Application of machine learning potential to predict grain boundary properties and development of its performant implementation

Takayuki Nishiyama

2022

Contents

Chapter 1

General Introduction

Electronic structural approaches have been changing considerably the strategy of materials science research. Calculations based on electronic theory have been recognized as one of the main strategies to elucidate materials properties. The success of this approach is due to the fact that first-principles calculations have made it possible to predict many of the properties of materials with accuracy comparable to or even superior to experiments. The establishment of the density functional theory (DFT) [1, 2] has greatly contributed to the development of first-principles calculations. In addition, due to the significant improvement in computing power in recent years, it became possible to access relatively complicated problems. By making it possible to predict the properties of materials quantitatively and accurately by calculation, it has become realistic to design materials showing desired properties using first-principles calculations. However, first-principles calculations are facing challenging problems such as exceeding the limits of the time scale, bridging the length scale, and performing thermodynamically accurate calculations.

The description of atomic bonds in a solid by first-principles calculations is limited to very few atoms and very short time scales. Specifically, it can handle only about 1,000 atoms at the maximum and is limited to calculations of several tens of picoseconds. In order to extend this most basic description method to large systems and long-scale computations, approximations must be introduced into quantum mechanical methods, such as tight-binding approximations. However, these approximations considerably limit the accuracy, which is the greatest advantage of quantum mechanical methods. Therefore, the range in which the properties of real materials can be simulated using quantum mechanics is substantially limited.

To avoid the limitations of first-principles calculations, purely classical pictures are used instead. This classical picture describes the interaction between atoms using empirical interatomic potential, which is referred to as atomistic simulation. But this classical picture introduces another severe approximation. In other words, the calculation of empirical interatomic potential essentially limits the ability to accurately capture indefinite

situations such as the formation and dissolution of bonds between atoms during simulation. These ambiguities of bonds limit the scope of physical phenomena that can be well explained by atomistic simulation.

Another practical aspect is that even in atomistic simulations, the capabilities of computers limit the system size and time scale that can be simulated. This is also a major factor that makes it difficult to obtain the required accuracy in materials simulation. However, this problem has been solved to a large extent by improvements in computer power. In recent years, the number of parallel processors and the communication speed between computational nodes have improved more than expected. By spatially decomposing on a parallel computer [3], the problem of system size has been almost solved. The problem of time extension is rather difficult, but for example, Parallel replica dynamics [4] are recognized as an effective method for parallel computers to accelerate molecular-dynamics simulations that require time-extending computations such as infrequent events.

Empirical potentials in atomistic simulations might be insufficient in the ability to express interatomic bonds, thus machine learning potentials (MLPs) have been developed to overcome this deficiency. MLPs aim not only to significantly relax the limitation in the system size and time scale of first-principles calculations but also to resolve the essential restrictions on atomic bonds of empirical potentials. MLPs estimate interatomic interactions using machine learning techniques for comprehensive first-principles calculation results for a given crystal structure dataset. More specifically, MLPs define structural features that represent adjacent atomic environments and describe a relationship between the potential energy of the crystal structure and the structural features using a machine learning model such as Gaussian process model [5, 6, 7, 8, 9], high-dimensional neural network model [10, 11, 12, 13, 14, 15], or polynomial model [16, 17, 18]. Since the number of structural features is large and the machine learning model is very flexible, atomistic simulations using MLPs are as accurate as first-principles calculations.

By following the framework of MLPs, a large number of potentials can systematically be generated from various hyperparameters of MLP. From these systematically generated MLPs, the optimum interatomic potential should be chosen in terms of the excellent balance between accuracy and computational cost, which forms Pareto optimal points. Since all MLPs on Pareto optimal points are solutions for multi-objective optimization of computational cost and accuracy, the optimal MLP employed in simulation should be selected with additional criteria specified for the problem in interest. In the first part of this thesis, this point is discussed using grain boundary energy as a concrete example.

A certain problem setting requires so high precision potentials that large-scale or time-extending calculation is usually difficult, even though the amount of calculation is smaller than that of first-principles calculation. For example, when the typical structures are unknown or cannot be set, the most accurate interatomic potential currently available should be employed. When accuracy is more important than efficiency, such as in the

search for crystal structures, the most accurate potential should be used. Therefore, it is necessary to develop a method to execute high-precision MLPs at high speed.

Material science simulations using MLPs are expected to solve a large system and time-extending problem with high accuracy, hence accelerating implementations have already been developed for most of MLPs. As an example of a particularly advanced study, the Spectral Neighbor Analysis Potential (SNAP) was implemented and demonstrated high efficiency using massively parallelized GPUs [19, 20]. Calculations of billions of atoms using SNAP were reported and the performance improvement with GPUs is overwhelming for large systems. However, it is known that SNAP can be improved by adding quadratic terms in the bispectrum components to the structural feature [21]. As another example, the Atomic Cluster Expansion (ACE) [18] is known as an MLP with a complete descriptive ability of the structural feature [22], but it does not support GPUs at the time of writing this thesis. Therefore, ACE may not be yet applicable to spatially large problems.

PolyMLP [23, 24] is an MLP developed with polynomial invariants that are enumerated using a group-theoretic procedure and derived from spherical harmonics. The smooth overlap of atomic positions kernel [25, 26], the bond-orientational order parameters [27], SNAP [17], etc., can be obtained as special cases or minor variations of the descriptor of PolyMLP. Although the structural features of PolyMLP are a part of those of ACE, the polynomial model formalism of PolyMLP is more general than ACE. The structural features of PolyMLP are complete like ACE or the moment tensor potentials (MTP) [28]. PolyMLP is one of the MLPs with the highest ability to reproduce PES, but the amount of computation is enormous instead. If PolyMLP is implemented in parallel on multiple GPUs, it will be possible to calculate problems of the comparable or larger system size with higher accuracy and larger scale than existing MLPs. Thus, the performant implementation of PolyMLP is expected to provide one of the most reliable ways for calculating realistic materials properties with high accuracy. Therefore, the author believes that the development of the implementation of PolyMLP on multiple GPUs, which is the second theme of this thesis, should contribute to materials science.

1.1 Outline of the Thesis

This thesis summarizes the results of verifying the predictive power of MLP and the results of examining the high-efficiency computation of MLP. The global search of grain boundary structures using PolyMLP [23, 29, 24] and the development of a performant implementation of PolyMLP using parallelized GPUs are discussed.

In Chapter 2, angle dependence of grain boundary structure and grain boundary energy for face-centered-cubic elemental metals Ag, Al, Au, Au, Pd, and Pt are discussed using PolyMLP. To determine the optimum PolyMLPs for the calculation of grain bound-

aries, grain boundary energies of test structures are verified for all Pareto-optimal MLPs. The most stable grain boundary structures are obtained via global search by applying various rigid body displacements to each grain boundary structure model. It is verified how much predictive power PolyMLP has for defect structures not included in the training data.

