Efficient parametrization of the atomic cluster expansion

Anton Bochkarev,* Yury Lysogorskiy,† Sarath Menon, Minaam Qamar, Matous Mrovec, and Ralf Drautz
ICAMS, Ruhr-Universität Bochum, Bochum, Germany

(Received 22 October 2021; revised 18 December 2021; accepted 6 January 2022; published 24 January 2022)

The atomic cluster expansion (ACE) provides a general, local, and complete representation of atomic energies. Here we present an efficient framework for parametrization of ACE models for elements, alloys, and molecules. To this end, we first introduce general requirements for a physically meaningful description of the atomic interaction, in addition to the usual equivariance requirements. We then demonstrate that ACE can be converged systematically with respect to two fundamental characteristics—the number and complexity of basis functions and the choice of nonlinear representation. The construction of ACE parametrizations is illustrated for several representative examples with different bond chemistries, including metallic copper, covalent carbon, and several multicomponent molecular and alloy systems. We discuss the Pareto front of optimal force to energy matching contributions in the loss function, the influence of regularization, the importance of consistent and reliable reference data, and the necessity of unbiased validation. Our ACE parametrization strategy is implemented in the freely available software package `pacemaker` that enables largely automated and GPU accelerated training. The resulting ACE models are shown to be superior or comparable to the best currently available ML potentials and can be readily used in large-scale atomistic simulations.

DOI: 10.1103/PhysRevMaterials.6.013804

I. INTRODUCTION

Recent years have witnessed a rapid advancement of machine learning (ML) in atomistic modeling [1–18]. Various ML interatomic potentials were developed for elemental as well as multicomponent systems, usually as surrogate models for electronic structure calculations based on the many-electron Schrödinger equation [19–33]. The ML models seek to provide an accurate representation of energies and forces while being computationally efficient to enable computations of complex thermodynamic, kinetic, and mechanical properties.

Some ML models were constructed with the specific aim to describe particular atomic configurations, chemical compositions, or ambient conditions. Such potentials were applied, for instance, for the computation of thermodynamic properties of high entropy alloys [34], hydrogen under extreme conditions [35], or specific phase transformations [36,37]. A significantly more challenging task is to build transferable models that are able to address states not foreseen during model construction, both in terms of structure and composition. Four key factors are critical for obtaining accurate and transferable ML interatomic potentials: (i) a general yet physically consistent machine learning model, (ii) an extensive, diverse and high-quality reference dataset, (iii) a robust and efficient training procedure, and (iv) a thorough validation including assessment of a model's uncertainty.

The recently developed atomic cluster expansion (ACE) provides a general and mathematically complete representation of the properties of interacting atoms [26,38,39]. The ACE basis is invariant under fundamental symmetry operations, such as translation, rotation, inversion and permutation, or more generally equivariant for the expansion of nonscalar quantities. In addition, ACE is scale and basis invariant (see below), which guarantees that the ACE models fulfill the basic requirements of physically consistent models. Due to its completeness, ACE was shown [26,38–40] to be able to represent many classical, ML and neural network (NN) interatomic potentials as well as important descriptors including Behler's and Parrinello's symmetry functions [19], SOAP and GAP [20,41], SNAP [21], and MTP [22].

ACE has been implemented within the `LAMMPS` molecular dynamics simulation software package [42] and its numerical efficiency is competitive or superior to other ML potentials. Recent ACE parametrizations for metallic Cu and covalently bonded Si provide more accurate potentials with better transferability and faster evaluation times than any available classical or ML models [40].

ML potentials are usually constructed by fitting to large reference datasets that contain mostly energies and forces for different atomic configurations. The quality and diversity of the training dataset are decisive for the accuracy and transferability of a ML potential. Still, the most common way to construct the reference dataset is based on previous knowledge of the system at hand, by empirically selecting the most relevant configurations and supplying randomness via molecular dynamics (MD) simulations. Active learning techniques have been employed recently to reduce the amount of required reference data and to generate it in an unbiased way

*anton.bochkarev@rub.de
†yury.lysogorskiy@icams.rub.de

[31,37,43–45]. This is especially relevant for non-equilibrium configurations that are of crucial importance when training transferable ML potentials.

The training of ML models is typically carried out by varying model parameters to minimize a loss function that measures the difference between the reference data and the model predictions. A robust training procedure leads to a smooth, monotonous decrease of the loss function with increasing model complexity and results in a converged and accurate representation of the potential energy. This task necessitates robust and well-designed software for both the efficient implementation of a ML interatomic potential and its numerically stable optimization with respect to a multitude of training parameters.

Finally, an unbiased validation is critical for assessing the accuracy and transferability of a ML potential. A minimal validation entails error analysis for a test dataset that was not part of the training. However, such unspecific analysis will often miss subtle but important deficiencies of the model, for example, the inability to capture small energy differences between allotropes or structurally similar phases. Reliable detection of problematic atomic configurations requires more elaborate validation procedures, such as high-temperature MD simulations or Monte Carlo (MC) sampling. In addition, a rapid "on-the-fly" validation during training is helpful to monitor the progress of the optimization. The assessment of a model's uncertainty based on ensemble learning [46] helps to ensure transferability and accurate predictions.

We present a general scheme for the construction of efficient and transferable ACE models for multicomponent materials and provide a dedicated computer code, which we call `pacemaker`. The `pacemaker` implementation contains largely automated training procedures and utilizes advanced optimization algorithms provided by the `tensorpotential` library. The `tensorpotential` library implements automated differentiation for general interatomic models and enables GPU acceleration via the TensorFlow software platform [47]. The pacemaker code is tightly integrated with the PACE/LAMMPS implementation [40] to streamline the development and validation of ACE models with their immediate application for complex atomistic studies, such as large-scale MD simulations or automated computations of phase diagrams [48]. A schematic chart summarizing the ACE development workflow is shown in Fig. 1.

The paper is structured as follows. In Sec. II a brief introduction to ACE theory is given, and the key symmetry and invariance requirements of ACE models are discussed. In Sec. III the parametrization setup is summarized, before the crucial aspects of ACE fitting strategy and convergence criteria are presented in Sec. IV. In Sec. V the construction of ACE models is demonstrated for several multicomponent systems. We conclude in Sec. VI. Detailed appendices provide additional information on all necessary functions and parameters.

II. ATOMIC CLUSTER EXPANSION

The atomic cluster expansion provides a formally complete description of atomic properties [26,38,39]. An atomic property p that is a function of the local atomic environment

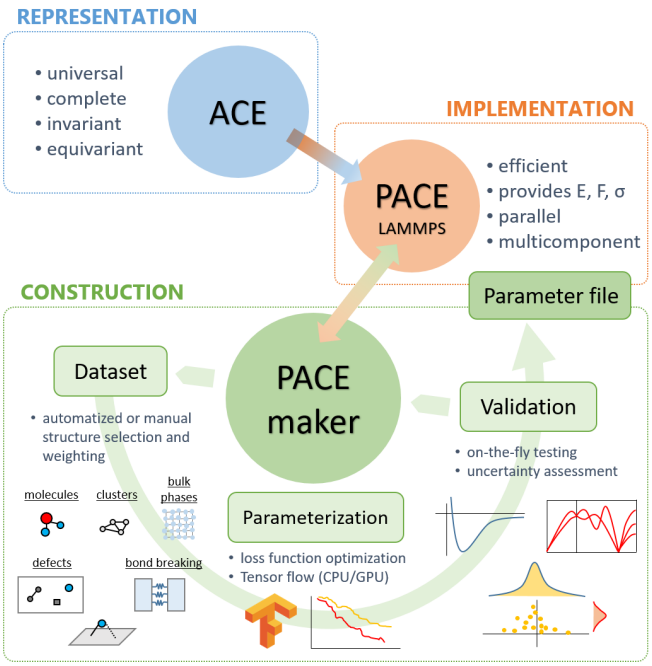

FIG. 1. Block scheme of the main `pacemaker` workflow.

of atom i is expanded as

$$\varphi_i^{(p)} = \sum_{\boldsymbol{v}} c_{\boldsymbol{v}}^{(p)} \boldsymbol{B}_{i\boldsymbol{v}}, \tag{1}$$

with expansion coefficients $c_{\boldsymbol{v}}^{(p)}$ and basis functions $\boldsymbol{B}_{i\boldsymbol{v}}$. The basis functions $\boldsymbol{B}_{i\boldsymbol{v}}$ depend on the atomic positions and species. Dependencies on other variables, such as magnetism or charge, can be added, too. The basis functions fulfill the fundamental translation, rotation, inversion, and permutation (TRIP) invariance for the representation of scalar variables, or equivariance for the expansion of vectorial or tensorial quantities. The basis functions are formed as linear combinations of products of single-bond basis functions and the number of products determines the body order/correlation order of a basis function.

A. Double convergence

Because of the completeness of the ACE representation, a linear model with a single density formally suffices for the expansion of the energy

$$E_i = \varphi_i^{(1)}. \tag{2}$$

This representation was employed in the construction of a recent Si ACE [40,49]. For faster convergence the energy is partitioned into different contributions. For example, if the energy can be separated into two contributions $E_i = E_{1i} + E_{2i}$, where the first part is expanded linearly $E_{1i} = \varphi_i^{(1)}$, while the square of the second part is expanded as $E_{2i}^2 = \varphi_i^{(2)}$, one obtains

$$E_i = \varphi_i^{(1)} + \sqrt{\varphi_i^{(2)}}, \tag{3}$$

which provides an extension of the Finnis-Sinclair potential [50] and second moment tight-binding methods [51–53] to

general many-atom interactions. This representation was used recently for the parametrization of Cu [40].

In the most general way, the atomic energy is represented as a function of P different properties

$$E_i = \mathcal{F}\big(\varphi_i^{(1)}, \ldots, \varphi_i^{(P)}\big), \tag{4}$$

with a general nonlinear embedding function $\mathcal{F}$. Hence, the accuracy of the expansion depends fundamentally on the number and complexity of the basis functions $\boldsymbol{B}_{i\boldsymbol{v}}$ but also on the number of atomic properties P and the nonlinear function $\mathcal{F}$. This opens up two independent mechanisms for optimization of ACE models that we term "double convergence", since ACE may be converged for each property with respect to the basis functions [cf. Eq. (1)] as well as the number of atomic properties and the functional form of $\mathcal{F}$ [cf. Eq. (4)].

Equation (4) also provides a link to neural network potentials. For example, the symmetry functions of Behler and Parrinello [19] can be expressed in the form of ACE descriptors $\varphi_i^{(p)}$ that comprise only basis functions of body order two and three, while the neural network architecture is described by a nonlinear function $\mathcal{F}$ that transforms the input from the P descriptors into the atomic energy. Therefore, neural network potentials may be integrated into the ACE framework as low-body-order ACE expansions combined with a complex nonlinear function $\mathcal{F}$.

In general, ML potentials that only employ low body order descriptors are not complete [54]. Therefore, for the current ACE development, we focus on simple nonlinear functions $\mathcal{F}$ but expand the atomic properties to a high, in principle arbitrary, body order. In this way, we exploit the completeness of ACE to ensure that the expansion can be converged. The selection of suitable nonlinear functions $\mathcal{F}$ is facilitated by requiring ACE models to be scale-invariant, universal, and invariant under basis transformations (SUB invariance), in addition to the TRIP invariance discussed above.

B. Scale invariance

Scale invariance means that the atomic properties as well as the embedding function $\mathcal{F}$ may not possess any inherent or natural scale. There must exist scaling functions $f_\lambda^{(p)}$, which scale the parameters as $c_{\boldsymbol{v}}^{(p)} \to f_\lambda^{(p)} c_{\boldsymbol{v}}^{(p)}$ and lead to a homogeneous linear scaling such that

$$\lambda \mathcal{F}\big(\varphi_i^{(1)}, \ldots, \varphi_i^{(P)}\big) = \mathcal{F}\big(f_\lambda^{(1)}\varphi_i^{(1)}, \ldots, f_\lambda^{(P)}\varphi_i^{(P)}\big). \tag{5}$$

For example, for the Finnis-Sinclair potential $f_\lambda^{(1)} = \lambda$ and $f_\lambda^{(2)} = \lambda^2$. Other examples of scale-invariant functions are polynomial expansions

$$\mathcal{F} = \sum_p \big(\varphi_i^{(p)}\big)^{m_p}, \tag{6}$$

or recursive expressions, such as

$$\mathcal{F} = \sqrt{\varphi_i^{(1)}\sqrt{\varphi_i^{(2)}\sqrt{\varphi_i^{(3)}\sqrt{\ldots}}}}. \tag{7}$$

The scale invariance is not an abstract requirement. It is necessary to ensure identical representations for arbitrary energy units (eV, Hartree, or Ry) without implicitly resorting to a particular internal energy scale of the reference data (e.g., the cohesive energy of the ground state structure for a particular chemistry). Note that, unlike the energy scale, the length scale needs to be defined as dimensionless, for example, based on the nearest neighbor distance in a particular crystal structure. This means that the length scale and the cutoff radius will depend on chemistry, which is physically correct.

C. Universality

The universality of ACE models means that the nonlinear function $\mathcal{F}$ does not depend on any particular chemistry or atomic structure, but that both characteristics enter the model through the ACE atomic properties φ_i only. This facilitates the development of transferable multicomponent models that can be naturally built up from their simpler constituents (binaries from unaries, ternaries from unaries and binaries, etc.). In practice, this limits the choice of possible functions $\mathcal{F}$ as they may not contain species-dependent parameters or be adapted to any particular chemistry or structure.

