratch. Second, U^{prior} also impacts the information content of the gradient by controlling the generation of trajectories in the beginning of the optimization (Eq. (15)). The local support of the cubic spline allows analyzing this relation empirically (Supplementary Fig. 2): The gradient is nonzero only in regions of the PES that are included in the reference trajectory. Hence, other regions of the PES are not optimized despite delivering a nonzero contribution to the loss. A well-chosen prior potential should therefore yield trajectories that are as close as possible to trajectories sampled from the true potential. However, satisfactory learning results can be obtained for a sensible range of prior scales (Supplementary Fig. 3).

We study the robustness of our results by varying the random seed that controls the initialization of the spline as well as the initial particle positions and velocities. Results from the variation study in Supplementary Fig. 4 demonstrate that the predicted $\rho(x)/\rho_0$ is robust to the random initialization. The corresponding $U_{\theta}(x)$ exhibits some variance at the left well boundary, mirroring difficult training in this region due to vanishing gradients for vanishing predicted densities (Supplementary Fig. 2) and minor influence of the exact wall position on the resulting density profile (Supplementary Fig. 4a).

For comparison, we have implemented gradient computation via direct reverse-mode AD through the simulation. This approach clearly suffers from the exploding gradients problem (Fig. 2d): The gradient magnitude increases exponentially as a function of the simulation length. Without additional modifications (e.g., as implemented by Ingraham et al.³¹), these gradients are impractical for longer trajectories. By contrast, gradients computed via DiffTRe show constant magnitudes irrespective of the simulation length.

To measure the speed-up over direct reverse-mode AD empirically, we simulate the realistic case of an expensive potential by substituting the numerically inexpensive spline with a fully connected neural network with two hidden layers and 100 neurons each. We measure speedups of $s_g = 486$ for gradient computations and $s = 3.7$ as overall speed-up per update when a new reference trajectory is sampled. However, these values are rather sensitive to the exact computational and simulation setup. Memory overflow in the direct AD method constrained trajectory lengths to ten retained states and a single state for equilibration (a total of 1100 time steps). Measuring speed-up for one of the real-world problems below would be desirable, but is prevented by the memory requirements of direct AD.

The measured speed-up values are in line with theoretical considerations: While direct AD backpropagates through the whole trajectory generation, DiffTRe only differentiates through the potential energy computation of decorrelated states $\{\mathbf{S}_i\}_{i=1}^N$ (Fig. 1). From this algorithmic difference, we expect speed-up values that depend on the subsampling ratio n , the number of skipped states during equilibration N_{equilib} and the cost multiple of backward passes with respect to forward passes G (details in Supplementary Method 2)

$$s_g \sim Gn \left(1 + N_{\text{equilib}}/N \right); \quad s \sim G + 1. \quad (9)$$

For this toy example setup, the rule-of-thumb estimates in Eq. (9) yield $s_g = 330$ and $s = 4$, agreeing with the measured values. In the next sections, we showcase the effectiveness of DiffTRe in real-world, top-down learning of NN potentials.

Atomistic model of diamond. To demonstrate the applicability of DiffTRe to solids on the atomistic scale, we learn a DimeNet++¹² potential for diamond from its experimental elastic stiffness tensor \mathbf{C} . Due to symmetries in the diamond cubic crystal, \mathbf{C} only consists of three distinct stiffness moduli $\tilde{C}_{11} = 1079$ GPa, $\tilde{C}_{12} = 124$ GPa and $\tilde{C}_{44} = 578$ GPa⁴⁵ (in Voigt notation). In addition, we assume the crystal to be in a stress-free state $\boldsymbol{\sigma} = \mathbf{0}$ for vanishing infinitesimal strain $\boldsymbol{\epsilon} = \mathbf{0}$. These experimental data define the loss

$$L = \frac{\gamma_{\sigma}}{9} \sum_{i=1}^3 \sum_{j=1}^3 \sigma_{ij}^2 + \frac{\gamma_C}{3} \left((C_{11} - \tilde{C}_{11})^2 + (C_{12} - \tilde{C}_{12})^2 + (C_{44} - \tilde{C}_{44})^2 \right), \quad (10)$$

where loss weights γ_{σ} and γ_C counteract the effect of different orders of magnitude of observables. To demonstrate learning, we select the original Stillinger–Weber potential⁴⁶ parametrized for silicon as U^{prior} . We have adjusted the length and energy scales to $\sigma_{\text{SW}} = 0.14$ nm and $\epsilon_{\text{SW}} = 200$ kJ/mol, reflecting the smaller size of carbon atoms. We found learning to be somewhat sensitive to U^{prior} in this example because weak prior choices can lead to unstable MD simulations. Simulations are run with a cubic box of size $L \approx 1.784$ nm containing 1000 carbon atoms (Fig. 3a) to match the experimental density ($\rho = 3512$ kg/m³)⁴⁵ exactly. The temperature in the experiment ($T = 298.15$ K⁴⁵) determines the simulation temperature. Each trajectory generation starts with 10 ps of equilibration followed by 60 ps of production, where a decorrelated state is saved every 25 fs. We found these trajectories to yield observables with acceptably small statistical noise. The stress tensor $\boldsymbol{\sigma}$ is computed via Eq. (13) and the stiffness tensor \mathbf{C} via the stress fluctuation method (Eq. (14)). Further details are summarized in Supplementary Method 4.