In Chapter 3, the development of performant implementation by using GPUs based on the spatial decomposition of the simulation box is discussed. The discussion is in two stages: speeding up iterative processes of potential calculation and applying spatial decomposition. For the iterative processes, a method to speed up utilizing parallel processing in a single GPU is discussed. In the spatial decomposition, boundary conditions for using inter-processor communication in PolyMLP computation are discussed. By measuring the execution speed for various system sizes, the effectiveness and application range of the performant implementation is evaluated.

Chapter 2

Application of machine learning potentials to predict grain boundary properties in fcc elemental metals

Abstract

Accurate interatomic potentials are in high demand for large-scale atomistic simulations of materials that are prohibitively expensive by density functional theory (DFT) calculation. In this study, the author applies machine learning potentials in a recently constructed repository to the prediction of the grain boundary energy in face-centered-cubic elemental metals, i.e., Ag, Al, Au, Cu, Pd, and Pt. The systematic application of machine learning potentials shows that they enable us to predict grain boundary structures and their energies accurately. The grain boundary energies predicted by the MLPs are in agreement with those calculated by DFT, although no grain boundary structures were included in training datasets of the present MLPs.

2.1 Introduction

Grain boundaries are interfaces between differently oriented crystals of the same phase [30]. The microstructures of grain boundaries can affect various properties of polycrystalline materials, including mechanical, thermal, and electrical properties [31, 32, 33, 34]. Thus, an attractive topic in materials science has been to establish the relationship between the properties of crystalline materials and grain boundary structures. Many theoretical studies have been made to cover a broad range of grain boundary structures and their excessive energies. Early fundamental studies employed pair potentials, such as the Lennard–Jones and Morse forms, to investigate the generic properties of grain boundaries such as the presence of cusps in a map of the rotation angle and the grain

boundary energy [35, 36, 37]. Empirical interatomic potentials such as the Finnis–Sinclair (FS) potentials [38] and embedded atom method (EAM) [39] potentials have been widely used to investigate symmetric and asymmetric grain boundaries of metallic materials. Quantitative predictions are becoming possible [40, 41, 42, 43, 44, 45, 46, 47, 48, 49], and strong correlations between theoretical and experimental grain boundary energies have been shown, especially for grain boundaries in elemental Al and Ni, which exhibit low grain boundary energies [50, 51]. However, the prediction error in the grain boundary energy may be significant in grain boundaries showing higher grain boundary energies. This error originates from the fact that their microscopic grain boundary structures differ from the atomic environment used to estimate interatomic potentials.

Density functional theory (DFT) calculation [1, 2] is an alternative way to predict grain boundary properties accurately. However, DFT calculation is practically impossible to apply to large-scale models of grain boundaries owing to its computational cost. Therefore, interatomic potentials that enable us to predict grain boundary properties accurately have been in high demand. Over the last decade, many groups have proposed frameworks to develop machine learning potentials (MLPs) based on extensive datasets generated by DFT calculation [10, 11, 5, 12, 13, 14, 15, 6, 52, 8, 9, 53, 54, 55, 56, 21, 57, 58, 59, 60, 61, 62]. The MLPs significantly improve the accuracy and transferability of interatomic potentials. Also, MLPs themselves are becoming available, such as those in Machine Learning Potential Repository [29] developed by Seko.

In this paper, the author demonstrates the predictive power of MLPs in the MLP repository for grain boundary properties. The author systematically evaluates the structures and excessive energies of $\langle 100 \rangle$ symmetric tilt grain boundaries (STGBs), $\langle 110 \rangle$ STGBs, and $\langle 100 \rangle$ pure-twist grain boundaries in the face-centered-cubic (fcc) elemental metals of Ag, Al, Au, Cu, Pd, and Pt. They are compared with those obtained from EAM potentials and DFT calculations. The MLP repository contains a set of Pareto optimal MLPs with different trade-offs between accuracy and computational efficiency; hence, the author carefully determines appropriate MLPs to predict grain boundary properties.

2.2 Methodology

2.2.1 Modeling and structure optimization of grain boundaries

Macroscopic structures of grain boundaries are characterized by five geometrical degrees of freedom. The author chooses three variables to specify the direction of the rotation axis and the rotation angle, which describe the misorientation between crystal lattices, and two variables to specify the direction of the boundary plane normal [30]. For a given set of macroscopic variables, the microscopic structure is associated with three degrees of freedom regarding rigid body displacements: two components parallel to the boundary

plane and one component normal to the plane. Hence, the globally optimal microscopic structure for a given set of macroscopic variables is achieved by optimizing the three microscopic variables in terms of potential energy.

In this study, the author investigates only STGBs and pure-twist grain boundaries. The periodicity of an STGB is identified from the orthogonal projection of its coincident site lattice (CSL) to its boundary plane. Also, the periodicity of a pure-twist grain boundary is given by the orthogonal projection of its displacement shift complete (DSC) lattice to its boundary plane. Therefore, the author restricts the ranges of the two in-plane microscopic variables to a domain defined by the periodicity of the grain boundaries.

The author explores the globally optimal microscopic structure for a set of macroscopic variables using a multi-start method. The multi-start method involves local structure optimizations for a given set of initial structures and regards the structure with the lowest energy among the converged final structures as the globally optimal structure. The author uses the conjugate gradient method implemented in the LAMMPS code [3] for the local structure optimizations. Initial microscopic structures are introduced from a 4×4 grid for the two in-plane components and a sequence for the component normal to the boundary plane. In other words, one crystal is shifted relative to the other crystal by a vector identified with an in-plane grid point and a value from the sequence for the normal component. For each initial microscopic structure, a calculation model is generated using PYMATGEN [63]. This model contains two parallel boundaries perpendicular to the c-axis of the model, separated by fcc layers corresponding to four repetitions of a cell of the CSL. However, the local structure optimization starting from some of the initial microscopic structures fails to converge when using both the MLPs and the EAM potentials, as shown in the next section. These structures are ignored in finding the globally optimal microscopic structure. Note that the optimization of the microscopic structure is performed in the whole domain here, although it is more efficient to restrict the domain to its symmetrically nonequivalent domain.