D. Basis invariance

The final condition imposed on ACE models is basis invariance. It implies that a transformation of ACE basis functions leaves the expansion unchanged provided that an equivalent opposite transformation is carried out for the expansion coefficients. While individual ACE basis functions are not invariant under basis transformations, the atomic properties are basis-invariant as they are written as a sum over the product of expansion coefficients and basis functions. Therefore the atomic properties are the fundamental descriptors in the ACE framework.

Overall, the SUB-TRIP invariance guarantees the ACE models to be physically sound. In contrast, many popular ML methods, such as neural network potentials, are not SUB-TRIP invariant in their commonly used forms, which may lead to poorer transferability and extendability.

III. ACE PARAMETRIZATION SETUP

A. Training parameters

The linear expansion in rotationally invariant or equivariant basis functions in Eq. (1) can be transformed into another basis as [26,40]

$$\varphi_i^{(p)} = \sum_{\boldsymbol{v}} c_{\boldsymbol{v}}^{(p)} \boldsymbol{B}_{i\boldsymbol{v}} = \sum_{\boldsymbol{v}} \tilde{c}_{\boldsymbol{v}}^{(p)} \boldsymbol{A}_{i\boldsymbol{v}}, \tag{8}$$

for faster evaluation (see Appendix B3 for details). The permutation-invariant many-body basis functions $\boldsymbol{A}_{i\boldsymbol{v}}$ of body order $\nu + 1$ are defined as a product [26,40]

$$\boldsymbol{A}_{i\boldsymbol{v}} = \prod_{t=1}^{\nu} A_{i\boldsymbol{v}_t}. \tag{9}$$

These products are generally not invariant or equivariant under rotation and inversion, but invariance/equivariance is obtained by multiplication with generalized Glebsch-Gordan coefficients as summarized in Appendix D. This results in the basis functions $\boldsymbol{B}_{i\boldsymbol{v}}$. The basis functions $\boldsymbol{B}_{i\boldsymbol{v}}$ are independent and have the appropriate symmetries, therefore training is

carried out in this basis, and the parameters $c_{\boldsymbol{v}}^{(p)}$ are optimized. After training, the expansion coefficients are transformed to $\tilde{c}_{\boldsymbol{v}}^{(p)}$ and the basis functions $\boldsymbol{A}_{i\boldsymbol{v}}$ are used in the PACE implementation as they can be evaluated more efficiently [40].

The basis functions $\boldsymbol{A}_{i\boldsymbol{v}}$ are formed as products of the atomic base $A_{i\boldsymbol{v}}$, which is obtained by projecting single-bond basis functions on the atomic density,

$$A_{i\boldsymbol{v}} = \langle \phi_{\boldsymbol{v}} | \rho_i \rangle. \tag{10}$$

The atomic density is given by

$$\rho_i = \sum_{j}^{j \neq i} \delta_{\mu\mu_j} \delta(r - r_{ji}), \tag{11}$$

where μ indicates chemical species, μ_j the species of atom j and r_{ji} the vector from atom i to j. The single-bond or one-particle basis functions $\phi_{\boldsymbol{v}}$ are taken as irreducible basis functions of the rotation group, i.e., a radial function R_{nl} multiplied with a spherical harmonics Y_{lm},

$$\phi_{\mu_i\mu_j nlm} = R_{nl}^{\mu_i\mu_j}(r_{ji}) Y_{lm}(\hat{\boldsymbol{r}}_{ji}). \tag{12}$$

In the description of ϕ, we expanded the compressed index $\boldsymbol{v}$ into an explicit list of indices that characterize the chemical species μ, the type of radial functions and spherical harmonics. The radial functions depend on the pairwise distance r_{ji} between atoms i and j.

In summary, the complexity of an ACE model is governed by the type of embedding function $\mathcal{F}$, the number P of densities $\varphi_i^{(p)}$, and the number and body-order of basis functions and their corresponding expansion coefficients $c_{\boldsymbol{v}}^{(p)}$. ACE training includes the optimization of the expansion coefficients $c_{\boldsymbol{v}}^{(p)}$ and, optionally, the radial functions $R_{nl}^{\mu_i\mu_j}$ via optimizing coefficients of the linear expansion of the radial basis function $g_k(r_{ji})$, see Appendix C. The basis functions $\boldsymbol{B}_{\boldsymbol{v}}$ can be ordered sequentially according to their complexity, so that ACE models can be converged systematically and automatically in a hierarchical manner (see Sec. IV D). In addition to these fundamental attributes, ACE models contain further parameters that need to be fixed before optimization and are not modified during training. These include pairwise inner and outer cutoff distances, which depend on the particular chemistry at hand, the representation of the radial functions, core repulsion parameters, etc. A detailed overview of these parameters and settings is provided in the appendices.

Finally, the linear expansion of atomic properties $\varphi_i^{(p)}$ enables to parametrize readily ACE models for multicomponent systems based on available subsystems. For example, when parametrizations for elements A and B are available, an ACE model for the binary A-B system requires an additional parametrization of the expansion coefficients $c_{\boldsymbol{v}}^{(p)}$ only for basis functions in which both A and B atoms appear. Similarly, for a ternary system only three- and higher-body basis functions comprising three different elements need to be added provided that elemental and binary ACE models exist.

B. Implementation

To facilitate ACE parametrizations we developed a dedicated open-source computer code `pacemaker` that is able to optimize multicomponent ACE models of different complexity using diverse training datasets. The training data is imported from a tailored *pandas* [55,56] dataframe, which needs to be generated beforehand. This enables data control and manipulation outside of the `pacemaker` workflow. Moreover, `pacemaker` offers options to sample the training set according to various training criteria (e.g., specific energy and force ranges, random sampling, etc.) and an automatic generation of a testing dataset for immediate model analysis.

The `pacemaker` code can be used to explore the double convergence of nonlinear ACE. The present implementation contains several functional forms for $\mathcal{F}$, including the basic linear energy scaling as well as a polynomial expansion. Other scale-invariant nonlinear functions for more densities can readily be implemented. The choice of $\mathcal{F}$ and the number of densities is important for an efficient convergence of ACE as demonstrated below. For a given $\mathcal{F}$ and the number of atomic properties P, `pacemaker` determines the expansion coefficients $c_{\boldsymbol{v}}^{(p)}$ and optimizes the radial functions $R_{nl}^{\mu_i\mu_j}$.

The `pacemaker` workflow and technical details of the implementation are given in Appendix A. `pacemaker` makes extensive use of the TensorFlow (TF) symbolic math library [57]. TF enables to evaluate all derivatives with respect to the training parameters analytically and is fully compatible with the C++ PACE/LAMMPS implementation [40]. The `pacemaker` implementation works on both CPU and GPU-based architectures and provides a significant speed-up of up to several orders of magnitude on modern GPU processing units.

C. Training datasets

Datasets for ACE training need to contain a sufficiently large number of different atomic configurations with their corresponding energies and forces. Standard existing datasets can be used as well as custom-made databases computed using high-level electronic structure methods. The electronic structure calculations must be carried out in a consistent manner, for example, with respect to k-mesh density, pseudopotentials and plane-wave energy cutoffs. Typically, we use 10^4 to 10^5 structures with a million and more different atomic environments to provide a comprehensive sampling of possible configurations. ACE models can also be trained on much smaller datasets. For these cases, `pacemaker` provides regularization parameters to control possible fit irregularities.

In this paper, we employed several density functional theory (DFT) computed datasets of different sizes and origins to demonstrate the versatility of `pacemaker`. For two distinct elements—the transition metal Cu and the covalent element C—we constructed our own extensive DFT datasets using the FHI-aims [58,59] and VASP codes [60,61], respectively. In addition, we used publicly available datasets for the ethanol molecule [62], ten binary alloys [5], and a high-entropy alloy (HEA) [63]. The main characteristics of all datasets are listed in Table I.

TABLE I. A description of datasets used in this paper.

Designation	Description	No. of structures		No. of atoms	
		Train	Test	Train	Test
	Datasets generated in this paper				
Cu-I	fcc, bcc, hcp; E-V curves (±10% from equilibrium), elastic constant and phonons	105	105	368	667
Cu-II	Cu-I + randomly selected structures within 1 eV/at above the ground state	1000	1000	11309	10541
Cu-III	Cu-II + long range E-V curves, 1000 fcc surface structures (strained, shaken)	9422	9442	81352	78485
C-I	diamond, graphene, graphite, dimer; E-V curves, elastic constant and phonons	129	129	378	382
C-II	C-I + randomly selected bulk structures within 3 eV/at above the ground state	883	883	7299	7975
C-III	C-II + long range E-V curves, nanotubes, fullerenes, bulk structures up to 4 eV/at above the ground state	4453	4453	51308	53031
	Publicly available datasets				
Ethanol	revMD-17 (Ref. [62])	1000	99000	9000	891000
10-B	10 metallic binary alloys (Ref. [5])	10000	5952	73182	43263
HEA	HEA rand.II (Ref. [63])	200	20	36000	3600

To examine the influence of dataset characteristics and size on the training procedure and the quality of the resulting ACE models, the two elemental datasets were further split into three nested subsets of increasing size and complexity. The smallest datasets (marked as Cu-I and C-I) contained only the fundamental bulk structures under small deformations relevant for the evaluation of elastic and vibrational properties (100 structures in total). The intermediate datasets (marked as Cu-II and C-II) comprised the smallest dataset and additional, randomly distorted structures within 1 eV/atom (Cu) or 3 eV/atom (C) energy above the ground state (∼1000 structures in total). Finally, the Cu-III and C-III datasets were composed of the intermediate datasets plus structures under large volume deformations, defect structures and, for C, molecular fullerene and nanotube allotropes. For all these datasets, complementary test datasets of the same size and character were generated as well (cf. Table I).

IV. FITTING STRATEGY AND CONVERGENCE

A. Model convergence

The parametrization of ML models entails optimization of trainable parameters to minimize the loss function that measures the error as compared to reference data. Since ML models are typically parameterized using large numbers of reference data and comprise a large number of adjustable parameters, it is possible to devise general rules for model convergence.

We assume that the reference data is sufficiently rich to sample different local atomic environments, the irreducible error in the reference data is negligible, and regularization contributions to the loss function are small. We further exploit the completeness of ACE (or any other complete ML potential) that in principle enables to achieve arbitrarily small errors.

We first consider the change of model error with the number of training data, often referred to as the learning curve. It can be expected that more training data lead to more accurate models. For a large number N of training data the learning curve decays algebraically [64,65] and the test root mean square error (RMSE) can be written as

$$\mathrm{RMSE}(N) = CN^{-Q}, \tag{13}$$

where C and Q are constants. This can be understood from the central limit theorem [66], according to which $\sigma \propto 1/\sqrt{N}$, and therefore we expect $Q = 1/2$. This is consistent with numerical values obtained from literature [64,67].

A second measure of convergence is based on increasing model complexity. We follow [65] and define the dependence of RMSE on the number of model parameters as the feature curve. As for the learning curve, for large N often algebraic convergence of the feature curve may be assumed [22],

$$\mathrm{RMSE}(n) = cn^{-q}, \tag{14}$$

where c and q are constants and n the number of parameters in the model. The orthogonality of ACE basis functions means that their contributions are independent for sufficiently rich reference data. As the error is evaluated by sampling the distribution of basis function contributions, the central limit theorem again implies $q = 1/2$ for large n. This should hold for other ML models provided that the model parameters are independent. If one conceptually obtains forces from finite displacements, then two energies need to be matched simultaneously for accurate force predictions. Hence, one may expect a quadratically slower convergence of force errors with $q = 1/4$, as observed for example in Fig. 4 or in Ref. [22].

An efficient model is then characterized by small values of the prefactors c or C and large values of the exponents q or Q, where the values of the exponents are decisive. It should be noted that values of the exponent q significantly larger than $1/2$ can arise due to incomplete or limited reference data that do not cover atomic

TABLE II. Summary of ACE configurations: cutoff radius (r_c), type of radial basis functions, order ν, n_{max}, l_{max} and maximum number of functions per element (# func./element) for each order ν for this configuration. For the hierarchical fits, the number of functions corresponds to the final iteration.

System/elements	r_c (Å)	Radial basis function	ν order	n_{max}	l_{max}	# func./element
Regularization, nonlinear, and force contribution fits						
Cu (Sec. IV B, IV E)	7.0	ChebExpCos	4	15/3/2/1	0/2/2/1	15/18/26/3
C (Sec. IV C)	4.5	ChebExpCos	3	15/5/4	0/2/2	15/45/160
Ethanol (Sec. IV C)	4.0	ChebExpCos	4	15/3/2/1	0/2/2/1	45/135/182/15
Hierarchical fit						
Cu (Sec. IV D)	7.0	ChebExpCos	4	15/5/4/3	0/4/3/2	15/75/304/31
Ethanol (Sec. V A)	4.5	ChebPow	5	20/6/3/1/1	0/5/3/2/1	60/483/765/141/57
10 binaries[1] (Sec. V B)	7.0	ChebExpCos	4	15/3/2/1	0/3/2/1	
Ag, Co, Mg, Nb, Ti, V						30/84/86/3
Al						75/264/266/3
Cu, Fe						60/204/206/3
HEA (Sec. V C)	7.0	ChebExpCos	5	15/3/2/1/1	0/2/2/1/1	75/300/300/10/35

[1]Only up to two species interactions were considered

configurations homogeneously and exhaustively. On the other hand, small values of Q or q often hint at inefficient and redundant model parameters or contributions to the reference data that cannot be captured by the model. For example, deep neural networks generally have an excess of parameters, which leads to poor convergence with increasing n, i.e., a small value of q. Numerical values that we extracted from Refs. [68,69] are approximately $q \approx 0.2$. Loss functions that combine energy and force errors are also expected to converge with $q < 1/2$.