Figure 3 visualizes convergence of the stress (b) and stiffness components (c). Given that the model is only trained on rather short trajectories, we test the trained model on a trajectory of 10 ns length to ensure that the model neither overfitted to initial conditions nor drifts away from the targets. The resulting stress and stiffness values $\sigma_1 = 0.29$ GPa, $\sigma_4 = 0.005$ GPa, $C_{11} = 1070$ GPa, $C_{12} = 114$ GPa, and $C_{44} = 560$ GPa are in good agreement with respective targets. These results could be improved by increasing the trajectory length, which reduces statistical sampling errors. The corresponding inverse stress–strain relation is given by the compliance tensor $\mathbf{S} = \mathbf{C}^{-1}$, which can be constructed from by Young’s modulus $E = 1047$ GPa, shear modulus $G = 560$ GPa, and Poisson’s ratio $\nu = 0.097$. The training loss curve and wall-clock time per update Δt are displayed in Supplementary Fig. 5a.

Computing the stress–strain curve (Supplementary Fig. 5b) from the trained model in the linear regime ($\epsilon_i < 0.005$) verifies that computing \mathbf{C} via Eq. (14) yields the same result as explicitly training the box and measuring stresses. In addition, this demonstrates that the DimeNet++ potential generalizes from the training box ($\epsilon = 0$) to boxes under small strain. We also strained the box beyond the linear regime, creating a distribution shift^{19,20}, to test generalization to unobserved state points. The predicted stress–strain curve in Fig. 3d shows good agreement with DFT data⁴⁷ for medium-sized natural strains $\epsilon_1 = \log(1 + \epsilon_1) < 0.02$. For large strains, the deviation quickly increases, including an early fracture. These incorrect predictions of the learned potential are due to limited extrapolation capacities of NN potentials: states under large strain are never encountered during training, leading to large uncertainty in predicted forces. Incorporating additional observables linked to states of large strain into the optimization, such as the point of maximum stress, should improve predictions.

To test the trained DimeNet++ potential on held-out observables, we compute the phonon density of states (PDOS).

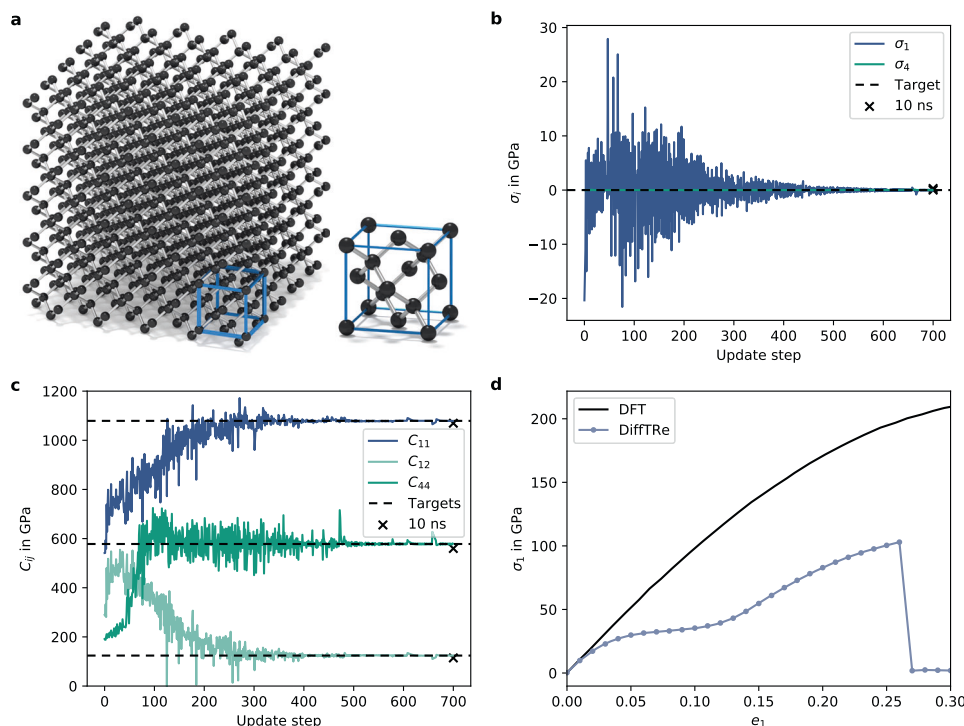


Fig. 3 Atomistic model of the diamond. The simulation box consists of five diamond unit cells in each direction, whose primary crystallographic directions $[1, 0, 0]$, $[0, 1, 0]$ and $[0, 0, 1]$ are aligned with the x , y , and z axes of the simulation box (a). Stress σ_i (b) and stiffness values C_{ij} (c) converge to their respective targets during the optimization. These results are robust to long simulation runs of 10 ns (marked with crosses). The stress-strain curve over normal natural strains e_1 agrees with density functional theory (DFT) data⁴⁷ for medium-sized strains ($e_1 < 0.02$), but deviates for large strains due to limited extrapolation capabilities of neural network potentials (d).

The predicted PDOS deviates from the experiment⁴⁸, analogous to a Stillinger–Weber potential optimized for diamond⁴⁹ (Supplementary Fig. 5c). The evolution of the predicted PDOS over the course of the optimization is shown in Supplementary Fig. 5d. Deviations of held-out observables are expected given that top-down approaches allow learning potentials that are consistent with target experimental observables but lack theoretical convergence guarantees of bottom-up schemes (in the limit of a sufficiently large data set and a sufficiently expressive model)². In principle, we expect sufficiently expressive top-down models to converge to the true potential in the limit of an infinite number of matched target observables. In practice, however, many different potentials can reproduce a sparse set of considered target observables, rendering the learned potential non-unique². In this particular example, we show that many different potentials can reproduce the target stress and stiffness, but predict different PDOSs: While predicted stress and stiffness values are robust to random initialization of NN weights and initial particle velocities within the statistical sampling error, the corresponding predicted PDOSs vary to a great extent (Supplementary Fig. 6). Incorporating additional observables more closely connected to phonon properties into the loss function could improve the predicted PDOS.