2.2.2 Machine learning potentials

The author employs MLPs in MACHINE LEARNING POTENTIAL REPOSITORY [29] to obtain the globally optimal microscopic structures of STGBs and pure-twist grain boundaries. In the repository, a set of Pareto optimal MLPs with different trade-offs between accuracy and computational efficiency is available, from which one can choose an appropriate MLP in accordance with the target and purpose. Potential energy models of the MLPs are either a polynomial model of Gaussian-type pairwise structural features or a polynomial model of polynomial invariants for the O(3) group, which are derived by a group-theoretical approach [23]. A brief description of the potential energy models and the structural features used for developing the MLPs is given in Appendix A.

The Pareto optimal MLPs in the repository have been developed using a dataset generated from structure generators. For Ag, Al, Au, and Cu, the Pareto optimal MLPs are adopted from what developed from a structure generator set composed of the fcc, body-centered-cubic (bcc), hexagonal-close-packed (hcp), simple cubic (sc), ω, and β tin structures. The dataset is composed of 3,000 structures constructed by introducing random lattice expansion, random lattice distortion, and random atomic displacements into a supercell of the equilibrium structure for one of the structure generators. For Pd and Pt, another set of 82 prototype structures are adopted as the structure generator set because the dataset derived from the six structure generators is not available in the repository. The dataset consists of 10,000 structures generated by the same procedure as above. For all structures in the dataset, DFT calculations were performed using the plane-wave-basis projector augmented wave method [64] within the Perdew–Burke–Ernzerhof exchange-correlation functional [65] as implemented in the VASP code [66, 67, 68]. Note that the datasets contain no structures generated from grain boundary models.

2.3 Results and discussion

First, the author chooses an accurate MLP requiring only a reasonable computational time to investigate the whole set of grain boundaries. A practical approach to selecting an MLP from the whole set of Pareto optimal MLPs is to find an MLP with high computational cost performance in terms of the prediction error for a test dataset. It can be obtained from the distribution of Pareto optimal MLPs shown in Appendix B. Another practical approach is to examine the convergence behavior of the target property in terms of the computational cost using the whole set of Pareto optimal MLPs. The author adopts the latter approach to select an MLP in this study. Therefore, the author systematically calculates the grain boundary energies of five grain boundaries using the whole set of Pareto optimal MLPs for each elemental metal. They are the $\Sigma5$ $\langle100\rangle$ STGB (at 53.1 degrees), the $\Sigma3$ $\langle110\rangle$ STGB (at 70.5 degrees), the $\Sigma3$ $\langle110\rangle$ STGB (at 109.5 degrees), the $\Sigma9$ $\langle110\rangle$ STGB (at 38.9 degrees), and the $\Sigma5$ $\langle100\rangle$ pure-twist grain boundary (at 36.9 degrees), the calculation models for which can be represented by a small number of atoms.

Figure 2.3.1 shows the convergence behavior of the grain boundary energy in terms of the computational time, obtained using the whole set of Pareto optimal MLPs. The grain boundary energy is identical to the lowest energy among the grain boundary energies of the microscopic structures. The grain boundary energy of a microscopic structure is measured from the energy of the equilibrium fcc structure. The computational time corresponding to the model complexity of an MLP is the elapsed time normalized by the number of atoms for a single point calculation of the energy, the forces, and the stress

tensors [1]. As can be seen in Fig. 2.3.1, the grain boundary energy converges well in all of the elemental metals and grain boundaries. Consequently, successive calculations for the whole set of grain boundaries are performed using the MLP that requires the lowest computational time among the MLPs showing convergence.

Table 2.1 lists the model parameters of the selected MLPs. As a consequence of the convergence behavior, fast MLPs are selected for Ag and Cu, while computationally expensive MLPs are selected for the others. Table 2.1 also shows the prediction errors for the datasets used in developing the MLPs. The MLPs for Pd and Pt show significant prediction errors, which originate from the fact that the datasets contain many hypothetical structures such as the graphite-type structure. Although the selected MLPs exhibit significant prediction errors for such abnormal structures, they show much smaller prediction errors for typical metallic structures, including grain boundary structures, as shown above.

The author also examines the transferability of the MLPs to the prediction of the grain boundary structures and energies because the datasets used in developing the MLPs contain no grain boundary structures. Therefore, the author evaluates the grain boundary energies of the $\Sigma 3$ $\langle 110 \rangle$ STGB (at 70.5 degrees), the $\Sigma 3$ $\langle 110 \rangle$ STGB (at 109.5 degrees), the $\Sigma 9$ $\langle 110 \rangle$ STGB (at 38.9 degrees), the $\Sigma 5$ $\langle 100 \rangle$ STGB (at 53.1 degrees), and the $\Sigma 5$ $\langle 100 \rangle$ pure-twist grain boundary (at 36.9 degrees) by DFT calculation, and compare them with those predicted using the MLPs. Figure 2.3.1 shows the DFT values of the grain boundary energy only for the grain boundary structures, DFT calculations for which converge successfully [2]. They are close to the grain boundary energies of the selected MLPs. Therefore, the selected MLPs should have high predictive power for grain boundary structures and their energies even though no grain boundary structures were used to develop the MLPs.

After confirming the transferability of the MLPs, the author calculates the energies of the grain boundary structures: $\langle 100 \rangle$ STGBs ($\Sigma 5$, $\Sigma 13$, $\Sigma 17$, $\Sigma 25$, $\Sigma 29$, $\Sigma 41$), $\langle 110 \rangle$ STGBs ($\Sigma 3$, $\Sigma 9$, $\Sigma 11$, $\Sigma 17$, $\Sigma 19$, $\Sigma 27$, $\Sigma 33$, $\Sigma 41$, $\Sigma 43$), and $\langle 100 \rangle$ pure-twist grain boundaries ($\Sigma 5$, $\Sigma 13$, $\Sigma 17$, $\Sigma 25$, $\Sigma 29$, $\Sigma 37$, $\Sigma 41$). Most of them are represented by large-scale models, hence they cannot be calculated by DFT calculation because of the large computational resources required. The number of atoms included in the grain boundaries ranges from 96 to 2112. Figure 2.3.2 shows the optimized STGB structures of some STGBs in Ag. Figure 2.3.3 shows the rotation angle dependence of the grain boundary energy obtained using the MLPs and EAM potentials [70, 71, 72, 73, 74, 75]. The values of the grain boundary energy in Al, Cu, and Pd computed using the MLPs are consistent with those computed using the EAM potentials and those computed by DFT

[1] The computational time is estimated using a single core of Intel Xeon E5-2695 v4 (2.10GHz).

[2] The DFT values of the grain boundary energy for $\Sigma 9$ STGB are missing in Ag, Au, Cu, Pd, and Pt. The electronic structure calculation failed to converge for some of the elements, and the structure optimization did not finish within a reasonably long time for the others.