Small values of the prefactors c or C can be achieved by feature engineering. By starting from a physically motivated model, such as the Finnis-Sinclair potential, Eq. (3), and specifically designed basis functions, Eq. (C1), we aim at models that are as accurate as possible with a minimum number of parameters (i.e., in the pre-asymptotic regime). Basis function optimization was also used by Willatt *et al.* [67] to achieve efficient learning curves.

B. Loss function

The loss function $\mathcal{L}$ as implemented in `pacemaker` comprises a weighted mean square error of the difference between energies and forces in the reference dataset and those predicted by ACE,

$$\mathcal{L} = (1-\kappa)\Delta_E^2 + \kappa\Delta_F^2 + \Delta_{\text{coeff}} + \Delta_{\text{rad}}. \qquad (15)$$

The relative contribution of the errors of energies Δ_E^2 and forces Δ_F^2 is weighted with the parameter κ. Furthermore, the regularization contribution Δ_{coeff} limits the absolute values of the expansion coefficients to help ensure smoothness of the energy hypersurface, while the regularization contribution Δ_{rad} ensures smooth radial functions. A detailed description of all terms in Eq. (15) is given in Appendix E. The influence of regularization parameters is presented in Sec. IV E.

The parameter κ in Eq. (15) enables to balance energy and force matching contributions to the loss function. This is demonstrated on the Pareto front shown in Fig. 2. Different ACE models were obtained by linearly varying κ from zero (energy error only, no force matching) to one (force error only, no energy matching). Training employed the Cu-III dataset and ACE with only 62 basis functions to ensure that the number of reference data is much larger than the number of ACE parameters. Details of the ACE configuration are given in Table II (see also Appendix G). The optimal value of κ for this combination of dataset and ACE model is ≈0.2. This result is consistent with the recent study of Krishnapriyan *et al.* [70] who suggested to choose the relative force and weight contributions based on their respective variances. For the Cu-III dataset, we estimated $\kappa = \sigma(E_{\text{pa}})^2/(\sigma(E_{\text{pa}})^2 + \sigma(F_{\text{n,i}})^2) \approx 0.33$, close to the Pareto optimum found numerically.

C. Nonlinear embedding and radial functions

The nonlinear embedding function $\mathcal{F}$ can improve convergence of ACE significantly. We generated four different parametrizations for each of the three systems Cu, C, and ethanol. The parametrizations differed by their choice for $\mathcal{F}$, where we partitioned the energy into one or two atomic properties, and by optimizing or not optimizing radial functions. A comparison of relative energy and force RMSE for all models is shown in Fig. 3.

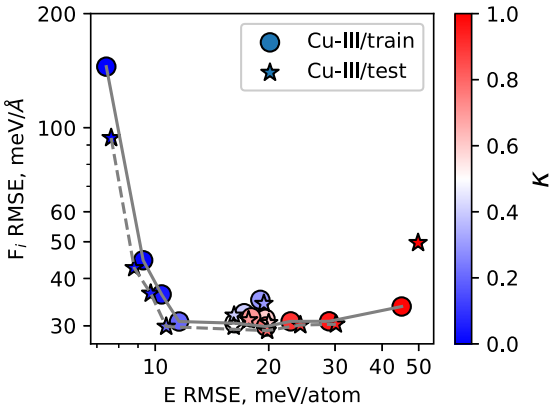

FIG. 2. Energy and force RMSE for train and test errors as a function of the relative energy and force contribution κ to the loss function for the Cu-III dataset.

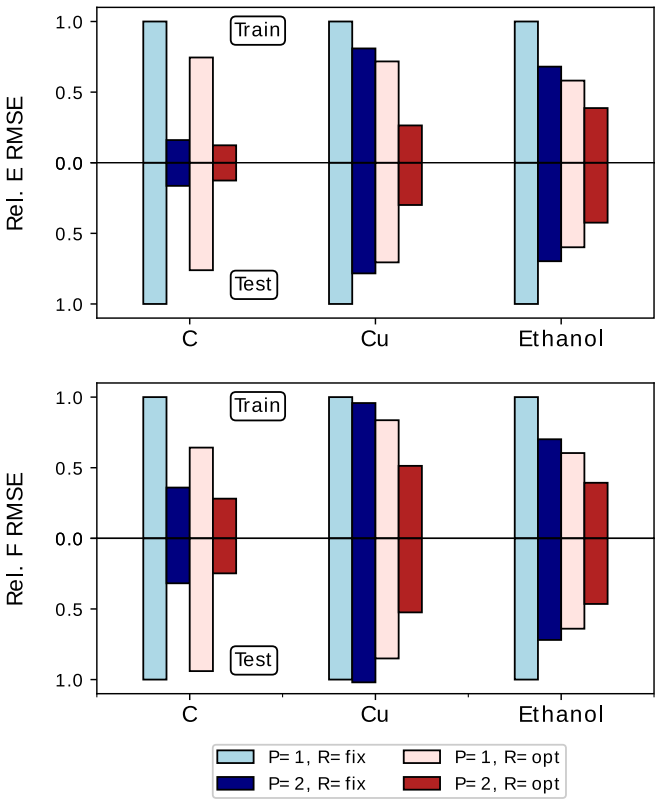

FIG. 3. Relative energy and force RMSE for different ACE parametrizations for Cu, C, and ethanol. See text for details.

A linear fit with only one atomic property, $P = 1$, represents the simplest reference models in which only the ACE expansion coefficients $c_{\boldsymbol{v}}^{(1)}$ were optimized while the radial functions, chosen as the scaled Chebyshev polynomials (see Appendix C), were kept fixed ($R = \text{fix}$). The second set of models was constructed in the same way except of using two atomic properties, $P = 2$, and the Finnis-Sinclair form for $\mathcal{F}$ in Eq. (3). The results relative to the first fit are indicated by the second column in Fig. 3. In all cases, the nonlinear embedding that employs two atomic properties improves the train and test errors for both energies and forces.

To study the influence of the optimization of radial functions, the parametrizations were repeated including the optimization of the radial functions, i.e., the radial functions were represented as linear combinations of Chebyshev polynomials. The relative errors for a single atomic property or two atomic properties are shown as the third and fourth column in Fig. 3.

The parametrization with the nonlinear embedding function $\mathcal{F}$ with $P = 2$ and the optimization of radial functions leads to the lowest relative RMSE across all models. The energy and force RMSE are reduced by 60% to 90% relative to the reference linear models, demonstrating the efficiency of the double convergence of ACE and motivating a further systematic exploration of nonlinear functions $\mathcal{F}$.

D. Hierarchical basis extension

Often the parametrization of ACE is made more robust by adding parameters iteratively in small batches. To this end, the ACE basis functions are ordered hierarchically according to their expected relative contributions. This strategy not only helps the optimization numerically, but it enables to systematically converge ACE with respect to a single parameter, the number of basis functions, and to terminate the parametrization run when the desired accuracy has been reached.

The hierarchical ordering of the ACE basis functions is built according to their body order ν and the orders $\boldsymbol{n}_t$ and $\boldsymbol{l}_t$ of the involved radial and angular functions, respectively.

The ordering of the basis functions is then given as $\nu + \sum_t \boldsymbol{n}_t + \sum_t \boldsymbol{l}_t$. Two slightly different fitting schemes are derived from this ordering and implemented in `pacemaker`. In the so-called power-order scheme, `pacemaker` adds batches of basis functions in sequential order, but within the limits of maximum angular and radial basis functions specified for each body order by the user. In the so-called body-order scheme, `pacemaker` follows the same sequence, but first adds all basis functions for a given body order before it proceeds to the next higher body order. The expansion coefficients of all previously added functions are still allowed to adjust when the new functions are added. Many other variations of hierarchical ordering schemes are possible and can be implemented easily into `pacemaker`.

Figure 4 shows the performance of the two hierarchical body-order and power-order schemes in comparison with a single fit (indicated by the star symbol), in which all model parameters were optimized at once. A value of $\kappa = 0.5$ was used in the loss function, which for the Cu-III dataset meant that the force error dominates over the energy error. Both hierarchical schemes show similar convergence and clearly outperform the single fit result in terms of both energy and force RMSE values. Information on computational efficiency of ACE potentials for different number of basis functions is provided in Appendix H.

E. Regularization

Two regularization mechanisms, which restrict the absolute values of the expansion coefficients and ensure smoothness of the radial functions, were incorporated via the Δ_{coeff} and Δ_{rad} terms, respectively, into the loss function in Eq. (15).

Figure 5 shows how different regularization settings affect the smoothness of energy vs the nearest-neighbor distance curves for different Cu and C parametrizations. The graphs include models trained on the different nested subsets without any regularization, with a weak L1-regularisation that penalizes the sum of the absolute values of the expansion coefficients, and with a w_0 regularization limiting the absolute magnitudes of the radial functions. The three different regularization settings were applied to the three different datasets for copper and carbon and the results for the different datasets are displayed in the different panels.

The Cu-I and C-I datasets include only data that is close in energy to the ground state. Such datasets tend to be used for the parametrization of ML potentials aimed only at properties of ground state structures, where structures with higher energy are not considered. It is evident that without regularization the description of structures with higher energies is poor. Regularization helps to make the expansion smooth, but a significant discrepancy to the DFT reference remains. Adding more data,

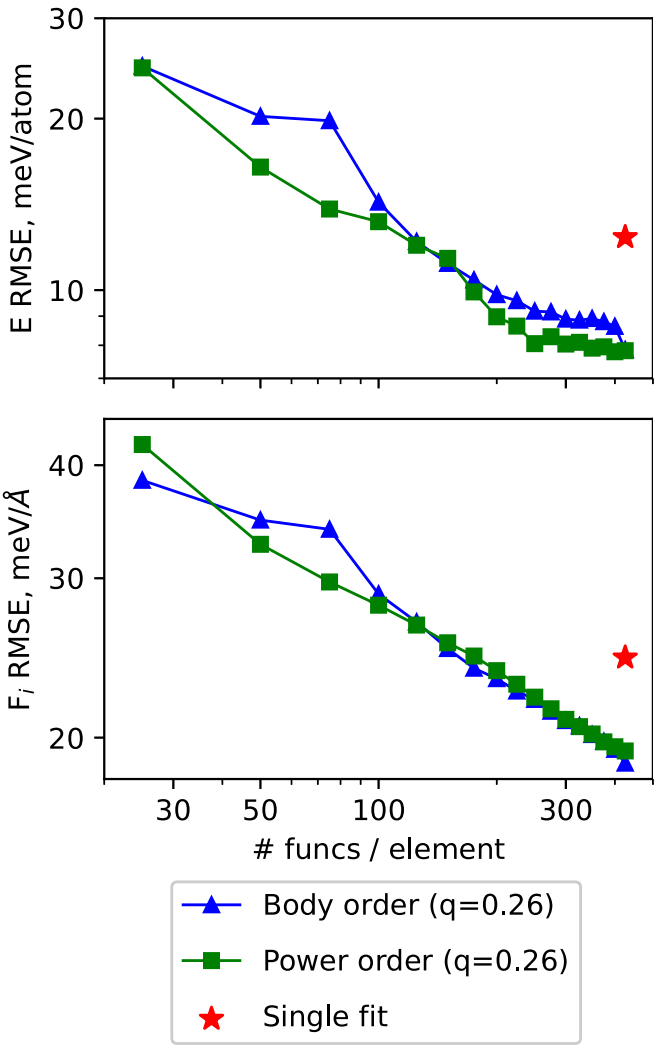

FIG. 4. Energy and force RMSE for Cu-III test set trained using hierarchical body-order and power-order schemes in comparison to a single fit with the same basis functions. The values of q were obtained from the force learning curve.

corresponding to the Cu-II and C-II datasets, helps to remove unphysical predictions and regularization helps to dampen the remaining artifacts. Finally, for the Cu-III and C-III datasets that comprise data across a wide range of energies, the spurious predictions disappear completely and the regularization has hardly any effect.

Note that some models show completely unphysical behavior at short interatomic distances where there is a lack of reference DFT data. As no regularization is able to cure this problem in a reliable way, we implemented in `pacemaker` a strictly positive core repulsion term, as described in Appendix C, that can be supplemented to any ACE model and removes unphysical attraction at short distance. A detailed discussion on the role of regularization and dealing with limited reference data can also be found in Ref. [71].

F. Validation and uncertainty assessment

Validation is critical for assessing the quality of a parametrization in terms of accuracy, transferability and numerical efficiency. `pacemaker` provides several options for quick validation and assessment of ACE parametrizations:

(1) On-the-fly validation. Errors in energies and forces for the train and test datasets are evaluated regularly during training. This enables to follow the progress of the parametrization continuously. It is also possible to monitor desired structures or properties. As an example, the variation of different stacking fault energies in Cu during training is shown in Fig. 6.