Coarse-grained water model. Finally, we learn a DimeNet++ potential for CG water. Water is a common benchmark problem due to its relevance in bio-physics simulations and its pronounced 3-body interactions, which are challenging for classical potentials⁵⁰. We select a CG-mapping, where each CG particle is centered at the oxygen atom of the corresponding atomistic water molecule (Fig. 4a). This allows using experimental oxygen–oxygen radial (RDF) and angular distribution functions (ADF) as target observables. Given that the reference

experiment⁵¹ was carried out at ambient conditions ($T = 296.15$ K), we can additionally target a pressure $\tilde{p} = 1$ bar. Hence, we minimize

$$L = \frac{1}{G} \sum_{g=1}^G (RDF(d_g) - \tilde{RDF}(d_g))^2 + \frac{1}{M} \sum_{m=1}^M (ADF(\alpha_m) - \tilde{ADF}(\alpha_m))^2 + \gamma_p (p - \tilde{p})^2. \quad (11)$$

As the prior potential, we select the repulsive term of the Lennard–Jones potential

$$U^{\text{prior}}(d) = \epsilon_R \left(\frac{\sigma_R}{d} \right)^{12}. \quad (12)$$

Drawing inspiration from atomistic water models, we have chosen the length scale of the SPC⁵² water model as $\sigma_R = 0.3165$ nm as well as a reduced energy scale of $\epsilon_R = 1$ kJ/mol to counteract the missing Lennard–Jones attraction term in Eq. (12). We build a cubic box of length 3 nm with 901 CG particles, implying a density of $\rho = 998.28$ g/l, to match the experimental water density of $\rho = 997.87$ g/l at 1 bar. Trajectory generation consists of 10 ps of equilibration and 60 ps of subsequent production, where a decorrelated state is saved every 0.1 ps. For additional details, see Supplementary Method 2.3.

Figure 4b–d displays properties predicted by the final trained model during a 10 ns production run: DiffTRe is able to train a DimeNet++ potential that simultaneously matches experimental oxygen RDF, ADF, and pressure to the line thickness. The evolution of predicted RDFs and ADFs as well as the loss and wall-clock times per update are displayed in Supplementary Fig. 7a–c. The learning process is robust to weak choices of U^{prior} : DiffTRe is able to converge to the same prediction quality as with the reference prior even if σ_R is misestimated by ± 0.1 nm (approximately $\pm 30\%$) compared to the classical SPC water model (Supplementary Fig. 8a, b). This represents a large variation given

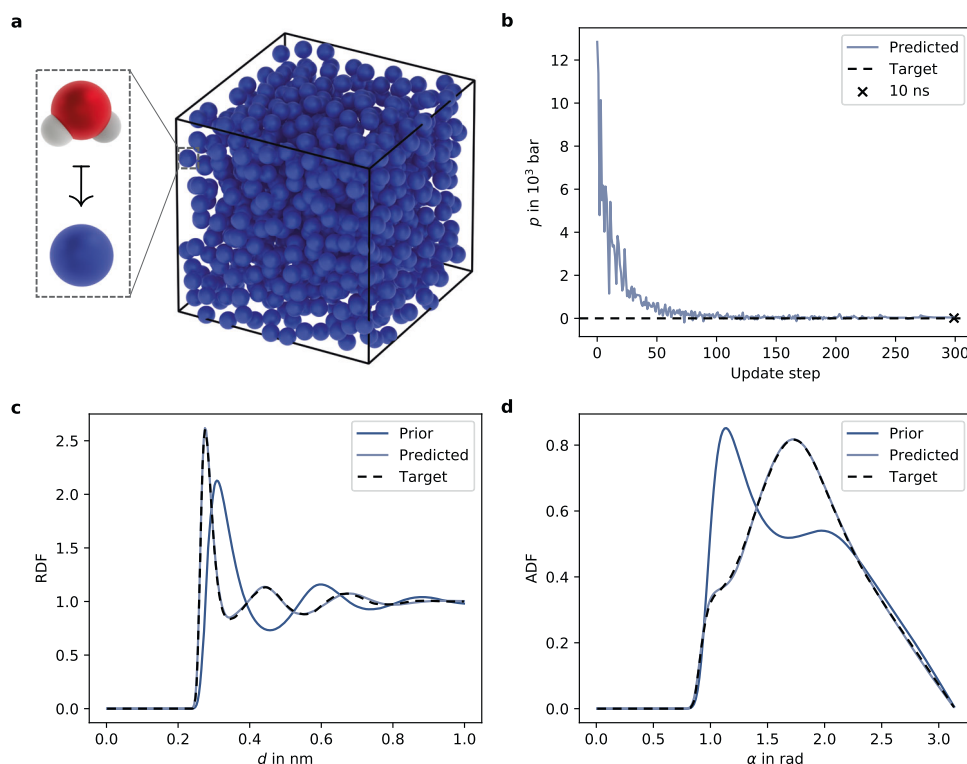


Fig. 4 Coarse-grained model of water. Coarse-grained particles representing water molecules are visualized as blue balls in the simulation box (a). The pressure p converges quickly toward its target of 1 bar during optimization and the subsequent 10 ns simulation (black cross; $p \approx 12.9$ bar) verifies the result (b). Over a 10 ns simulation, the learned potential reconstructs the experimental radial distribution function (RDF) and angular distribution function (ADF) well (c, d).

that within common atomistic water models, σ_R varies by $<0.5\%$ ⁵³.