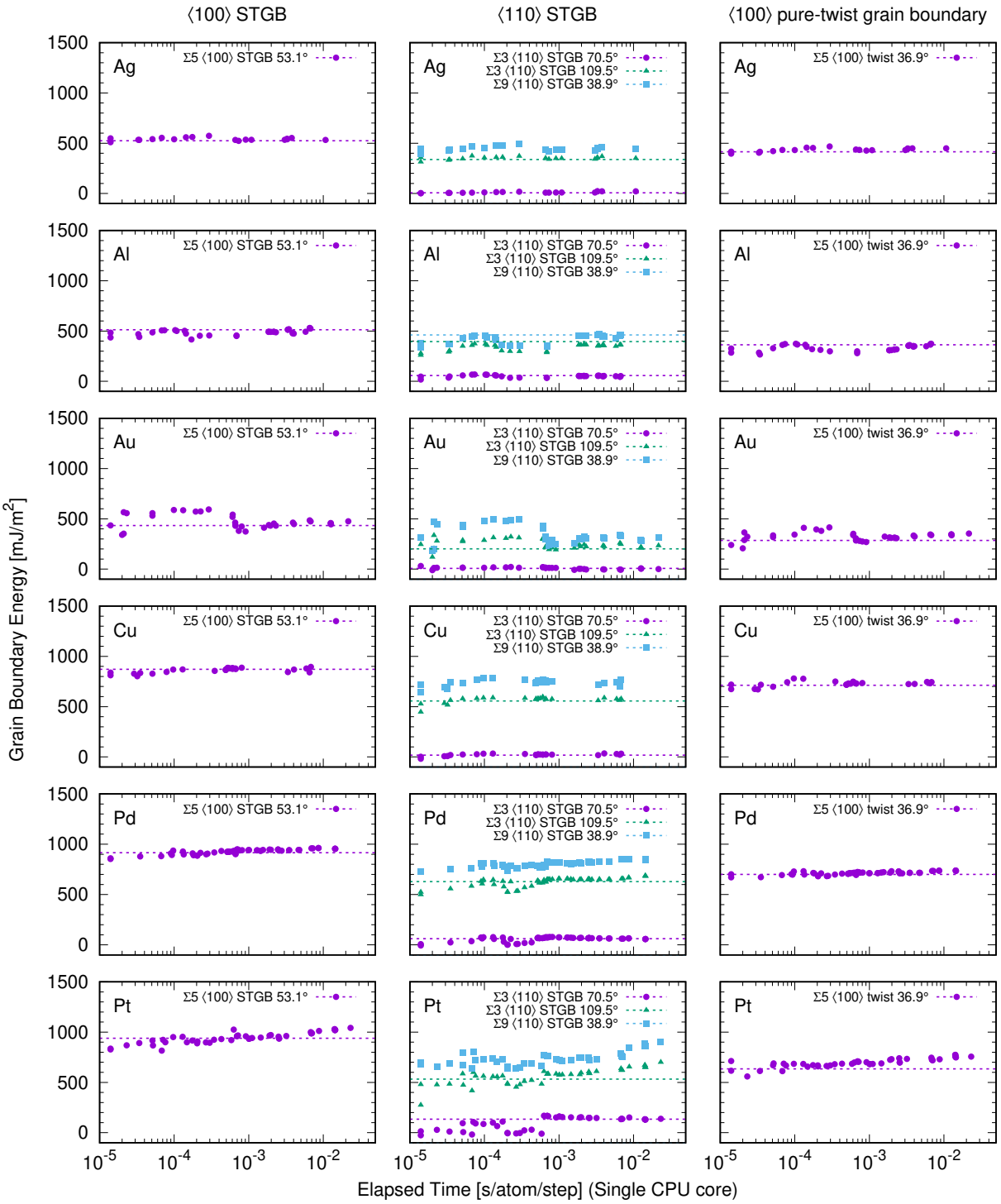


Figure 2.3.1: Grain boundary energies of Σ5 ⟨100⟩ STGB in 53.1 degrees, Σ3 ⟨110⟩ STGB in 70.5 degrees, Σ3 ⟨110⟩ STGB in 109.5 degrees, Σ9 ⟨110⟩ STGB in 38.9 degrees, and Σ5 ⟨100⟩ pure-twist grain boundary in 36.9 degrees for elemental Ag, Al, Au, Cu, Pd, and Pt, predicted using the Pareto optimal MLPs. The grain boundary energies computed by DFT calculation are also shown by broken lines.

Table 2.1: Model parameters of the MLPs used to estimate the grain boundary structures and energies. The identification of the feature type, the model type, and the polynomial orders can be found in the Appendix A.

	Ag	Al	Au	Cu	Pd	Pt
MLP-ID	pair-44	gtinv-336	gtinv-111	pair-23	gtinv-722	gtinv-533
RMSE (energy) [meV/atom]	2.2	0.8	0.7	2.2	6.3	12.9
RMSE (force) [eV/Å]	0.010	0.008	0.012	0.013	0.097	0.172
Time [ms/atom/step] [69]	0.05	1.85	0.66	0.04	0.52	0.63
Number of coefficients	815	1100	475	285	500	1595
Feature type	Pair	Invariants	Invariants	Pair	Invariants	Invariants
Cutoff radius [Å]	7.0	8.0	6.0	7.0	6.0	6.0
Number of radial functions	15	15	10	10	5	5
Model type	2	3	3	2	4	2
Polynomial order (function F)	3	3	3	3	2	2
Polynomial order (invariants)	–	3	3	–	3	3
Spherical harmonics truncation $\{l^{(2)}_{\max}, l^{(3)}_{\max}\}$	–	[4, 4]	[4, 4]	–	[4, 0]	[4, 2]

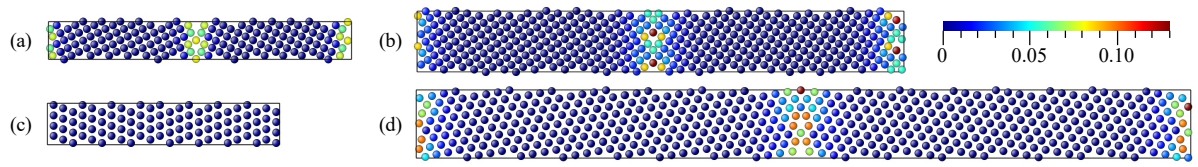


Figure 2.3.2: Optimized structures of (a) Σ5 ⟨100⟩ STGB (at 53.1 degrees), (b) Σ13 ⟨100⟩ STGB (at 67.3 degrees), (c) Σ3 ⟨110⟩ STGB (at 70.5 degrees), and (d) Σ33 ⟨110⟩ STGB (at 20.0 degrees) in Ag. They are represented by 160, 416, 96, and 1056 atoms, respectively. They are visualized using AtomEye [76], and colors are assigned to atoms according to their local von Mises shear strain invariant.

calculation. Therefore, both the MLPs and the EAM potentials have high predictive power for the grain boundary structures and their energies. In Ag, Au, and Pt, the values of the grain boundary energy computed using the MLPs are almost the same as those computed by DFT calculation, whereas they deviate from those computed using the EAM potentials. The MLPs should be more reliable than the EAM potentials for obtaining not only the grain boundary structures and their energies but also the other defect structures in Ag, Au, and Pt. Note that a fine sequence is required for the component normal to the boundary plane to obtain converged microscopic structures when using the EAM potentials for Ag and Au. This implies that the EAM potentials for Ag and Au lack accuracy for predicting the potential energy surface around the globally optimal microscopic structure.