(2) Basic statistical analysis. Upon completion of training, `pacemaker` generates a series of graphs for assessing the accuracy of the fitted model with respect to the reference data. These graphs include error distributions resolved with respect to energy, force and shortest bond length that enable an easy detection of outliers. An example for the error distribution in the Cu-III test dataset is shown in Fig. 7.

(3) Uncertainty assessment. By constructing an ensemble of ACE parametrizations from different subsets of the reference data or from different parameter initializations, the uncertainty of the parametrizations and their sensitivity towards predicting particular properties may be estimated [46].

The analysis tools of `pacemaker` provide an overview of the resulting ACE parametrizations. A more detailed validation may directly follow training by making use of a broad range of packages compatible with the `LAMMPS` suite. For instance, the recently developed CALPHY software package [48] is directly compatible with PACE/LAMMPS and enables the automated computation of phase diagrams from the parametrizations at hand.

V. ACE PARAMETRIZATIONS FOR PUBLICLY AVAILABLE DATASETS

In this section, we demonstrate on several publicly available datasets that the training strategy implemented in `pacemaker` achieves parametrizations that rival the best published ML potentials. To show that the general approach implemented in `pacemaker` is suitable for constructing accurate models for different bonding types, we chose three multicomponent systems including ethanol, a set of binary metallic alloys, and a high entropy alloy containing five elements.

A. Ethanol

The ACE for ethanol was parameterized using the revMD-17 dataset [62]. The corresponding learning curve for MAE of the force components as a function of the number of ACE basis functions is shown in Fig. 8. As the reference data was obtained from MD snapshots, the data points are correlated. Therefore, the training set comprised only 10^3 randomly selected samples, while the test set contained 99×10^3 samples. The learning curve was obtained by adding training parameters hierarchically using the body order scheme. The slopes of the learning curves for the training ($q = 0.75$) and test ($q = 0.50$) datasets are different, which may indicate that the small training set is not as diverse as the much larger test set. Two results of single fit runs, where all training parameters were optimized simultaneously, are also marked in Fig. 8 by red symbols. The errors of the single fits are close to the hierarchical basis extension, which implies that the training set is

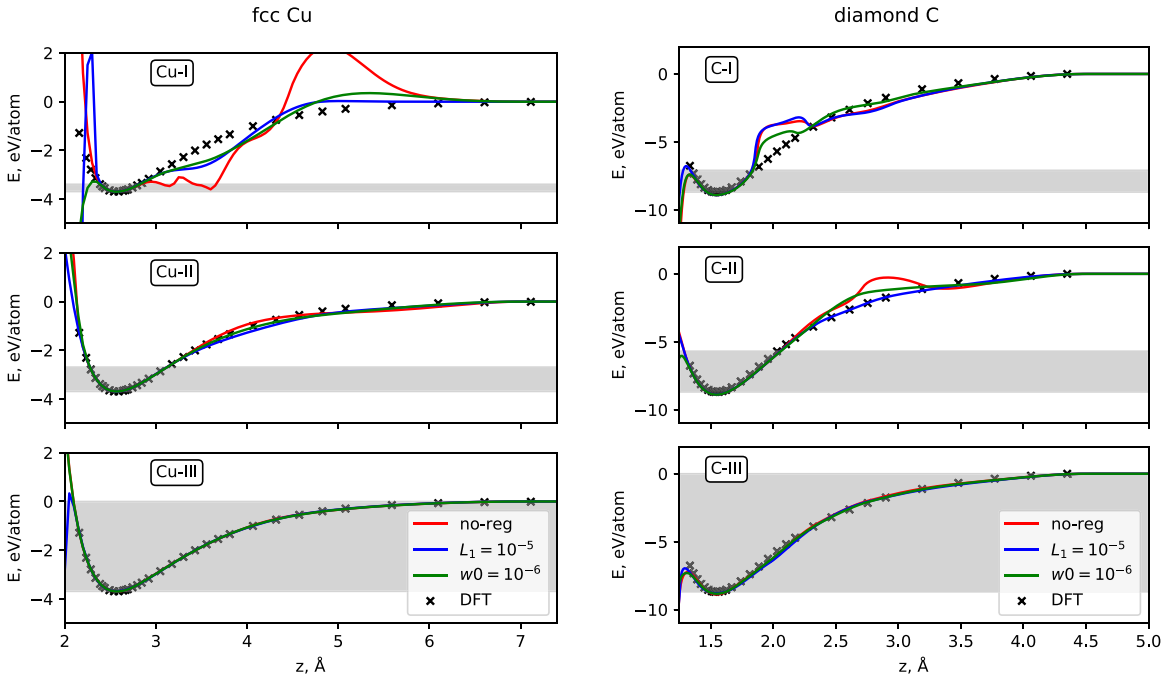

FIG. 5. Energy vs nearest-neighbor distance curves for nonregularized, L1-regularized, and w_0-regularized potentials for fcc Cu (left panels) and diamond C (right panels) structures, trained on the three nested datasets of different sizes. The grey shaded regions in the graphs mark the energy ranges covered by each training dataset.

sufficiently simple so that the nonlinear optimization problem with about 1400 parameters converges when all parameters are initialized from zero.

Our ACE parametrization, employing only ∼1500 functions per element, achieves a test MAE of 0.16 meV/atom for the energy and 8.7 meV/Å per force component. These values are very close to 0.13 meV/atom and 7.3 meV/Å reported by Kovacs *et al.* [72], and 0.1 meV/atom and 6.2 meV/Å obtained by Christensen *et al.* [64], which are the lowest MEA values reported for the revMD-17 dataset.

The ACE learning curve in Fig. 8 indicates that the accuracy of our model can be further improved by adding more basis functions. However, here we do not aim at reaching the smallest possible MAE, but to demonstrate that improvements can be achieved systematically.

B. Binary alloys

Nyshadham *et al.* [5] recently fitted several state-of-the-art surrogate ML models to ten binary alloy datasets and compared the predicted enthalpies of formation. In Fig. 9, we present how ACE parametrizations developed in this work (cf. Table II) perform in comparison with these models.

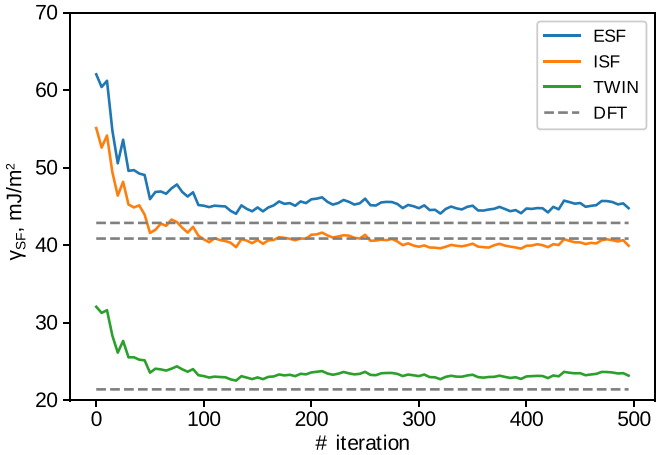

FIG. 6. On-the-fly validation: variation of stacking fault energies in Cu during training [40]. ESF, ISF, and TWIN denote extrinsic, intrinsic, and twin stacking faults, respectively. Dashed lines mark the reference DFT values; the order of the ACE predicted energies matches that of DFT.

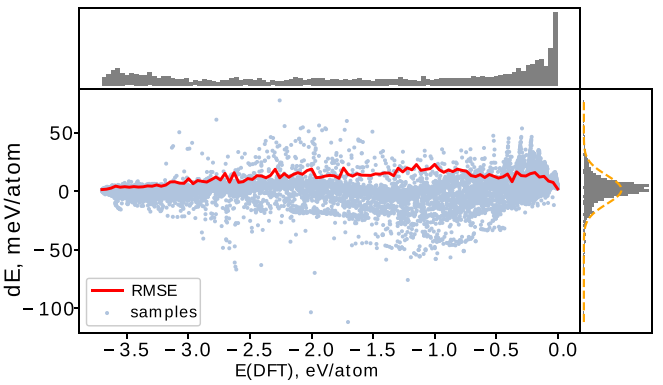

FIG. 7. Error in the energy per atom as a function of the reference energy for the Cu-III test dataset. The red line shows the average error, the top panel the number of samples, and the right panel the overall error distribution, including a normal distribution with identical second moment.

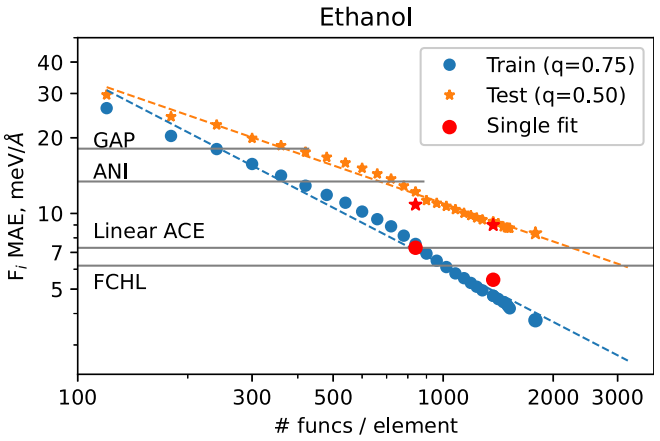

FIG. 8. Convergence of force components for ethanol train and test datasets. Horizontal lines indicate test results of ML potentials given in Ref. [72]. See text for details.

The upper panel of Fig. 9 shows the convergence of the enthalpy of formation when the ten binaries were fitted simultaneously. The ACE training error decreases monotonously with increasing number of basis functions, reaching a value of less than 2 meV/atom, which lies below the error achieved by the best performing model (MTP). The ACE test error converges more slowly, possibly due to the relatively small

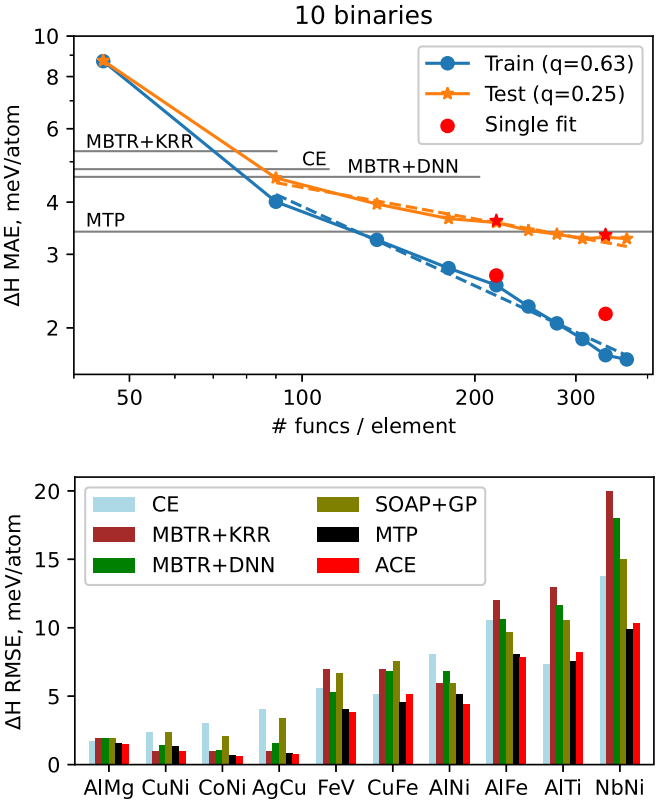

FIG. 9. MAE for the enthalpy of formation versus the number of basis functions per element (only binary interactions were included in ACE). Horizontal lines indicate results of other ML methods investigated in Ref. [5] (upper pannel) and RMSE for the enthalpy of formation for individual binary alloys (lower panel).

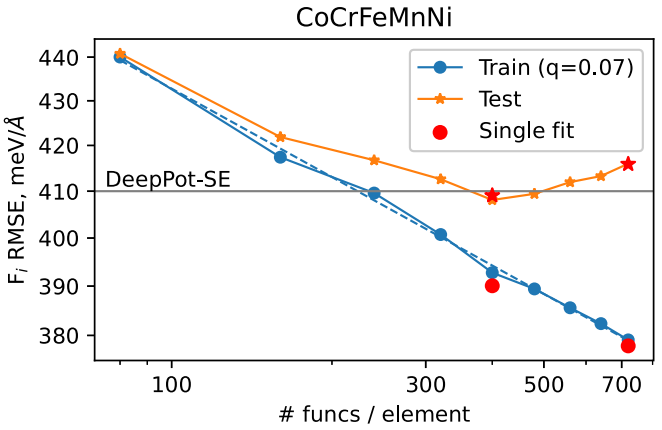

FIG. 10. ACE learning curve for the HEA dataset taken from Ref. [63].

number of reference data. The two single-fit errors illustrate that for complex models the hierarchical addition of training parameters is beneficial.

The lower panel of Fig. 9 shows a comparison for individual binaries. Also here, ACE and MTP consistently give similar results and outperform all other methods. As MTP can be cast in the form of ACE [38], it is not surprising that the predictions of both methods nearly coincide for most alloys. Small deviations between the two methods may be attributed to details of the ACE basis set and `pacemaker` fitting strategy. The hierarchical extension of the basis functions in ACE, Sec. IV D, enables `pacemaker` to achieve these results without user intervention.