To test the learned potential on held-out observables, we compute the tetrahedral order parameter q ⁵⁴ and the self-diffusion coefficient D . $q \approx 0.569$ matches the experimental value of $\tilde{q} = 0.576$ closely. This is expected as q considers the structure of four nearest neighbor particles, which is closely related to the ADF. The learned CG water model predicts a larger self-diffusion coefficient than were experimentally measured ($D = 10.91 \mu\text{m}^2/\text{ms}$ vs. $\tilde{D} = 2.2 \mu\text{m}^2/\text{ms}$)⁵⁵. With the same simulation setup, a single-site tabulated potential parametrized via iterative Boltzmann inversion³⁹ with pressure correction^{39,56} predicts $D = 14.15 \mu\text{m}^2/\text{ms}$. These results are in line with the literature: Due to smoother PESs, CG models exhibit accelerated dynamical processes compared to atomistic models². For CG water models specifically, diffusion coefficients decrease with increasing number of interaction sites⁵⁷. In this context, the decreasing diffusion coefficients over the course of the optimization (Supplementary Fig. 7d) could indicate that U_θ acts effectively as a single-site model in the beginning, while learning 3-body interactions during the optimization casts U_θ more similar to multisite CG models. Obtained results are robust to random initialization of NN weights and initial particle velocities, both for predicted target (Supplementary Fig. 8c, d) and held-out observables ($D = 10.93 \pm 0.20 \mu\text{m}^2/\text{ms}$).

The accuracy of predicted 2 and 3-body interactions (Fig. 4c, d) showcases the potency of graph neural network potentials in top-down molecular modeling: capturing 3-body interactions is essential for modeling water given that pair potentials trained via force matching fail to reproduce both RDF and ADF of the underlying high-fidelity model⁵⁰. Other top-down CG water models with simple functional form tend to deviate from the

experimental RDF^{58,59}. Deviations from experimental structural properties, albeit smaller in size, also arise in DFT simulations^{22,60}, limiting the accuracy of bottom-up trained NN potentials⁸.

Discussion

In this work, we demonstrate numerically efficient learning of NN potentials from experimental data. The main advantages of our proposed DiffTRe method are its flexibility and simplicity: DiffTRe is applicable to solid and fluid materials, coarse-grained and atomistic models, thermodynamic, structural and mechanical properties, as well as potentials of arbitrary functional form. To apply DiffTRe, practitioners only need to set up a MD simulation with corresponding observables and a loss function, while gradients are computed conveniently in an end-to-end fashion via AD. The demonstrated speedups and limited memory requirements promote application to larger systems.

Without further adaptations, DiffTRe can also be applied as a bottom-up model parametrization scheme. In this case, a high-fidelity simulation, rather than an experiment, provides target observables. For CG models, DiffTRe generalizes structural coarse-graining schemes such as iterative Boltzmann inversion³⁹ or Inverse Monte Carlo⁴⁰. DiffTRe overcomes the main limitations of these approaches: First, structural coarse-graining is no longer restricted to one-dimensional potentials, and matching many-body correlation functions (e.g., ADFs) is therefore feasible. Second, the user can integrate additional observables into the optimization without relying on hand-crafted iterative update rules, for instance for pressure-matching^{39,56}. This is particularly useful if an observable needs to be matched precisely (e.g., pressure in certain multiscale simulations⁶¹). Matching many-body

correlation functions will likely allow structural bottom-up coarse-graining to take on significance within the new paradigm of many-body CG potentials^{8–10}.

For the practical application of DiffTRe, a few limitations need to be considered. The reweighting scheme renders DiffTRe invariant to the sequence of states in the trajectory. Hence, dynamical properties cannot be employed as target observables. In addition, the NN potential test cases considered in this work required a reasonably chosen prior potential. Lastly, two distinct sources of overfitting when learning from experimental data for a single system need to be accounted for¹: To avoid overfitting to a specific initial state, DiffTRe uses a different initial state for each reference trajectory. Moreover, increasing the system size and trajectory length ensures representative reference trajectories. Irrespective of overfitting, generalization to different systems, observables, and thermodynamic-state points remains to be addressed, for instance via training on multi-systemic experimental data sets. To this end, an in-depth assessment of out-of-sample properties of top-down learned NN potentials is required.

From a machine learning (ML) perspective, DiffTRe belongs to the class of end-to-end differentiable physics approaches^{62–64}. These approaches are similar to reinforcement learning in that the target outcome of a process (here a MD simulation) represents the data. A key difference is the availability of gradients through the process, allowing for efficient training. Differentiable physics approaches, increasingly popular in control applications^{34,65–67}, enable direct training of the ML model via the physics simulator, advancing the ongoing synthesis of ML and physics-based methods.

Finally, the combination of bottom-up and top-down approaches for learning NN potentials, i.e., considering information from both the quantum and macroscopic scale, represents an exciting avenue for future research. For top-down approaches, pre-training NN potentials on bottom-up data sets can serve as a sensible extrapolation for the PES in areas unconstrained by the experimental data. In DiffTRe, a pre-trained model could also circumvent the need for a prior potential. Bottom-up trained NN potentials, on the other hand, can be enriched with experimental data, which enables targeted refinement of the potential. This is particularly helpful for systems in which DFT accuracy is insufficient or the generation of a quantum mechanical data set is computationally intractable.