For every grain boundary, the grain boundary energy in Cu, Pd, and Pt are higher than that in Ag, Al, and Au, as shown in Fig. 2.3.3. This trend can be qualitatively understood by considering a simple approximation within the elasticity theory. Using the Read-Shockley equation [77], the shear stress component of the grain boundary energy is obtained by integrating the contributions of dislocations distributed evenly on the interface. Given the shear modulus G and cubic lattice constant a, the excess energies of grain boundaries are approximately proportional to a coefficient Ga. By employing the Voigt averages calculated from the elastic constants of single crystals [78], the coefficients for Ag, Al, Au, Cu, Pd, and Pt are estimated as 1.3, 1.0, 1.2, 1.8, 1.8, and 2.2, respectively, which are normalized by the coefficient for Al. They can be classified into two groups, which is consistent with the grain boundary energy trend.

2.4 Conclusion

The author has examined the predictive power of MLPs in an MLP repository for grain boundary properties by systematically evaluating the grain boundary energy for ⟨100⟩ STGBs, ⟨110⟩ STGBs, and ⟨100⟩ pure-twist grain boundaries in the fcc elemental metals of Ag, Al, Au, Cu, Pd, and Pt. In every elemental metal, the values of the grain boundary

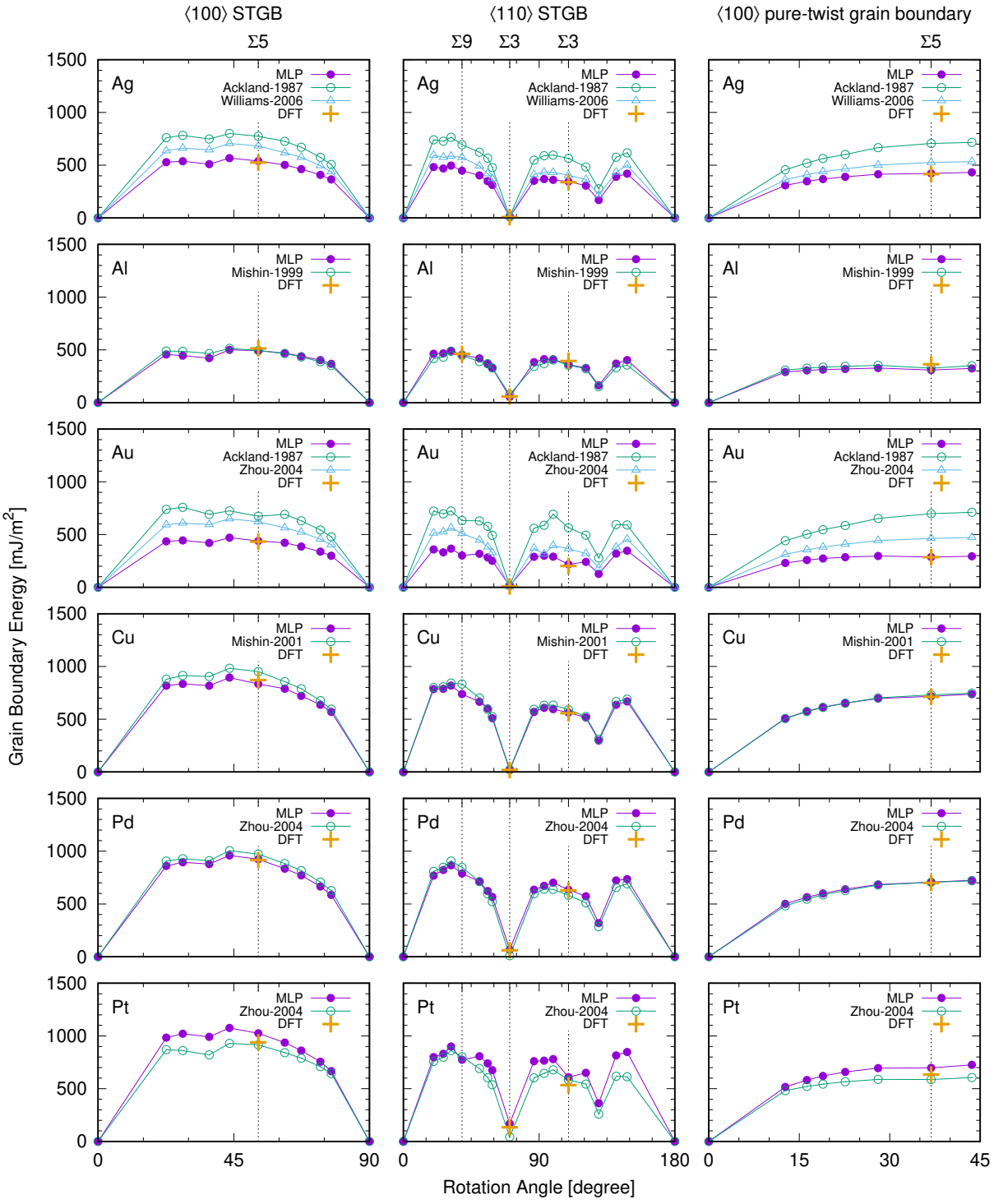


Figure 2.3.3: Rotation angle dependence of the grain boundary energy for ⟨100⟩ STGBs, ⟨110⟩ STGBs, and ⟨100⟩ pure-twist grain boundaries for elemental Ag, Al, Ag, Cu, Pd, and Pt, predicted using the MLPs. For comparison, the grain boundary energies predicted using EAM potentials for Ag [70, 71], Al [72], Au [70, 73], Cu [74], Pd [73], and Pt [73] are shown by open symbols. The grain boundary energies computed by DFT calculation are also shown by crosses.

energy computed using the MLP are consistent with those computed by DFT calculation. The author emphasizes that the training datasets used to develop the MLPs contain no grain boundary structures. Therefore, the consistency indicates that the MLPs have high predictive power for the grain boundary structures and their energies. The present results also imply that the MLPs in the repository, including those for other systems, should be useful in accurately predicting grain boundary properties and other complex defect properties.