C. High entropy alloy

The HEA dataset [63] is relatively small (cf. Table I) and appears to be rather inconsistent, possibly due to the numerical noise or varying magnetic states. This is indicated by the relatively large RMSE values and the low value of $q = 0.07$ for the training curve shown in Fig. 10. It is possible that either the model does not have the required capacity to learn the data, for example, when the reference data comprises different magnetic states while the model is nonmagnetic, or the data is inconsistent and/or insufficient. The low q value reflects the fact that the improvement is marginal when the number of basis functions is increased. Furthermore, a clear overfitting is observed when the model exceeds about 400 functions/element. Nevertheless, the ACE parametrization is able to reach a better accuracy than the reference DeepPot-SE model [63]. However, both models have likely exhausted the capacity of the reference data, as their test errors are similarly large.

For this dataset, the single fits are of comparable accuracy as the hierarchical fit. Therefore a larger and more diverse dataset is necessary to fully benefit from the hierarchical fitting strategy.

VI. SUMMARY AND CONCLUSIONS

We present a general strategy for obtaining accurate parametrizations of the atomic cluster expansion for elements,

alloys and molecules. We provide the open source software package `pacemaker` that implements our strategy and comprises advanced optimization features targeted at GPU hardware. Based on general considerations, we limit ACE to SUB-TRIP invariant representations and suggest that the SUB-TRIP invariance should be fulfilled by any physically meaningful, machine-learned representation of the energy. We highlight the importance of the double convergence of ACE and demonstrate that simple nonlinear embedding functions can significantly improve the accuracy. By ordering the ACE basis functions hierarchically, parametrizations are converged systematically with respect to a single parameter, namely, the number of basis functions.

We demonstrate the application of `pacemaker` on available databases and show that the results are *en par* with the best ML potentials in the reference publications. Given the efficiency of ACE [40], we expect that ACE models parameterized with `pacemaker` compete with or surpass the best ML potentials with respect to both accuracy and execution time.

The `pacemaker` code and the `tensorpotential` fitting backend are available on GitHub [73,74].

ACKNOWLEDGMENTS

R.D. acknowledges funding through the German Science Foundation (DFG), Projects No. 405621217 and No. 403582885. Y.L. acknowledges funding through the German Science Foundation (DFG), Project No. 405602047. M.M. acknowledges funding through the German Science Foundation (DFG), Project No. 405621081. R.D. acknowledges computational resources of the research center ZGH, Ruhr-University Bochum, Germany.

APPENDIX A: `pacemaker` WORKFLOW

The `pacemaker` workflow is described in the following and summarized in Fig. 11:

(i) `pacemaker` starts by constructing the potential according to the user specified basis configuration (ν-order, $n_{\max}$, $l_{\max}$, etc.) or loads it from an available potential file. Then the B-basis functions are constructed, to this end generalized Clebsch-Gordan coefficients are set up for generating product basis functions that are invariant with respect to rotation and inversion, see Appendices B and D.

(ii) Then `pacemaker` constructs the neighborlist for all structures in the dataframe. The neighborlist can be added to the reference dataframe for a fast restart of future parametrization runs.

(iii) Next the weights for each structure and atom as required by the loss function are set up. `pacemaker` provides different weighting schemes, Appendix F. The weights are then added to the reference dataframe. Weights may also be added directly to the reference dataframe, so that the user has full control over the weights for each structure and force.

(iv) `pacemaker` splits the dataset for training and for testing.

(v) The further specification of L_1, L_2 and radial smoothness w_0, w_1, w_2 regularization contributions and the relative weight κ of energy and force errors enables `pacemaker` to set up the loss function, Appendix E.

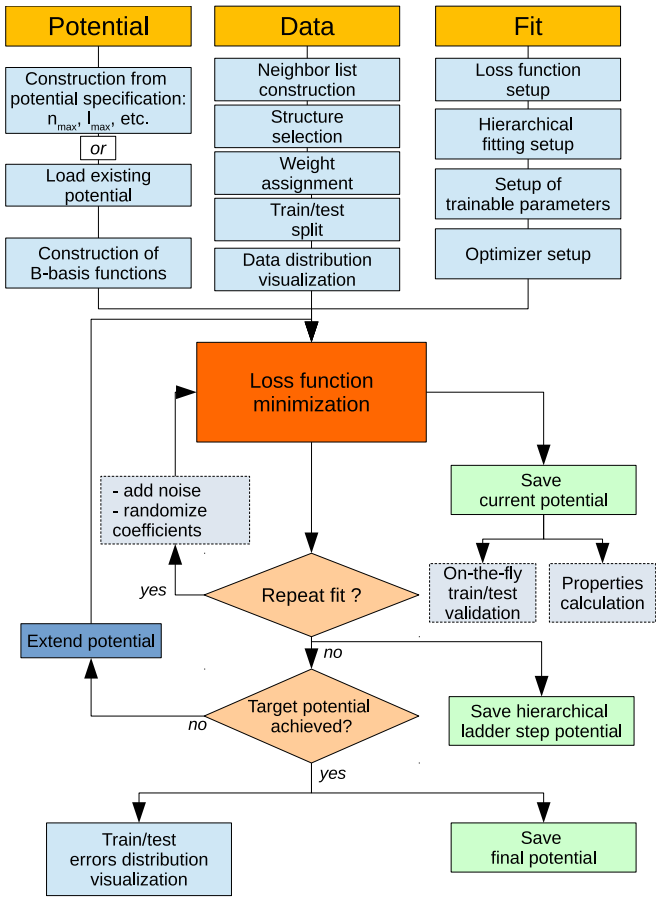

FIG. 11. `pacemaker` workflow scheme. See text.

(vi) The hierarchical basis extension is setup as ladder fitting scheme if requested by the user.

(vii) The optimization of the loss function can be carried out with different optimizers and optimization strategies. For each optimization step `pacemaker` stores the current potential and computes error metrics for energies and forces. In addition, external Python code can be called to perform specific calculations for advanced on-the-fly validation.

(viii) If requested, optimization is repeated with intermediate randomization of the training parameter.

(xi) During and at the end of loss function optimization `pacemaker` provides outputs for assessing the quality and convergence of a parametrization.

(x) `pacemaker` stores (and loads) the ACE potentials in a transparent YAML [75] file format.

APPENDIX B: ATOMIC CLUSTER EXPANSION AS IMPLEMENTED IN `pacemaker`

ACE has been introduced in detail in recent publications [26,38–40]. Here we provide an overview of all terms relevant for the parametrization of ACE with `pacemaker`.

1. Atomic base

The one-particle basis function $\phi_{\mu_i\mu_j nlm}$ are defined as a product of radial function R_{nl} and spherical harmonics Y_{lm},

$$\phi_{\mu_i\mu_j nlm} = R_{nl}^{\mu_i\mu_j}(r_{ji})Y_{lm}(\hat{\boldsymbol{r}}_{ji}), \tag{B1}$$

where the chemical species are indexed as μ_i, μ_j. Radial functions depend on the pairwise distance r_{ij} between atoms i and j. They are zero outside the cutoff distance $r_{\text{cut}}^{\mu_i\mu_j}$ and for shorter distances than the inner cutoff $r_{\text{in}}^{\mu_i\mu_j} - \delta_{\text{in}}^{\mu_i\mu_j}$.

The sum of one-particle basis functions over the neighbours j of the central atom i gives the atomic base

$$A_{i\mu nlm} = \sum_j \delta_{\mu\mu_j}\phi_{\mu_i\mu_j nlm}(\boldsymbol{r}_{ji}). \tag{B2}$$

2. ACE basis functions

The permutation-invariant many-body basis functions of order ν are defined as a product of atomic bases with different indices

$$\boldsymbol{A}_{i\mu\mathbf{nlm}} = \prod_{t=1}^{\nu} A_{i\mu_t n_t l_t m_t}, \tag{B3}$$

where $\mu = (\mu_1, \mu_2, \ldots \mu_\nu)$, $\mathbf{n} = (n_1, n_2, \ldots n_\nu)$, $\mathbf{l} = (l_1, l_2, \ldots l_\nu)$ and $\mathbf{m} = (m_1, m_2, \ldots m_\nu)$ are tuples of length ν of corresponding indices.

The products $\boldsymbol{A}$ are in general not invariant under rotation and inversion. Basis functions that are invariant under rotation and inversion are obtained by multiplication with generalized Glebsch-Gordan coefficients

$$\boldsymbol{B}_{i\mu\mathbf{nlL}} = \sum_{\mathbf{m}} \left(\begin{matrix} lm \\ LM \end{matrix} L_R = 0\right) \boldsymbol{A}_{i\mu\mathbf{nlm}}. \tag{B4}$$

The symbol $(\begin{matrix} lm \\ LM \end{matrix}^{L_R=0})$ indicates generalized Clebsch-Gordan coefficients (GCG), $\mathbf{L} = (L_1, L_2, \ldots, L_{\nu-2})$ is a tuple of orbital moments that characterize the coupling of the GCG and the sum is taken over possible combination $\mathbf{m}$ that gives nonzero contributions. For an interatomic potential we expect invariance under rotation so that the resulting moment of the expansion is zero, $L_R = 0$.

3. ACE expansion of an atomic property

An atomic property that is rotationally invariant, like, for example, the energy of an atom, is expanded as a linear combination of rotationally invariant basis functions

$$\varphi_i^{(p)} = \sum_{\mu\mathbf{nlL}} c_{\mu\mathbf{nlL}}^{(p)} \boldsymbol{B}_{i\mu\mathbf{nlL}} \tag{B5}$$

where p denotes the index of atomic property if several atomic properties are required for an atom and $c_{\mu\mathbf{nlL}}^{(p)}$ are independent expansion coefficients.

Alternatively, using Eq. (B4), this representation can be transformed to

$$\varphi_i^{(p)} = \sum_{\mu\mathbf{nlm}} \tilde{c}_{\mu\mathbf{nlm}}^{(p)} \boldsymbol{A}_{i\mu\mathbf{nlm}}, \tag{B6}$$

where rotational invariance is ensured by the expansion coefficients $\tilde{c}_{\mu\mathbf{nlm}}^{(p)} = \sum_{\mathbf{L}} (\begin{matrix} lm \\ LM \end{matrix}^{L_R=0)} c_{\mu\mathbf{nlL}}^{(p)}$. As can be seen from Eq. (B4), the same product of atomic bases $\boldsymbol{A}_{i\mu\mathbf{nlm}}$ can appear in $\boldsymbol{B}_{i\mu\mathbf{nlL}}$ with different $\mathbf{L}$. The representation Eq. (B6) groups these together, thus providing a computationally more efficient representation than Eq. (B5) [40].

4. Energy and forces

The atomic energy is represented as a general nonlinear function $\mathcal{F}$ of atomic properties [Eq. (4)]

$$E_i = \mathcal{F}\big(\varphi_i^{(1)}, \ldots, \varphi_i^{(P)}\big).$$

`pacemaker` provides different choices for $\mathcal{F}$. For one atomic property a linear relation is assumed,

$$E_i = \varphi_i^{(1)}, \tag{B7}$$

for two properties a Finnis-Sinclair-type expansion is taken

$$E_i = \varphi_i^{(1)} + \sqrt{\varphi_i^{(2)}}, \tag{B8}$$

where the square-root embedding function is slightly modified close to zero argument, see supplementary information for [40]. Other nonlinear functions for more densities can be easily implemented.

The total energy of the atomic system is then obtained as

$$E = \sum_i E_i, \tag{B9}$$

and the forces acting on each atom are given by

$$\boldsymbol{F}_i = -\nabla_i E. \tag{B10}$$

The forces are evaluated analytically using expressions presented in Refs. [26,38–40], and implemented in C++ in `pacemaker`. In `tensorpotential` analytical forces are computed via automatic differentiation.

The expressions for forces and energies are slightly modified when atoms come closer than the inner cutoff, see Appendix C.

APPENDIX C: RADIAL AND CUTOFF FUNCTIONS

For any practical implementation, specific choices for radial functions need to be made. Here we summarize the radial and cutoff functions implemented in `pacemaker`.

1. Radial basis functions

`pacemaker` represents radial functions as a linear combination of radial basis functions

$$R_{nl}^{\mu_i\mu_j}(r) = \sum_{k=1}^{k_{\max}^{\mu_i\mu_j}} c_{nlk}^{\mu_i\mu_j} g_k(r), \tag{C1}$$

with the radial basis functions $g_k(r)$. The radial expansion coefficients $c_{nlk}^{\mu_i\mu_j}$ are initialized as $c_{nlk}^{\mu_i\mu_j} = \delta_{nk}$ and can be kept fixed or optimized. The number of different radial basis functions $k_{\max}^{\mu_i\mu_j}$ is another parameter that needs to be set.