Methods

Differentiable histogram binning. To obtain an informative gradient $\frac{\partial L}{\partial \theta}$, predicted observables need to be continuously differentiable. However, many common observables in MD, including density and structural correlation functions, are computed by discrete histogram binning. To obtain a differentiable observable, the (discrete) Dirac function used in binning can be approximated by a narrow Gaussian probability density function (PDF)³⁴. Similarly, we smooth the non-differentiable cutoff in the definition of ADFs via a Gaussian cumulative distribution function (CDF) centered at the cutoff (details on differentiable density, RDF, and ADF in Supplementary Method 3).

Stress-strain relations. Computing the virial stress tensor σ^V for many-body potentials, e.g., NN potentials, under periodic boundary conditions requires special attention. This is due to the fact that most commonly used formulas are only valid for non-periodic boundary conditions or pairwise potentials⁶⁸. Therefore, we resort to the formulation proposed by Chen et al.⁶⁹, which is well suited for vectorized computations in NN potentials.

$$\sigma^V = \frac{1}{\Omega} \left[- \sum_{k=1}^{N_p} m_k \mathbf{v}_k \otimes \mathbf{v}_k - \mathbf{F}^T \mathbf{R} + \left(\frac{\partial U}{\partial \mathbf{h}} \right)^T \mathbf{h} \right], \quad (13)$$

where N_p is the number of particles, \otimes represents the dyadic or outer product, m_k and \mathbf{v}_k are mass and thermal excitation velocity of particle k , \mathbf{R} and \mathbf{F} are $(N_p \times 3)$ matrices containing all particle positions and corresponding forces, \mathbf{h} is the (3×3) lattice tensor spanning the simulation box, and $\Omega = \det(\mathbf{h})$ is the box volume.

Due to the equivalence of the ensemble-averaged virial stress tensor $\langle \sigma^V \rangle$ and the Cauchy stress tensor σ^0 , we can compute the elastic stiffness tensor from MD

simulations and compare it to continuum mechanical experimental data (details in Supplementary Method 5). In the canonical ensemble, the isothermal elastic stiffness tensor \mathbf{C} can be calculated at constant strain ϵ via the stress fluctuation method⁷¹:

$$C_{ijkl} = \frac{\partial \langle \sigma_{ij}^V \rangle}{\partial \epsilon_{kl}} = \langle C_{ijkl}^B \rangle - \Omega \beta \left(\langle \sigma_{ij}^B \sigma_{kl}^B \rangle - \langle \sigma_{ij}^B \rangle \langle \sigma_{kl}^B \rangle \right) + \frac{N_p}{\Omega \beta} \left(\delta_{ik} \delta_{jl} + \delta_{il} \delta_{jk} \right), \quad (14)$$

with the Born contribution to the stress tensor $\sigma_{ij}^B = \frac{1}{\Omega} \frac{\partial U}{\partial \epsilon_{ij}}$, the Born contribution to the stiffness tensor $C_{ijkl}^B = \frac{1}{\Omega} \frac{\partial^2 U}{\partial \epsilon_{ij} \partial \epsilon_{kl}}$ and Kronecker delta δ_{ij} . Eq. (14) integrates well into DiffTRe by reweighting individual ensemble average terms (Eq. (3)) and combining the reweighted averages afterwards. Implementing the stress fluctuation method in differentiable MD simulations is straightforward: AD circumvents manual derivation of strain-derivatives, which is non-trivial for many-body potentials⁷².

Statistical mechanics foundations. Thermodynamic fluctuation formulas allow to compute the gradient $\frac{\partial L}{\partial \theta}$ from ensemble averages^{73–75}. Specifically, considering a MSE loss for a single observable $O(U_\theta)$ in the canonical ensemble⁷³,

$$\frac{\partial L}{\partial \theta} = 2 \left(\langle O(U_\theta) \rangle - \tilde{O} \right) \left[\left\langle \frac{\partial O(U_\theta)}{\partial \theta} \right\rangle - \beta \left(\left\langle O(U_\theta) \frac{\partial U_\theta}{\partial \theta} \right\rangle - \langle O(U_\theta) \rangle \left\langle \frac{\partial U_\theta}{\partial \theta} \right\rangle \right) \right]. \quad (15)$$

It can be seen that the AD routine in DiffTRe estimates $\frac{\partial L}{\partial \theta}$ by approximating ensemble averages in Eq. (15) via reweighting averages (Derivation in Supplementary Method 5). End-to-end differentiation through the reweighting scheme simplifies optimization by combining obtained gradients from multiple observables. This is particularly convenient for observables that are not merely averages of instantaneous quantities, e.g., the stiffness tensor \mathbf{C} (Eq. (14)).

DimeNet++. We employ a custom implementation of DimeNet++^{11,12} that fully integrates into Jax MD²⁹. Our implementation takes advantage of neighbor lists for efficient computation of the sparse atomic graph. We select the same NN hyperparameters as in the original publication¹² except for the embedding sizes, which we reduced by factor 4. This modification allowed for a significant speed-up while retaining sufficient capacity for the problems considered in this work. For diamond, we have reduced the cutoff to 0.2 nm yielding an atomic graph, where each carbon atom is connected to its four covalently bonded neighbors. A comprehensive list of employed DimeNet++ hyperparameters is provided in Supplementary Method 6.

Data availability

Simulation setups and trained DimeNet++ models have been deposited in <https://github.com/tummmfm/difftre>. The data generated in this study are provided in the paper or in the Supplementary information file.