Appendix A: Potential Energy Models

This section summarizes potential energy models used for developing MLPs in Machine Learning Potential Repository [29]. In MLPs of the repository, the short-range part of the total energy for a structure is expressed by the sum of the atomic energy. The atomic energy is given by a function of invariants for the O(3) group [25, 16] as

$$E^{(i)} = F\left(d_1^{(i)}, d_2^{(i)}, \cdots\right), \tag{2.4.1}$$

where $d_n^{(i)}$ denotes a structural feature or an invariant derived from order parameters representing the neighboring atomic density of atom i. In the repository, a set of structural features derived only from radial functions (`feature type = pair`) and a set of polynomial invariants of the O(3) group derived from radial and spherical harmonic functions (`feature type = invariants`) are employed for developing MLPs.

The repository uses polynomial functions as function F representing the relationship between the atomic energy and a given set of structural features, $D = \{ d_1, d_2, \cdots \}$. The polynomial functions with regression coefficients $\{ w \}$ are given as follows.

$$\begin{aligned} F_1(D) &= \sum_i w_i d_i \\ F_2(D) &= \sum_{\{i,j\}} w_{ij} d_i d_j \\ F_3(D) &= \sum_{\{i,j,k\}} w_{ijk} d_i d_j d_k \\ &\vdots \end{aligned} \tag{2.4.2}$$

A potential energy model is identified with a combination of the polynomial functions and structural features.

In this paper, MLPs with the following four types of the potential energy model are selected as listed in Table 2.1, although six types of the potential energy model have been introduced in the repository. When a set of pairwise structural features is described

as $D^{(i)}_{\text{pair}}$, the model (`model type = 2, feature type = pair`) is a polynomial of the pairwise structural features with their cross terms, expressed as

$$E^{(i)} = F_1\left(D^{(i)}_{\text{pair}}\right) + F_2\left(D^{(i)}_{\text{pair}}\right) + F_3\left(D^{(i)}_{\text{pair}}\right) + \cdots . \tag{2.4.3}$$

The other three models are derived from the polynomial invariants. When a set of the polynomial invariants is expressed by the union of sets of pth-order polynomial invariants $D^{(i)}_p$ as

$$D^{(i)} = D^{(i)}_{\text{pair}} \cup D^{(i)}_2 \cup D^{(i)}_3 \cup D^{(i)}_4 \cup \cdots . \tag{2.4.4}$$

The model (`model type = 2, feature type = invariants`) is given by a polynomial of the polynomial invariants as

$$E^{(i)} = F_1\left(D^{(i)}\right) + F_2\left(D^{(i)}\right) + F_3\left(D^{(i)}\right) + \cdots . \tag{2.4.5}$$

The model (`model type = 3, feature type = invariants`) is the sum of a linear polynomial form of the polynomial invariants and a polynomial of pairwise structural features, described as

$$E^{(i)} = F_1\left(D^{(i)}\right) + F_2\left(D^{(i)}_{\text{pair}}\right) + F_3\left(D^{(i)}_{\text{pair}}\right) + \cdots . \tag{2.4.6}$$

The model (`model type = 4, feature type = invariants`) is the sum of a linear polynomial form of the polynomial invariants and a polynomial of pairwise structural features and second-order polynomial invariants. This is written as

$$E^{(i)} = F_1\left(D^{(i)}\right) + F_2\left(D^{(i)}_{\text{pair}} \cup D^{(i)}_2\right) + \cdots . \tag{2.4.7}$$

Appendix B: Pareto Optimality

Figure 2.4.1 shows the Pareto optimal MLPs for elemental Ag, Al, Au, Cu, Pd, and Pt. The distribution of MLPs is obtained by a systematic grid search to find optimal parameters controlling the accuracy and computational efficiency. The prediction error is estimated using the root mean square (RMS) error of the energy for the test dataset. The computational efficiency is estimated using the elapsed time to compute the energy, the forces and the stress tensors of a structure with 284 atoms. The prediction error converges more slowly than the grain boundary energy, which originates from the fact that the prediction error is estimated from the test dataset composed of a wider variety of structures.

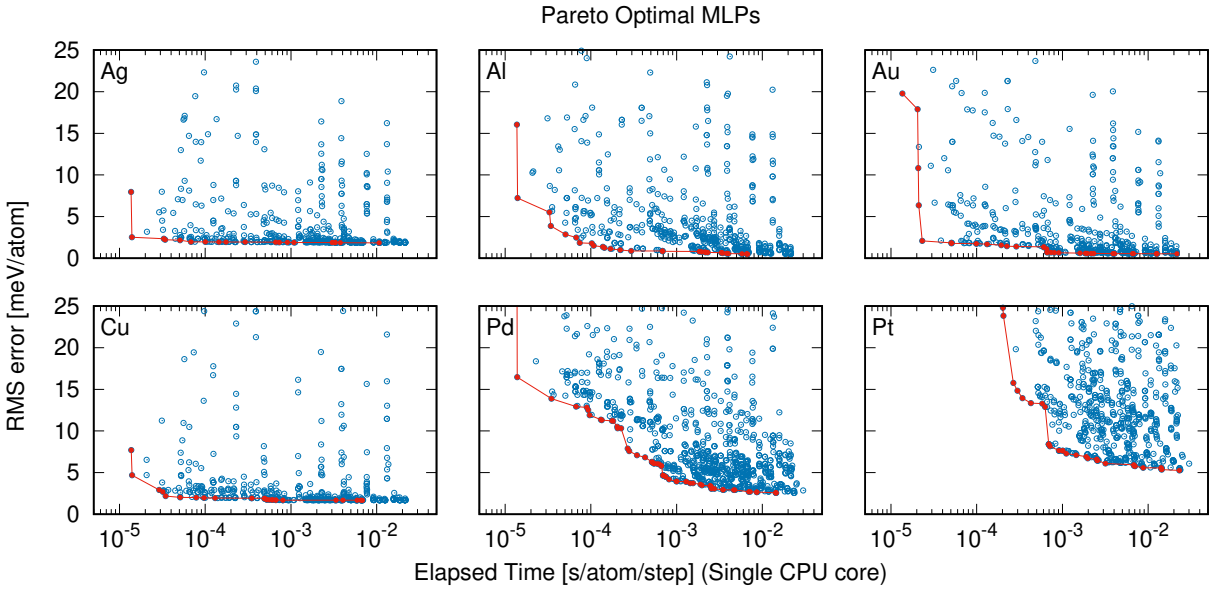


Figure 2.4.1: Distribution of MLPs for Ag, Al, Au, Cu, Pd, and Pt [29]. The elapsed time for a single point calculation is estimated using a single core of Intel® Xeon® E5-2695 v4 (2.10 GHz). The closed red circles show the Pareto optimal points of the distribution.