The radial basis functions are smoothly taken to zero outside the cutoff distance r_{cut} and for smaller distances than the

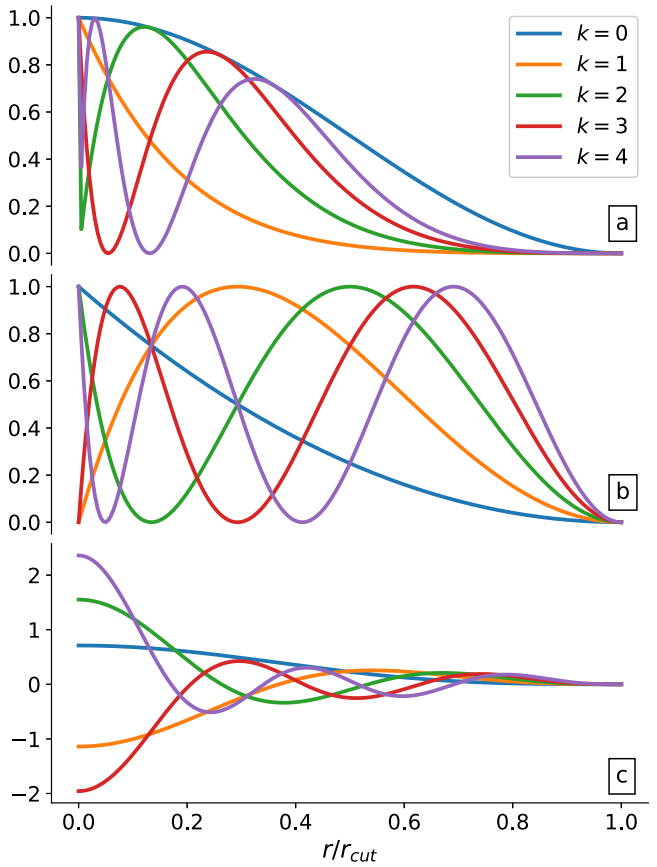

FIG. 12. Radial basis functions $g_k(r)$ available in `pacemaker`. (a) Exponentially-scaled Chebyshev polynomials with $\lambda = 5.25$, (b) Power-law scaled Chebyshev polynomials with $\lambda = 2.0$, (c) Simplified spherical Bessel functions. For all radial basis functions cutoff distance $r_{\text{cut}} = 5$.

inner cutoff r_{in} by multiplicative cutoff functions

$$g_k(r) \to g_k(r) f_{\text{cut}}(r, r_{\text{cut}}, \Delta_{\text{cut}})[1 - f_{\text{cut}}(r, r_{\text{in}}, \Delta_{\text{in}})], \quad \text{(C2)}$$

where f_{cut} is discussed in Appendix C 2.

`pacemaker` offers several different types of radial basis functions. The first few $g_k(r)$ of each type are plotted in Fig. 12 and their detailed description is given below. The selection of a particular type can be viewed as a model hyperparameter. Similar to Zeni *et al.* [76], we find that for small $k \lesssim 10$ and small datasets the spherical Bessel radial basis functions tend to perform best. In addition, Bessel radial basis functions are convenient as they do not require setting the intrinsic parameter λ.

a. Power-law scaled Chebyshev polynomials

The radial basis functions are given by shifted and scaled Chebyshev polynomials of the first kind T_k

$$g_k(r) = (1 - T_k(x))/2, \quad \text{(C3)}$$

with the scaling length

$$x = 1 - 2(1 - r/r_{\text{cut}})^{\lambda}, \quad \text{(C4)}$$

and the parameter λ. The radial basis functions g_k and their first derivative vanish at the cutoff r_{cut} for $\lambda > 1$.

b. Exponentially-scaled Chebyshev polynomials

The radial basis functions are given by shifted and scaled Chebyshev polynomials of the first kind T_k with a cosine envelope

$$g_k(r) = \tfrac{1}{4}(1 - T_k(x))(1 + \cos(\pi r/r_{\text{cut}})), \quad \text{(C5)}$$

with the scaling length

$$x = 1 - 2\left(\frac{e^{-\lambda(r/r_{\text{cut}}-1)} - 1}{e^{\lambda} - 1}\right) \quad \text{(C6)}$$

and the parameter λ. The radial basis functions g_k and their first derivative vanish at the cutoff.

c. Simplified spherical Bessel functions

The spherical Bessel radial basis functions were introduced by Kocer *et al.* [77] and can be expressed by the following recurrence relation,

$$g_k(r) = \frac{1}{\sqrt{d_k}}\left[f_k(r) + \sqrt{\frac{e_k}{d_{k-1}}} g_{k-1}(r)\right], \quad \text{(C7)}$$

with

$$f_k(r) = (-1)^k \frac{\sqrt{2}\pi}{r_c^{3/2}} \frac{(k+1)(k+2)}{\sqrt{(k+1)^2 + (k+2)^2}} \times \left\{\text{sinc}\left[r\frac{(k+1)\pi}{r_c}\right] + \text{sinc}\left[r\frac{(k+2)\pi}{r_c}\right]\right\},$$

$$e_k = \frac{k^2(k+2)^2}{4(k+1)^4 + 1},$$

$$d_k = 1 - \frac{e_k}{d_{k-1}}.$$

The recurrence is initialized with $d_0 = 1$ and $g_0(r) = f_0(r)$.

2. Cutoff function

The cutoff function vary smoothly from 1 to 0 over the interval $[r_c - \Delta, r_c]$,

$$f_{\text{cut}}(r, r_c, \Delta) = \begin{cases} 1 & \text{for } r \leqslant r_c - \Delta, \\ \frac{1}{2}(1 + y) & \text{for } r_c - \Delta < r < r_c, \\ 0 & \text{for } r \geqslant r_c. \end{cases} \quad \text{(C8)}$$

`pacemaker` implements a polynomial cutoff function

$$y = ax + bx^3 + cx^5, \quad \text{(C9)}$$

with

$$x = 1 - 2\left(1 + \frac{r - r_c}{\Delta}\right) \quad \text{(C10)}$$

and $a = 1.875$, $b = -1.25$, $c = 0.375$. The function y varies between $y(1) = 1$ and $y(-1) = -1$ and ensures vanishing first and second derivatives at the boundaries.

3. Core repulsion

At very short bond length the interatomic energy is strictly repulsive. Often no or little data is provided at short interatomic distances as the correspondingly high energies are not of interest for many simulations. For parametrizations for which short interatomic distances are not of interest we provide a core repulsion that smoothly replaces ACE with a strictly repulsive function. This is done to ensure that no unphysical energies are predicted in case short interatomic distances occur during a simulation.

Cutting out ACE and replacing it with a hard core repulsion is done in two steps. A pairwise, strictly positive hard core repulsion is evaluated as

$$\rho_{\text{core}} = \sum_j \phi_{\text{core}}(r_{ij}), \tag{C11}$$

with

$$\phi_{\text{core}}(r) = w_0 \frac{\exp(-w_1 r^2)}{r} f_{\text{cut}}(r, r_{\text{in}}, \Delta_{\text{in}}) \tag{C12}$$

with parameters w_0 and w_1 and $\phi_{\text{core}}(r \geqslant r_{\text{in}}) = 0$. Note that from Eq. (C2) $g_k(r < r_{\text{in}} - \Delta_{\text{in}}) = 0$, so that short-range contributions to ACE are set to zero.

Next, the energy predicted by ACE is smoothly replaced by the hard core repulsion as

$$E_i = \mathcal{F}\big(\varphi_i^{(1)}, \ldots, \varphi_i^{(P)}\big) f_{\text{cut}}(\rho_{\text{core}}, \rho_{\text{c}}, \Delta_{\text{c}}) + \rho_{\text{core}}, \tag{C13}$$

where the cutoff function arguments are not distances but the values of the core repulsion. Typically one chooses $\Delta_{\text{c}} = \rho_{\text{c}}$, so that the cutoff kicks in once ρ_{core} is different from zero and ρ_c is evaluated from the energy of a dimer at distance $r_{\text{in}} - \Delta_{\text{in}}$.

APPENDIX D: GENERALIZED CLEBSCH-GORDAN COEFFICIENTS

The generalized Clebsch-Gordan coefficients are computed using an angular momenta coupling tree that depends on the order ν of the coupled angular momenta $l_1, l_2, \ldots, l_\nu$, abbreviated here as LM-tree [38,39,78]. For example, for a basis function $\boldsymbol{B}$ of order $\nu = 4$, the angular momenta contributions $\boldsymbol{l} = (l_1, l_2, l_3, l_4)$ are coupled to intermediate/internal angular momenta $\boldsymbol{L} = (L_1, L_2)$. The possible values of $\boldsymbol{L}$ are constrained by the angular momentum selection rules and the projections $\boldsymbol{m}$ just couple as $M_1 = m_1 + m_2$ and $M_2 = m_3 + m_4$. The intermediate $\boldsymbol{L}$ are then coupled to the resulting angular momentum L_R. For an interatomic potential that is rotationally invariant only $L_R = 0$ and $M_R = 0$ is possible, which also implies $\sum m_i = 0$. Figure 13 shows the corresponding LM-tree with l-inputs and intermediate L nodes. Coupling is facilitated by multiplication with a Clebsch-Gordan coefficient at each node of the LM-tree, for $\nu = 4$ the generalized Clebsch-Gordan coefficient (GCG) is obtained as

$$\left(\begin{matrix} lm \\ LM \end{matrix} L_R = 0\right) = \sum_{\boldsymbol{m}} C^{M_R M_1 M_2}_{L_R L_1 L_2} C^{M_1 l_1 l_2}_{L_1 l_1 l_2} C^{M_2 m_3 M_4}_{L_2 l_3 l_4}, \tag{D1}$$

with $M_1 = m_1 + m_2$ and $M_2 = m_3 + m_4$ and $L_R = M_R = 0$.

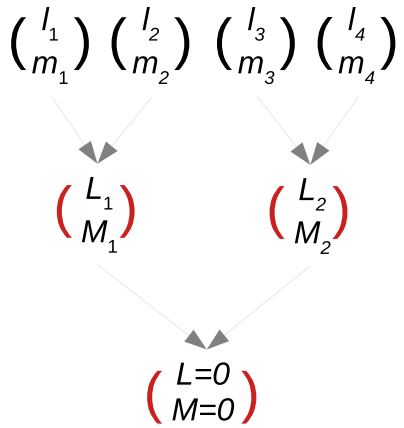

FIG. 13. LM-tree for $\nu = 4$.

APPENDIX E: LOSS FUNCTION

The parametrization of ACE is accomplished by minimizing the loss function. An appropriate design of the loss function is key to the quality of the resulting potential. The following loss function is implemented in `pacemaker`,

$$\mathcal{L} = (1-\kappa) \sum_{n=1}^{N_{\text{struct}}} w_n^{(E)} \left(\frac{E_n^{\text{ACE}} - E_n^{\text{ref}}}{n_{\text{at,n}}} \right)^2 + \kappa \sum_{n=1}^{N_{\text{struct}}} \sum_{i=1}^{n_{\text{at,n}}} w_{ni}^{(F)} \big(\boldsymbol{F}_{ni}^{\text{ACE}} - \boldsymbol{F}_{ni}^{\text{ref}} \big)^2 + \Delta_{\text{coeff}} + \Delta_{\text{rad}}, \tag{E1}$$

where κ is the relative force weight in the loss function, N_{struct} is the number of structures employed in the parametrization, $n_{\text{at,n}}$ the number of atoms in structure n, and $w_n^{(E)}$ and $w_n^{(F)}$ are per-structure and per-atom weights for the energy and force residuals. Δ_{coeff} is the regularization term that penalizes the absolute values of the expansion coefficients $c^{(p)}_{\mu\mathbf{nlL}}$ through the elastic-net regularization [79]

$$\Delta_{\text{coeff}} = L_1 \sum_{p\mu\mathbf{nlL}} \big| c^{(p)}_{\mu\mathbf{nlL}} \big| + L_2 \sum_{p\mu\mathbf{nlL}} \big| c^{(p)}_{\mu\mathbf{nlL}} \big|^2, \tag{E2}$$

where L_1 and L_2 are regularization parameters. Figure 14 shows the impact of L1 and L2 regularization on the energy and force errors for three Cu datasets. The regularization helps to improve the test error for the small Cu-I dataset, where the train and test errors are very different. For the Cu-II and Cu-III dataset, with train and test errors that nearly coincide, the regularization has little impact. This is further illustrated in Fig. 15, where the parametrization from the small Cu-I dataset is used to predict the larger Cu-II and Cu-III datasets. Here regularization helps to lower the test errors, but it cannot make up for the fact that the configurations in the Cu-I dataset are rather restricted and therefore extrapolation to Cu-II and Cu-III is poor.

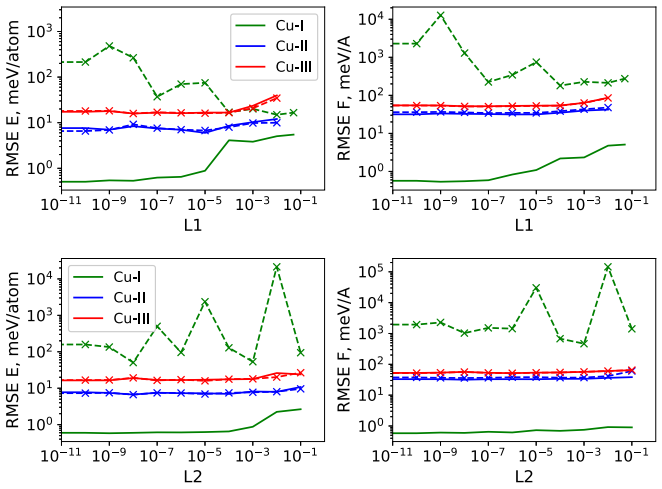

FIG. 14. Dependencies of the energy per atom and force vectors RMSE on the L1 (top) and L2 (bottom) regularization parameters for train (solid lines) and test (dashed lines with crosses) sets of Cu-I, Cu-II, and Cu-III.