Code availability

The code for DiffTRe and its application to the three test cases is available at <https://github.com/tummmfm/difftre>⁷⁶.

Received: 2 June 2021; Accepted: 9 November 2021;

Published online: 25 November 2021

References

- Fröhling, T., Bernetti, M., Calonaci, N. & Bussi, G. Toward empirical force fields that match experimental observables. *J. Chem. Phys.* **152**, 230902 (2020).
- Noid, W. G. Perspective: coarse-grained models for biomolecular systems. *J. Chem. Phys.* **139**, 090901 (2013).
- Schütt, K. T., Arbabzadah, F., Chmiela, S., Müller, K. R. & Tkatchenko, A. Quantum-chemical insights from deep tensor neural networks. *Nat. Commun.* **8**, 13890 (2017).
- Noé, F., Tkatchenko, A., Müller, K. R. & Clementi, C. Machine learning for molecular simulation. *Annu. Rev. Phys. Chem.* **71**, 361–390 (2020).
- Behler, J. & Parrinello, M. Generalized neural-network representation of high-dimensional potential-energy surfaces. *Phys. Rev. Lett.* **98**, 146401 (2007).
- Schütt, K. T. et al. SchNet: a continuous-filter convolutional neural network for modeling quantum interactions. in *Advances in Neural Information Processing Systems* Vol. 30, 992–1002 (Curran Associates, Inc., 2017).
- Gilmer, J., Schoenholz, S. S., Riley, P. F., Vinyals, O. & Dahl, G. E. Neural message passing for quantum chemistry. in *Proceedings of the 34th International Conference on Machine Learning*. 1263–1272 (PMLR, 2017).
- Zhang, L., Han, J., Wang, H., Car, R. & Weinan, W. E. DeePCG: constructing coarse-grained models via deep neural networks. *J. Chem. Phys.* **149**, 034101 (2018).