Chapter 3

Performant Implementation of Linearized Machine Learning Potential

Abstract

Efficient computation of machine learning interatomic potentials (MLIPs) has been expected to enable quantitative estimation of the properties of the materials. There are successful implementations of MLIPs that have been accelerated by spatial decomposition and/or using parallelized processing units to access large problems [79, 80, 81, 82, 83, 20, 19, 84]. The current investigation involved a method for parallelizing polyMLP [23, 29, 24] to calculate energy and force with many atoms efficiently. By modifying the method to decompose the simulation box spatially, polyMLP is efficiently implemented to be capable of high-precision large-scale molecular dynamics. Parallelized on 40 GPUs reached 1100 times the performance on CPU single core for a large system of 256k atoms. Large-scale problems that cannot be executed by a single GPU due to insufficient memory capacity can now be executed by using multiple GPUs. Spatial parallelization with CPUs linearly accelerated even for a relatively small number of 500 atoms. These results suggest that high-speed and accurate calculations can be performed for a wide range of system sizes using either CPU or GPU according to the system size.

3.1 Introduction

Machine learning interatomic potentials (MLIPs) have received much attention in recent years due to their prediction accuracy comparable to the density functional theory (DFT) and high computational efficiency [11, 5, 17, 21, 85, 86, 28, 18, 79, 87, 88, 89, 90, 91, 92, 93, 83, 94, 23, 29, 24]. These MLIPs fit parameters using machine learning techniques to reproduce potential energy surfaces (PESs) which are calculated by DFT. A series of studies of polyMLP by Seko et al. [23, 29, 24] demonstrated excellent accuracy for

use in materials applications such as phonon dispersion or generalized stacking fault energy calculations. However, these MLIPs were approximate 10 to 1000 times more computationally demanding than empirical ones [24], such as embedded atom method (EAM) potentials; it has significantly limited their practical range of use.

Calculating interatomic potentials in a large system requires so many iterative calculations that it has been established to use Graphical Processing Units (GPUs) as an effective way for acceleration. GPUs are highly parallel computational engines especially suitable for simple calculations with large loops and recently supercomputers equipped with many GPUs have been actively built. Although considerable effort has been made for many simple interatomic potentials to accelerate using GPUs, there are only a few MLIPs accelerated using GPUs [19, 81] at the time of writing this thesis. One of the leading implementations of the MLIPs is spectral neighbor analysis potential (SNAP), which is parallelized using Kokkos framework [19, 20]. Kokkos is a C++ library for writing performance portable applications targeting high-performance computing (HPC) platforms to manage both fine-grain data parallelism and memory access patterns [95, 96, 97]. This implementation of SNAP with Kokkos can indeed handle large problems up to 20 billion atoms with high accuracy. However, SNAP can be improved by extending to quadratic terms in the bispectrum components or chemically-labeled descriptors [21, 86], as well as neural network energy models [85]. There is therefore still a strong need for faster calculation of machine learning interatomic potential with more comprehensive descriptors and generic models.

Parallelizing the calculation of the interatomic potential, especially for force calculation, involves an appropriate spatial decomposition according to the model of the potential. The spatial decomposition requires special care for MLIPs. The classes of MLIP registered in the main branch of LAMMPS commonly use a combination of the full neighbor style and Newton's 3rd law enabled for particles on processor borders. In this paper, it is discussed how PolyMLP can be efficiently implemented on distributed processors with spatial decomposition. This thesis reports benchmarks both by Central Processing Units (CPUs) and GPUs.

In this thesis, the implementation for GPUs presented is orders of magnitude faster than the serial implementation for CPUs in systems with more than 10,000 particles. In addition, the implementation for CPUs with the spatial decomposition is, unexpectedly, also orders of magnitude faster than the serial implementation for CPUs even in a system with a relatively small number of 500 particles. This acceleration in a small number of particles would be a characteristic phenomenon seen when the energy model involves extraordinarily complicated computation with a large loop.

This chapter is organized as follows: Section 3.2.1 deals with the revisitation of PolyMLP, Section 3.2.2 the algorithmic aspect of PolyMLP, Section 3.2.4 the Benchmark conditions, Section 3.3 the Results and Discussion, Section 3.4 the Conclusion.

3.2 Methodology

When considering parallelization of polyMLP, it is necessary to distinguish between the parts that can make use of the same procedure as in preceding studies and the parts that need improvements. To distinguish these parts and improve them appropriately, general aspects of descriptors and models are revisited. Also, a concise derivation of the order parameter, the structural feature, and the energy model are provided to clarify what parts should be parallelized. An algorithmic aspect of polyMLP is discussed and presented to provide a reproducible description of the implementation. The discussion involves the adjoint method reducing redundant calculations, efficient and accurate computation of spherical harmonics, and types of spatial decomposition. All these topics were integrated into the performant implementation.

3.2.1 Machine Learning Potential Revisited

Here, the definition of polyMLP is reconsidered and the necessary information in parallel computing is organized. To simplify symbols, this section uses the following notation:

- The index of the atom is written in the lower left
- The index of the irreducible representation is enclosed in parentheses in the upper left of the symbol
- For projection matrices, the original representation and the projected representation are distinguished by vertical bars in parentheses
- Representation matrices are treated as tensor

Density expansion around atom i by arbitrary bases of its vector space

Atomic density centering an atom i is a sum of delta functions shifted to the atoms within a cutoff distance from the atom i,

$$ {}_i\rho\left(\mathbf{r}\right) \overset{\text{def}}{=} \sum_{j \in {}_i\mathcal{N}} \delta\left(\mathbf{r} - {}_j\mathbf{r}\right), \tag{3.2.1} $$

where ${}_i\mathcal{N}$ is a list of neighbor atoms around atom i.

Theorem 3.1 (Fourier Expansion)**.** *Let $\mathcal{H}$ be a separable and infinite-dimensional Hilbert space, $\{\phi_k\}_{k=1,2,\ldots}$ be the complete orthonormal system of $\mathcal{H}$, then arbitrary $u \in \mathcal{H}$ can be*

expanded as:

$$u = \sum_{k=1}^{\infty} u_k \phi_k, \qquad u_k = \langle \phi_k, u \rangle . \tag{3.2.2}$$

The atomic density can be decomposed approximately into a linear combination of bases through the Fourier Expansion.