The smoothness of the radial functions $R_{nl}(r)$ is controlled through

$$\Delta_{\text{rad}} = \frac{w_0}{r_{\text{cut}}^2}\int r^2 \sum_{nl} |R_{nl}(r)|\,dr + \frac{w_1}{r_{\text{cut}}^2}\int r^2 \sum_{nl}\left|\frac{dR_{nl}(r)}{dr}\right| dr + \frac{w_2}{r_{\text{cut}}^2}\int r^2 \sum_{nl}\left|\frac{d^2R_{nl}(r)}{dr^2}\right| dr, \tag{E3}$$

where w_0, w_1, and w_2 are radial smoothness regularization parameters. Figure 16 illustrates the impact of the radial smoothness regularization on the loss function, respectively the energy and force errors. Δ_{rad} puts restrictions on the shape of the radial functions. It can be seen in Fig. 16 that it hardly affects the accuracy of the model provided that its contribution to the total loss does not exceed a few percent. However, even small values of Δ_{rad} help preventing oscillations in regions where training data is missing, as shown in Fig. 5.

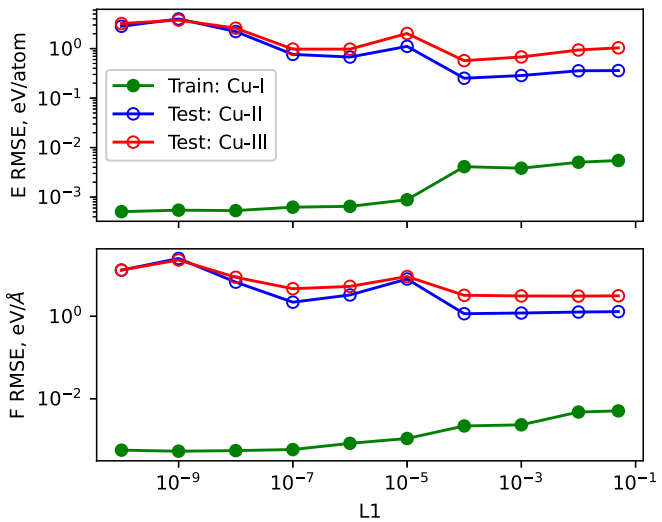

FIG. 15. Extrapolation of ACE potential trained on Cu-I dataset to the prediction of Cu-II and Cu-III datasets.

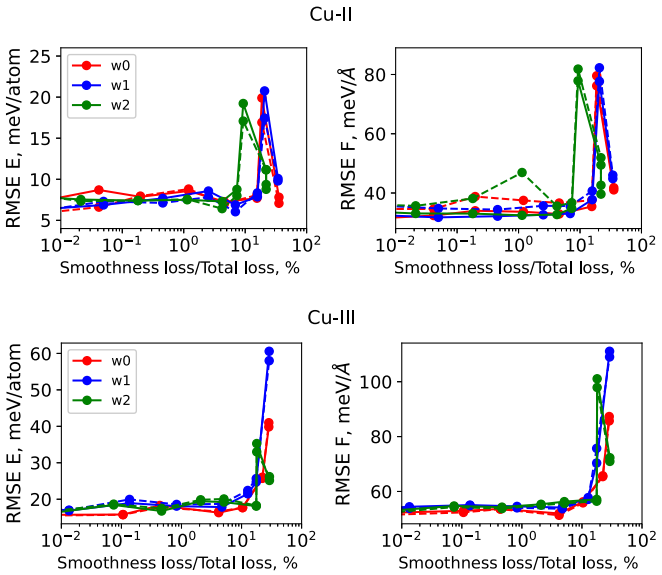

FIG. 16. Dependencies of the energy per atom and force vectors RMSE on the relative contribution of smoothness regularizations w_0, w_1 and w_2 to the total loss function for train (solid lines) and test (dashed lines) subsets of Cu-II and Cu-III.

APPENDIX F: REFERENCE DATA PREPARATION

In general structure selection and weighting can be done by manipulating structures and adding weights directly in the *pandas* reference dataframe. In addition, `pacemaker` implements a few standard schemes to facilitate data preparation and for setting up parametrization runs quickly.

1. Structure selection

`pacemaker` provides useful rules for structure selection:

(i) Select only periodic or only nonperiodic structures or both.

(ii) Select randomly a given number of structures in the dataset below a certain energy difference to the lowest energy/atom structure $\Delta E^{(\text{low})}$.

(iii) Select randomly a given number of structures with energy/atom above $\Delta E^{(\text{low})}$ and below a certain energy difference to the lowest energy/atom structure $\Delta E^{(\text{up})}$.

(iv) Select all structures in the dataset with forces below a certain absolute value $\Delta F^{(\text{low})}$.

Typical values are $\Delta E^{(\text{low})} = 1$ eV, $\Delta E^{(\text{up})} = 10$ eV, and $\Delta F^{(\text{low})} = 50$ eV/Å.

2. Structure weighting

The structure and force weights in the loss function Eq. (E1) can be read from the reference dataframe or assigned by a weighting policy. The energy weights are assigned to each structure in the reference dataframe, whereas force weights are assigned to each atom in each structure. `pacemaker` implements two standard weighting policies: uniform and energy-based. In both policies the weights for energies and forces are normalized to sum to one, respectively.

The uniform weighting policy assigns equal weights to all structures, independent of the number of atoms in a structure. The energy-based weighting scheme assigns weight to a structure depending on its energy per atom and distance to the ground state or to the convex hull. The weight of structure n in the reference dataframe is given by

$$w_n^{(E)} = \omega_E / \left[E_n^{\mathrm{ref}} - E_{\mathrm{min}}^{\mathrm{ref}} + \Delta_E\right]^2, \tag{F1}$$

where Δ_E is a shift to avoid overweighting of the ground state (typically $\Delta_E \approx 1\,\mathrm{eV}$), ω_E is chosen to ensure $\sum_n w_n^{(E)} = 1$ and energies are per atom.

Weights are further assigned to every atom i in each structure n to ensure a suitable match of small forces,

$$w_{ni}^{(F)} = \omega_F w_n^{(E)} / \left[\left(F_{ni}^{\mathrm{ref}}\right)^2 + \Delta_F\right], \tag{F2}$$

where Δ_F is a shift to avoid overweighting of atoms with zero force [typically $\Delta_F \approx 1(\mathrm{eV/\AA})^2$] and ω_F is chosen to ensure $\sum_{ni} w_{ni}^{(F)} = 1$.

Finally, the relative weight of the structures below $\Delta E^{(\mathrm{low})}$ and between $\Delta E^{(\mathrm{low})}$ and $\Delta E^{(\mathrm{up})}$ is fixed to ensure the fit is not changed too much when more structures are added. To this end the weights for the structures below $\Delta E^{(\mathrm{low})}$ are multiplied to fulfill

$$f_{low} = w_{low} \sum_n^{E_n^{\mathrm{ref}} - E_{\mathrm{min}}^{\mathrm{ref}} \leqslant \Delta E^{(\mathrm{low})}} w_n^{(E)}, \tag{F3}$$

$$f_{up} = w_{up} \sum_n^{E_n^{\mathrm{ref}} - E_{\mathrm{min}}^{\mathrm{ref}} > \Delta E^{(\mathrm{low})}} w_n^{(E)}, \tag{F4}$$

with $f_{low} + f_{up} = 1$ and f_{low}, f_{up} input parameters that are both positive. Each structure weight is then multiplied with

$$w_{low/up} w_n^{(E)} \rightarrow w_n^{(E)}, \tag{F5}$$

depending on E_n^{ref}.

APPENDIX G: ACE CONFIGURATION DETAILS

Table II summarizes the hyperparameters for the ACE potentials that were used in this paper. The multi-index $\boldsymbol{v}$ in Eq. (8) is expanded as

$$\varphi_i^{(p)} = \sum_{\nu} \sum_{\mu \mathbf{nlm}} \tilde{c}^{(p)}_{\mu_i \mu \mathbf{nlm}} \prod_{t=1}^{\nu} A_{i \mu_t n_t l_t m_t}, \tag{G1}$$

where μ, $\mathbf{n}$, $\mathbf{l}$, and $\mathbf{m}$ are all tuples of length ν with maximum values determined by the user-defined hyperparameters n_{max}, l_{max} for each order ν.

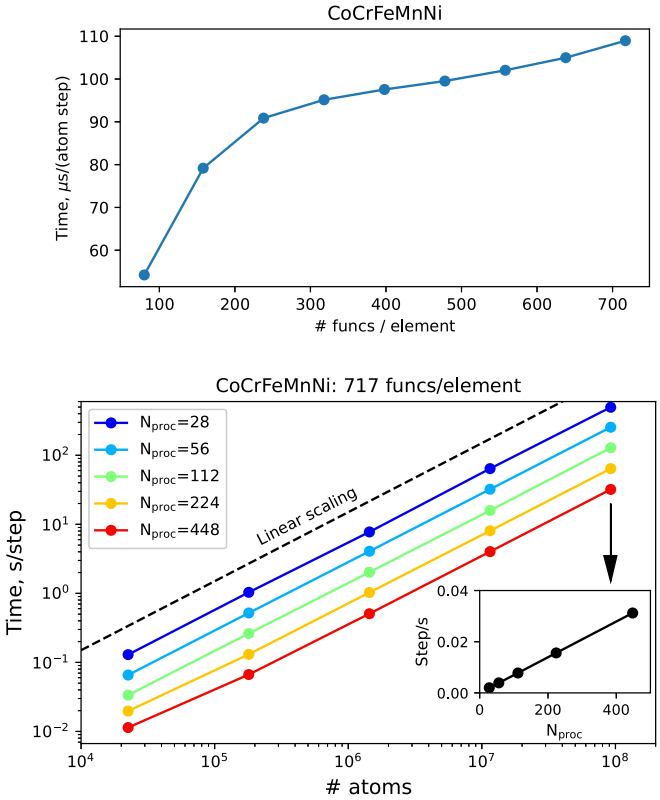

FIG. 17. Dependency of the computational time for the HEA ACE potential with respect to the number of basis functions per element (upper panel) and with respect to the number of atoms and number of cores (lower panel).

APPENDIX H: SCALABILITY AND COMPUTATIONAL COST

Timing tests provided here were done on one or more computational nodes equipped with dual Intel(R) Xeon(R) Gold 6132 processors, using LAMMPS (version 27Oct2021) compiled with the GCC 7.3.0 compiler. Figure 17 (upper panel) shows the scaling of the wall time per atom per MD step for the HEA ACE potential on a single core for different numbers of basis functions. There are two main contributions: setting up the atomic base and computing the product basis functions [40]. Figure 17 (lower panel) demonstrates the linear scaling of the computational time per MD step with respect to the number of atoms. The inset shows the linear scaling with respect to the number of cores. A comparison of the computational time and accuracy to other ML potentials was presented in detail in Ref. [40].

[1] J. Behler, Perspective: Machine learning potentials for atomistic simulations, J. Chem. Phys. **145**, 170901 (2016).

[2] A. P. Bartók, S. De, C. Poelking, N. Bernstein, J. R. Kermode, G. Csányi, and M. Ceriotti, Machine learning unifies the modeling of materials and molecules, Sci. Adv. **3**, e1701816 (2017).

[3] B. Sanchez-Lengeling and A. Aspuru-Guzik, Inverse molecular design using machine learning: Generative

models for matter engineering, Science **361**, 360 (2018).

[4] L. Zhang, J. Han, H. Wang, R. Car, and Weinan E, Deep Potential Molecular Dynamics: A Scalable Model with the Accuracy of Quantum Mechanics, Phys. Rev. Lett. **120**, 143001 (2018).

[5] C. Nyshadham, M. Rupp, B. Bekker, A. V. Shapeev, T. Mueller, C. W. Rosenbrock, G. Csányi, D. W. Wingate, and G. L. Hart, Machine-learned multi-system surrogate models for materials prediction, npj Comput. Mater. **5**, 51 (2019).

[6] F. Musil, A. Grisafi, A. P. Bartók, C. Ortner, G. Csányi, and M. Ceriotti, Physics-inspired structural representations for molecules and materials, Chem. Rev. **121**, 9759 (2021).

[7] O. T. Unke, S. Chmiela, H. E. Sauceda, M. Gastegger, I. Poltavsky, K. T. Schütt, A. Tkatchenko, and K.-R. Müller, Machine learning force fields, Chem. Rev. **121**, 10142 (2021).

[8] O. A. von Lilienfeld and K. Burke, Retrospective on a decade of machine learning for chemical discovery, Nat. Commun. **11**, 4895 (2020).

[9] M. Bogojeski, L. Vogt-Maranto, M. E. Tuckerman, K.-R. Müller, and K. Burke, Quantum chemical accuracy from density functional approximations via machine learning, Nat. Commun. **11**, 5223 (2020).