9. Wang, J. et al. Machine learning of coarse-grained molecular dynamics force fields. *ACS Cent. Sci.* **5**, 755–767 (2019).
10. Husic, B. E. et al. Coarse graining molecular dynamics with graph neural networks. *J. Chem. Phys.* **153**, 194101 (2020).
11. Klicpera, J., Groß, J. & Günnemann, S. Directional message passing for molecular graphs. In *8th International Conference on Learning Representations, ICLR* (2020).
12. Klicpera, J., Giri, S., Margraf, J. T. & Günnemann, S. Fast and uncertainty-aware directional message passing for non-equilibrium molecules. in *Machine Learning for Molecules Workshop at NeurIPS* (2020).
13. Qiao, Z., Welborn, M., Anandkumar, A., Manby, F. R. & Miller, T. F. OrbNet: deep learning for quantum chemistry using symmetry-adapted atomic-orbital features. *J. Chem. Phys.* **153**, 124111 (2020).
14. Vlachas, P. R., Zavadlav, J., Praprotnik, M. & Koumoutsakos, P. Accelerated simulations of molecular systems through learning of their effective dynamics. Preprint at <https://arxiv.org/abs/2011.14115> (2021).
15. Jain, A. C. P., Marchand, D., Glensk, A., Ceriotti, M. & Curtin, W. A. Machine learning for metallurgy III: a neural network potential for Al-Mg-Si. *Phys. Rev. Mater.* **5**, 053805 (2021).
16. Ko, T. W., Finkler, J. A., Goedecker, S. & Behler, J. A fourth-generation high-dimensional neural network potential with accurate electrostatics including non-local charge transfer. *Nat. Commun.* **12**, 398 (2021).
17. Ramakrishnan, R., Dral, P. O., Rupp, M. & Von Lilienfeld, O. A. Quantum chemistry structures and properties of 134 kilo molecules. *Sci. Data* **1**, 1–7 (2014).
18. Faber, F. A. et al. Prediction errors of molecular machine learning models lower than hybrid DFT error. *J. Chem. Theory Comput.* **13**, 5255–5264 (2017).
19. Cubuk, E. D. & Schoenholz, S. S. Adversarial forces of physical models. in *3rd NeurIPS workshop on Machine Learning and the Physical Sciences* (2020).
20. Schwalbe-Koda, D., Tan, A. R. & Gómez-Bombarelli, R. Differentiable sampling of molecular geometries with uncertainty-based adversarial attacks. *Nat. Commun.* **12**, 5104 (2021).
21. Zhang, L., Lin, D.-Y., Wang, H., Car, R. & Weinan, E. Active learning of uniformly accurate interatomic potentials for materials simulation. *Phys. Rev. Mater.* **3**, 023804 (2019).
22. Gillan, M. J., Alfè, D. & Michaelides, A. Perspective: how good is DFT for water? *J. Chem. Phys.* **144**, 130901 (2016).
23. Smith, J. S. et al. Approaching coupled cluster accuracy with a general-purpose neural network potential through transfer learning. *Nat. Commun.* **10**, 2930 (2019).
24. Saucedo, H. E., Vassilev-Galindo, V., Chmiela, S., Müller, K. R. & Tkatchenko, A. Dynamical strengthening of covalent and non-covalent molecular interactions by nuclear quantum effects at finite temperature. *Nat. Commun.* **12**, 442 (2021).
25. Cornell, W. D. et al. A second generation force field for the simulation of proteins, nucleic acids, and organic molecules. *J. Am. Chem. Soc.* **117**, 5179–5197 (1995).
26. Oostenbrink, C., Villa, A., Mark, A. E. & Van Gunsteren, W. F. A biomolecular force field based on the free enthalpy of hydration and solvation: the GROMOS force-field parameter sets 53A5 and 53A6. *J. Comput. Chem.* **25**, 1656–1676 (2004).
27. Marrink, S. J., Risselada, H. J., Yefimov, S., Tieleman, D. P. & De Vries, A. H. The MARTINI force field: coarse grained model for biomolecular simulations. *J. Phys. Chem. B* **111**, 7812–7824 (2007).
28. Baydin, A. G., Pearlmutter, B. A., Radul, A. A. & Siskind, J. M. Automatic differentiation in machine learning: a survey. *J. Mach. Learn. Res.* **18**, 1–43 (2018).
29. Schoenholz, S. S. & Cubuk, E. D. JAX MD: A Framework for Differentiable Physics. in *Advances in Neural Information Processing Systems* Vol. 33, 11428–11441 (Curran Associates, Inc., 2020).
30. Doerr, S. et al. TorchMD: a deep learning framework for molecular simulations. *J. Chem. Theory Comput.* **17**, 2355–2363 (2021).
31. Ingraham, J., Riesselman, A., Sander, C. & Marks, D. Learning protein structure with a differentiable simulator. in *7th International Conference on Learning Representations, ICLR* (2019).
32. Goodrich, C. P., King, E. M., Schoenholz, S. S., Cubuk, E. D. & Brenner, M. P. Designing self-assembling kinetics with differentiable statistical physics models. *Proc. Natl Acad. Sci. USA* **118**, e2024083118 (2021).
33. Chen, R. T. Q., Rubanova, Y., Bettencourt, J. & Duvenaud, D. Neural ordinary differential equations. in *Advances in Neural Information Processing Systems* Vol. 31 (Curran Associates, Inc., 2018).
34. Wang, W., Axelrod, S. & Gómez-Bombarelli, R. Differentiable molecular simulations for control and learning. in *ICLR 2020 Workshop on Integration of Deep Neural Models and Differential Equations*. (2020).
35. Norgaard, A. B., Ferkinghoff-Borg, J. & Lindorff-Larsen, K. Experimental parameterization of an energy function for the simulation of unfolded proteins. *Biophys. J.* **94**, 182–192 (2008).
36. Li, D. W. & Brüschweiler, R. Iterative optimization of molecular mechanics force fields from NMR data of full-length proteins. *J. Chem. Theory Comput.* **7**, 1773–1782 (2011).
37. Carmichael, S. P. & Shell, M. S. A new multiscale algorithm and its application to coarse-grained peptide models for self-assembly. *J. Phys. Chem. B* **116**, 8383–8393 (2012).
38. Wang, L. P., Chen, J. & Van Voorhis, T. Systematic parametrization of polarizable force fields from quantum chemistry data. *J. Chem. Theory Comput.* **9**, 452–460 (2013).
39. Reith, D., Pütz, M. & Müller-Plathe, F. Deriving effective mesoscale potentials from atomistic simulations. *J. Comput. Chem.* **24**, 1624–1636 (2003).
40. Lyubartsev, A. P. & Laaksonen, A. Calculation of effective interaction potentials from radial distribution functions: a reverse Monte Carlo approach. *Phys. Rev. E* **52**, 3730–3737 (1995).
41. Binder, K., Heermann, D., Roelofs, L., Mallinckrodt, A. J. & McKay, S. Monte Carlo simulation in statistical physics. *Comput. Phys.* **7**, 156 (1993).
42. Zwanzig, R. W. High-temperature equation of state by a perturbation method. I. Nonpolar gases. *J. Chem. Phys.* **22**, 1420–1426 (1954).
43. Daw, M. S. & Baskes, M. I. Embedded-atom method: derivation and application to impurities, surfaces, and other defects in metals. *Phys. Rev. B* **29**, 6443 (1984).
44. Kingma, D. P. & Ba, J. L. Adam: a method for stochastic optimization. in *3rd International Conference on Learning Representations, ICLR* (2015).
45. McSkimin, H. J., Andreatch, P. & Glynn, P. The elastic stiffness moduli of diamond. *J. Appl. Phys.* **43**, 985–987 (1972).
46. Stillinger, F. H. & Weber, T. A. Computer simulation of local order in condensed phases of silicon. *Phys. Rev. B* **31**, 5262–5271 (1985).
47. Jensen, B. D., Wise, K. E. & Odegard, G. M. Simulation of the elastic and ultimate tensile properties of diamond, graphene, carbon nanotubes, and amorphous carbon using a revised reaxFF parametrization. *J. Phys. Chem. A* **119**, 9710–9721 (2015).
48. Dolling, G. & Cowley, R. A. The thermodynamic and optical properties of germanium, silicon, diamond and gallium arsenide. *Proc. Phys. Soc.* **88**, 463 (1966).
49. Barnard, A. S., Russo, S. P. & Leach, G. I. Nearest neighbour considerations in stillinger-weber type potentials for diamond. *Mol. Simul.* **28**, 761–771 (2002).
50. Scherer, C. & Andrienko, D. Understanding three-body contributions to coarse-grained force fields. *Phys. Chem. Chem. Phys.* **20**, 22387–22394 (2018).
51. Soper, A. K. & Benmore, C. J. Quantum differences between heavy and light water. *Phys. Rev. Lett.* **101**, 065502 (2008).
52. Berendsen, H. J. C., Postma, J. P. M., van Gunsteren, W. F. & Hermans, J. Interaction models for water in relation to protein hydration. in *Intermolecular forces*. (ed. Pullman, B.) 331–342 (Springer, 1981).
53. Wu, Y., Tepper, H. L. & Voth, G. A. Flexible simple point-charge water model with improved liquid-state properties. *J. Chem. Phys.* **124**, 024503 (2006).
54. Errington, J. R. & Debenedetti, P. G. Relationship between structural order and the anomalies of liquid water. *Nature* **409**, 318–321 (2001).
55. Mills, R. Self-diffusion in normal and heavy water in the range 1–45°. *J. Phys. Chem.* **77**, 685–688 (1973).
56. Wang, H., Junghans, C. & Kremer, K. Comparative atomistic and coarse-grained study of water: what do we lose by coarse-graining? *Eur. Phys. J. E* **28**, 221–229 (2009).
57. Matysiak, S., Clementi, C., Praprotnik, M., Kremer, K. & Delle Site, L. Modeling diffusive dynamics in adaptive resolution simulation of liquid water. *J. Chem. Phys.* **128**, 024503 (2008).
58. Molinero, V. & Moore, E. B. Water modeled as an intermediate element between carbon and silicon. *J. Phys. Chem. B* **113**, 4008–4016 (2009).
59. Chan, H. et al. Machine learning coarse grained models for water. *Nat. Commun.* **10**, 379 (2019).
60. Distasio, R. A., Santra, B., Li, Z., Wu, X. & Car, R. The individual and collective effects of exact exchange and dispersion interactions on the ab initio structure of liquid water. *J. Chem. Phys.* **141**, 084502 (2014).
61. Thaler, S., Praprotnik, M. & Zavadlav, J. Back-mapping augmented adaptive resolution simulation. *J. Chem. Phys.* **153**, 164118 (2020).
62. Belbute-Peres, F. D. A., Smith, K. A., Allen, K. R., Tenenbaum, J. B. & Kolter, J. Z. End-to-end differentiable physics for learning and control. in *Advances in Neural Information Processing Systems* Vol. 31 (Curran Associates, Inc., 2018).
63. Innes, M. et al. A differentiable programming system to bridge machine learning and scientific computing. Preprint at <https://arxiv.org/abs/1907.07587> (2019).
64. Hu, Y. et al. DiffTaichi: differentiable programming for physical simulation. in *8th International Conference on Learning Representations, ICLR* (2020).
65. Degraeve, J., Hermans, M., Dambre, J. & Wyffels, F. A differentiable physics engine for deep learning in robotics. *Front. Neurobot.* **13**, 6 (2019).
66. Holl, P., Koltun, V. & Thurey, N. Learning to control PDEs with differentiable physics. in *8th International Conference on Learning Representations, ICLR* (2020).

67. Schäfer, F., Kloc, M., Bruder, C. & Lörch, N. A differentiable programming method for quantum control. *Mach. Learn. Sci. Technol.* **1**, 35009 (2020).
68. Thompson, A. P., Plimpton, S. J. & Mattson, W. General formulation of pressure and stress tensor for arbitrary many-body interaction potentials under periodic boundary conditions. *J. Chem. Phys.* **131**, 154107 (2009).
69. Chen, X. et al. TensorAlloy: an automatic atomistic neural network program for alloys. *Comput. Phys. Commun.* **250**, 107057 (2020).
70. Subramanian, A. K. & Sun, C. T. Continuum interpretation of virial stress in molecular simulations. *Int. J. Solids Struct.* **45**, 4340–4346 (2008).
71. Van Workum, K., Yoshimoto, K., De Pablo, J. J. & Douglas, J. F. Isothermal stress and elasticity tensors for ions and point dipoles using Ewald summations. *Phys. Rev. E - Stat. Nonlinear, Soft Matter Phys.* **71**, 061102 (2005).
72. Van Workum, K., Gao, G., Schall, J. D. & Harrison, J. A. Expressions for the stress and elasticity tensors for angle-dependent potentials. *J. Chem. Phys.* **125**, 144506 (2006).
73. Di Pierro, M. & Elber, R. Automated optimization of potential parameters. *J. Chem. Theory Comput.* **9**, 3311–3320 (2013).
74. Wang, L. P. et al. Systematic improvement of a classical molecular model of water. *J. Phys. Chem. B* **117**, 9956–9972 (2013).
75. Wang, L. P., Martinez, T. J. & Pande, V. S. Building force fields: an automatic, systematic, and reproducible approach. *J. Phys. Chem. Lett.* **5**, 1885–1891 (2014).
76. Thaler, S. & Zavadlav, J. Learning neural network potentials from experimental data via Differentiable Trajectory Reweighting. <https://github.com/tummmf/diffre>, <https://doi.org/10.5281/zenodo.5643099> (2021).

Author contributions

S.T. conceptualized, implemented, and applied the DiffTRe method and conducted MD simulations as well as postprocessing. S.T. and J.Z. planned the study, analyzed and interpreted the results, and wrote the paper.

Funding

Open Access funding enabled and organized by Projekt DEAL.

Competing interests

The authors declare no competing interests.

Additional information

Supplementary information The online version contains supplementary material available at <https://doi.org/10.1038/s41467-021-27241-4>.

Correspondence and requests for materials should be addressed to Stephan Thaler or Julija Zavadlav.

Peer review information *Nature Communications* thanks Ekin Cubuk and the other, anonymous, reviewer(s) for their contribution to the peer review of this work.

Reprints and permission information is available at <http://www.nature.com/reprints>

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.



Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this license, visit <http://creativecommons.org/licenses/by/4.0/>.

© The Author(s) 2021