[10] H. Chan, B. Narayanan, M. J. Cherukara, F. G. Sen, K. Sasikumar, S. K. Gray, M. K. Chan, and S. K. Sankaranarayanan, Machine learning classical interatomic potentials for molecular dynamics from first-principles training data, J. Phys. Chem. C **123**, 6941 (2019).

[11] V. L. Deringer, M. A. Caro, and G. Csányi, Machine learning interatomic potentials as emerging tools for materials science, Adv. Mater. **31**, 1902765 (2019).

[12] G. Carleo, I. Cirac, K. Cranmer, L. Daudet, M. Schuld, N. Tishby, L. Vogt-Maranto, and L. Zdeborová, Machine learning and the physical sciences, Rev. Mod. Phys. **91**, 045002 (2019).

[13] T. Mueller, A. Hernandez, and C. Wang, Machine learning for interatomic potential models, J. Chem. Phys. **152**, 050902 (2020).

[14] Y. Mishin, Machine-learning interatomic potentials for materials science, Acta Mater. **214**, 116980 (2021).

[15] P. Friederich, F. Häse, J. Proppe, and A. Aspuru-Guzik, Machine-learned potentials for next-generation matter simulations, Nat. Mater. **20**, 750 (2021).

[16] G. L. Hart, T. Mueller, C. Toher, and S. Curtarolo, Machine learning for alloys, Nat. Rev. Mater. **6**, 730 (2021).

[17] D. Lemm, G. F. von Rudorff, and O. A. von Lilienfeld, Machine learning based energy-free structure predictions of molecules, transition states, and solids, Nat. Commun. **12**, 4468 (2021).

[18] T. W. Ko, J. A. Finkler, S. Goedecker, and J. Behler, A fourth-generation high-dimensional neural network potential with accurate electrostatics including non-local charge transfer, Nat. Commun. **12**, 398 (2021).

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

[20] A. P. Bartók, M. C. Payne, R. Kondor, and G. Csányi, Gaussian Approximation Potentials: The Accuracy of Quantum Mechanics, without the Electrons, Phys. Rev. Lett. **104**, 136403 (2010).

[21] A. Thompson, L. Swiler, C. Trott, S. Foiles, and G. Tucker, Spectral neighbor analysis method for automated generation of quantum-accurate interatomic potentials, J. Comput. Phys. **285**, 316 (2015).

[22] A. V. Shapeev, Moment tensor potentials: A class of systematically improvable interatomic potentials, Mult. Model. Simul. **14**, 1153 (2016).

[23] N. Artrith and A. Urban, An implementation of artificial neural-network potentials for atomistic materials simulations: Performance for TiO_2, Comput. Mater. Sci. **114**, 135 (2016).

[24] A. Takahashi, A. Seko, and I. Tanaka, Linearized machine-learning interatomic potentials for non-magnetic elemental metals: Limitation of pairwise descriptors and trend of predictive power, J. Chem. Phys. **148**, 234106 (2018).

[25] D. Dragoni, T. D. Daff, G. Csányi, and N. Marzari, Achieving dft accuracy with a machine-learning interatomic potential: Thermomechanics and defects in bcc ferromagnetic iron, Phys. Rev. Materials **2**, 013808 (2018).

[26] R. Drautz, Atomic cluster expansion for accurate and transferable interatomic potentials, Phys. Rev. B **99**, 014104 (2019).

[27] G. P. P. Pun, R. Batra, R. Ramprasad, and Y. Mishin, Physically informed artificial neural networks for atomistic modeling of materials, Nat. Commun. **10**, 2339 (2019).

[28] P. Rowe, V. L. Deringer, P. Gasparotto, G. Csányi, and A. Michaelides, An accurate and transferable machine learning potential for carbon, J. Chem. Phys. **153**, 034702 (2020).

[29] V. L. Deringer, M. A. Caro, and G. Csányi, A general-purpose machine-learning force field for bulk and nanostructured phosphorus, Nat. Commun. **11**, 5461 (2020).

[30] H. Mori and T. Ozaki, Neural network atomic potential to investigate the dislocation dynamics in bcc iron, Phys. Rev. Materials **4**, 040601(R) (2020).

[31] J. S. Smith, B. Nebgen, N. Mathew, J. Chen, N. Lubbers, L. Burakovsky, S. Tretiak, H. A. Nam, T. Germann, S. Fensin *et al.*, Automated discovery of a robust interatomic potential for aluminum, Nat. Commun. **12**, 1 (2021).

[32] M. Stricker, B. Yin, E. Mak, and W. A. Curtin, Machine learning for metallurgy II. A neural-network potential for magnesium, Phys. Rev. Materials **4**, 103602 (2020).

[33] A. C. P. Jain, D. Marchand, A. Glensk, M. Ceriotti, and W. A. Curtin, Machine learning for metallurgy III: A neural network potential for Al-Mg-Si, Phys. Rev. Materials **5**, 053805 (2021).

[34] B. Grabowski, Y. Ikeda, P. Srinivasan, F. Körmann, C. Freysoldt, A. I. Duff, A. Shapeev, and J. Neugebauer, *Ab initio* vibrational free energies including anharmonicity for multicomponent alloys, npj Comput. Mater. **5**, 80 (2019).

[35] B. Cheng, G. Mazzola, C. J. Pickard, and M. Ceriotti, Evidence for supercritical behaviour of high-pressure liquid hydrogen, Nature (London) **585**, 217 (2020).

[36] R. Jinnouchi, F. Karsai, and G. Kresse, On-the-fly machine learning force field generation: Application to melting points, Phys. Rev. B **100**, 014105 (2019).

[37] Y. Xie, J. Vandermause, L. Sun, A. Cepellotti, and B. Kozinsky, Bayesian force fields from active learning for simulation of inter-dimensional transformation of stanene, npj Comput. Mater. **7**, 1 (2021).

[38] R. Drautz, Atomic cluster expansion of scalar, vectorial, and tensorial properties including magnetism and charge transfer, Phys. Rev. B **102**, 024104 (2020).

[39] G. Dusson, M. Bachmayr, G. Csányi, R. Drautz, S. Etter, C. van der Oord, and C. Ortner, Atomic cluster expansion: Completeness, efficiency and stability, arXiv:1911.03550.

[40] Y. Lysogorskiy, C. van der Oord, A. Bochkarev, S. Menon, M. Rinaldi, T. Hammerschmidt, M. Mrovec, A. Thompson,

G. Csányi, C. Ortner *et al.*, Performant implementation of the atomic cluster expansion (PACE): Application to copper and silicon, npj Comput. Mater. **7**, 97 (2021).
[41] A. P. Bartók, R. Kondor, and G. Csányi, On representing chemical environments, Phys. Rev. B **87**, 184115 (2013).
[42] https://www.lammps.org.
[43] E. V. Podryabinkin and A. V. Shapeev, Active learning of linearly parametrized interatomic potentials, Comput. Mater. Sci. **140**, 171 (2017).
[44] N. Bernstein, G. Csányi, and V. L. Deringer, De novo exploration and self-guided learning of potential-energy surfaces, npj Comput. Mater. **5**, 99 (2019).
[45] L. Zhang, D.-Y. Lin, H. Wang, R. Car, and Weinan E, Active learning of uniformly accurate interatomic potentials for materials simulation, Phys. Rev. Materials **3**, 023804 (2019).
[46] A. A. Peterson, R. Christensen, and A. Khorshidi, Addressing uncertainty in atomistic machine learning, Phys. Chem. Chem. Phys. **19**, 10978 (2017).
[47] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard *et al.*, Tensorflow: A system for large-scale machine learning, in *12th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 16)* (USENIX Association, 2016) pp. 265–283.
[48] S. Menon, Y. Lysogorskiy, J. Rogal, and R. Drautz, Automated free-energy calculation from atomistic simulations, Phys. Rev. Materials **5**, 103801 (2021).
[49] https://github.com/ACEsuit/ACE.jl.
[50] M. W. Finnis and J. E. Sinclair, A simple empirical N-body potential for transition metals, Philos. Mag. A **50**, 45 (1984).
[51] J. Friedel, Transition metals. electronic structure of the d-band. its role in the crystalline and magnetic structures, in *Electrons*, edited by J. M. Ziman, Physics of Metals, Vol. 1, (Pergamon, London, 1969).
[52] F. Ducastelle, J. Phys. (Paris) **31**, 1055 (1970).
[53] R. P. Gupta, Lattice relaxation at a metal surface, Phys. Rev. B **23**, 6265 (1981).
[54] S. N. Pozdnyakov, M. J. Willatt, A. P. Bartók, C. Ortner, G. Csányi, and M. Ceriotti, Incompleteness of Atomic Structure Representations, Phys. Rev. Lett. **125**, 166001 (2020).
[55] W. McKinney, Data structures for statistical computing in Python, in *Proceedings of the 9th Python in Science Conference*, edited by S. van der Walt and J. Millman (2010) pp. 56–61, doi: 10.25080/Majora-92bf1922-00a.
[56] J. Reback, jbrockmendel, W. McKinney, J. Van den Bossche, T. Augspurger, P. Cloud, S. Hawkins, M. Roeschke, gfyoung, Sinhrks, A. Klein *et al.*, The pandas development team, pandas-dev/pandas: Pandas, 1.4.0rc0 (v1.4.0rc0), Zenodo, doi: 10.5281/zenodo.5824773.
[57] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin *et al.*, TensorFlow: Large-scale machine learning on heterogeneous systems (2015), software available from tensorflow.org.
[58] V. Blum, R. Gehrke, F. Hanke, P. Havu, V. Havu, X. Ren, K. Reuter, and M. Scheffler, *Ab initio* molecular simulations with numeric atom-centered orbitals, Comput. Phys. Commun. **180**, 2175 (2009).
[59] V. Havu, V. Blum, P. Havu, and M. Scheffler, Efficient O(N) integration for all-electron electronic structure calculation using numerically tabulated basis functions, J. Comput. Phys. **228**, 8367 (2009).
[60] G. Kresse and J. Furthmüller, Efficient iterative schemes for *ab initio* total-energy calculations using a plane-wave basis set, Phys. Rev. B **54**, 11169 (1996).
[61] G. Kresse and D. Joubert, From ultrasoft pseudopotentials to the projector augmented-wave method, Phys. Rev. B **59**, 1758 (1999).
[62] A. Christensen and O. A. von Lilienfeld, Revised MD17 dataset (2020), Materials Cloud Archive 2020.82 doi: 10.24435/materialscloud:wy-kn.
[63] L. Zhang, J. Han, H. Wang, W. A. Saidi, R. Car and W. E, End-to-end symmetry preserving inter-atomic potential energy model for finite and extended systems, arXiv:1805.09003.
[64] A. S. Christensen and O. A. von Lilienfeld, On the role of gradients for machine learning of molecular energies and forces, Mach. Learn.: Sci. Technol. **1**, 045018 (2020).
[65] T. Viering and M. Loog, The shape of learning curves: A review, arXiv:2103.10948.
[66] G. Pólya, Über den zentralen grenzwertsatz der wahrscheinlichkeitsrechnung und das momentenproblem, Math. Z. **8**, 171 (1920).
[67] M. J. Willatt, F. Musil, and M. Ceriotti, Feature optimization for atomistic machine learning yields a data-driven construction of the periodic table of the elements, Phys. Chem. Chem. Phys. **20**, 29661 (2018).
[68] N. Artrith and J. Behler, High-dimensional neural network potentials for metal surfaces: A prototype study for copper, Phys. Rev. B **85**, 045439 (2012).
[69] T. Yokoi, Y. Noda, A. Nakamura, and K. Matsunaga, Neural-network interatomic potential for grain boundary structures and their energetics in silicon, Phys. Rev. Materials **4**, 014605 (2020).
[70] A. Krishnapriyan, P. Yang, A. M. N. Niklasson, and M. J. Cawkwell, Numerical optimization of density functional tight binding models: Application to molecules containing carbon, hydrogen, nitrogen, and oxygen, J. Chem. Theory Comput. **13**, 6191 (2017).
[71] C. van der Oord, G. Dusson, G. Csányi, and C. Ortner, Regularised atomic body-ordered permutation-invariant polynomials for the construction of interatomic potentials, Mach. Learn.: Sci. Technol. **1**, 015004 (2020).
[72] D. P. Kovacs, C. van der Oord, J. Kucera, A. Allen, D. Cole, C. Ortner, and G. Csanyi, Linear atomic cluster expansion force fields for organic molecules: Beyond RMSE, J. Chem. Theo. Comp. **17**, 7696 (2021).
[73] https://github.com/ICAMS/python-ace.
[74] https://github.com/ICAMS/tensorpotential.
[75] https://yaml.org/.
[76] C. Zeni, K. Rossi, A. Glielmo, and S. de Gironcoli, Compact atomic descriptors enable accurate predictions via linear models, J. Chem. Phys. **154**, 224112 (2021).
[77] E. Kocer, J. K. Mason, and H. Erturk, A novel approach to describe chemical environments in high-dimensional neural network potentials, J. Chem. Phys. **150**, 154102 (2019).
[78] A. P. Yutsis, I. B. Levinson, and V. V. Vanagas, *The Theory of Angular Momentum* (Israel Program for Scientific Translations, Jerusalem, 1962).
[79] H. Zou and T. Hastie, Regularization and variable selection via the elastic net, J. R. Stat. Soc.: Ser. B (Stat. Methodol.) **67**, 301 (2005).