Theorem 3.2 (Atomic density decomposition)**.** *An atomic density can be decomposed via Theorem 3.1:*

$$ {}_i\rho(\mathbf{r}) = \sum_q \left\langle {}_i\phi_q, {}_i\rho \right\rangle \cdot {}_i\phi_q(\mathbf{r}) \tag{3.2.3}$$

$$ = \sum_q \left(\int_{\mathbf{r}\in\Omega} {}_i\rho(\mathbf{r}) \, \overline{{}_i\phi_q(\mathbf{r})} \mathrm{d}\mathbf{r} \right) \cdot {}_i\phi_q(\mathbf{r}), \tag{3.2.4}$$

where the atomic density is a summation of delta function and ϕ_q is q-th basis function of the vector space.

The expansion formula includes the integral over the whole volume Ω, but the domain of the atomic density is limited around a cutoff length r_{cut}, thus the atomic density can be calculated by the summation over i-th neighbor list, ${}_i\mathcal{N}$, in lieu of the integration over Ω.

Lemma 3.1. *An inner product of an atomic density with a function can be calculated as a summation over its neighbor list:*

$$\left\langle {}_i\phi_q, {}_i\rho \right\rangle = \sum_{j\in {}_i\mathcal{N}} \overline{{}_i\phi_q({}_j\mathbf{r})}. \tag{3.2.5}$$

Proof. Substituting (3.2.1) into the expression (3.2.5), getting

$$\left\langle {}_i\phi_q, {}_i\rho \right\rangle = \int_{\mathbf{r}\in\Omega} {}_i\rho(\mathbf{r}) \, \overline{{}_i\phi_q(\mathbf{r})} \mathrm{d}\mathbf{r} \tag{3.2.6}$$

$$ = \int_{\mathbf{r}\in\Omega} \sum_{j\in {}_i\mathcal{N}} \delta(\mathbf{r} - {}_j\mathbf{r}) \, \overline{\phi_q(\mathbf{r} - {}_i\mathbf{r})} \mathrm{d}\mathbf{r} \tag{3.2.7}$$

$$ = \sum_{j\in {}_i\mathcal{N}} \overline{\phi_q\left({}_j\mathbf{r} - {}_i\mathbf{r}\right)} \tag{3.2.8}$$

$$ = \sum_{j\in {}_i\mathcal{N}} \overline{{}_i\phi_q({}_j\mathbf{r})}, \tag{3.2.9}$$

where

$${}_i\phi_q(\mathbf{r}) \overset{\text{def}}{=} \phi_q(\mathbf{r} - {}_i\mathbf{r}) \tag{3.2.10}$$

is a basis function centering atom i. □

Thus it is possible to expand the atomic density by means of the Fourier expansion, whose coefficients are summation of local bases over each neighbor list:

$$ {}_i\rho\left(\mathbf{r}\right) = \sum_q \left(\sum_{j \in {}_i\mathcal{N}} \overline{{}_i\phi_q\left({}_j\mathbf{r}\right)} \right) {}_i\phi_q\left(\mathbf{r}\right). \tag{3.2.11} $$

The summation of basis function over i-th neighbor list is referred to as an *order parameter of atom* i, ${}_ia^q$:

$$ {}_ia_q \overset{\text{def}}{=} \left\langle {}_i\phi_q, {}_i\rho \right\rangle = \sum_{j \in {}_i\mathcal{N}} \overline{{}_i\phi_q\left({}_j\mathbf{r}\right)} \tag{3.2.12} $$

$$ = \sum_{j \in {}_i\mathcal{N}} \overline{\phi_q\left({}_j\mathbf{r} - {}_i\mathbf{r}\right)} = \sum_{j \in {}_i\mathcal{N}} \overline{\phi_q\left({}_{ij}\mathbf{r}\right)}, \tag{3.2.13} $$

where ${}_{ij}\mathbf{r}$ is a vector from atom i to atom j. Finally the atomic density can be expanded using the order parameters and bases as

$$ {}_i\rho\left(\mathbf{r}\right) = \sum_q {}_ia_q\, {}_i\phi_q\left(\mathbf{r}\right). \tag{3.2.14} $$

Also note that, in practice, q should be truncated by balance between computational cost and accuracy.

Density expansion around atom i by basis of irreducible representation

If the Hamiltonian has a symmetric group $\mathcal{G}$, eigenfunctions corresponding to a energy span an irreducible representation of the symmetric group. In other words, when there is a set of irreducible representation accompanied by eigenfunctions, $\left({}^{(\alpha)}D, \left\{ {}^{(\alpha)}\psi_q \right\}_{q=1}^{d_\alpha} \right)$, these eigenfunctions correspond to the same eigenvalue of the Hamiltonian, and these eigenfunctions will be transformed by

$$ \left[{}_iR\, {}^{(\alpha)}_i\psi_q \right]\left(\mathbf{r}\right) = \sum_{q'} {}^{(\alpha)}_i\psi_{q'}\left(\mathbf{r}\right) {}^{(\alpha)}D^{q'}_q\left({}_iR\right), \tag{3.2.15} $$

where ${}_iR$ is a rotation around atom i. The atomic density can be expanded in the same way as above.

Corollary 3.1. *The atomic density can be decomposed via Theorem 3.1 even when using*

the bases of the irreducible representations.

$$ {}_i\rho\left(\mathbf{r}\right) = \sum_{\alpha} \sum_{q\in\{d_\alpha\}} \left\langle {}^{(\alpha)}_{i}\psi_q, {}_i\rho \right\rangle \cdot {}^{(\alpha)}_{i}\psi_q\left(\mathbf{r}\right) \tag{3.2.16} $$

$$ = \sum_{\alpha} \sum_{q\in\{d_\alpha\}} \left(\int_{\mathbf{r}\in\Omega} {}_i\rho\left(\mathbf{r}\right) \overline{{}^{(\alpha)}_{i}\psi_q\left(\mathbf{r}\right)} \mathrm{d}\mathbf{r} \right) \cdot {}^{(\alpha)}_{i}\psi_q\left(\mathbf{r}\right), \tag{3.2.17} $$

where ${}^{(\alpha)}_{i}\psi_q$ *is a* q*-th basis function of a irreducible representation* ${}^{(\alpha)}D$*, centering atom* i*.*

The conversion from integral to sum is almost trivial when the Lemma 3.1 is used.

Corollary 3.2. *An inner product of an atomic density with an eigenfunction of an irreducible representation can be calculated as a summation over its neighbor list:*

$$ \left\langle {}^{(\alpha)}_{i}\psi_q, {}_i\rho \right\rangle = \sum_{j\in{}_i\mathcal{N}} \overline{{}^{(\alpha)}_{i}\psi_q\left({}_j\mathbf{r}\right)}. \tag{3.2.18} $$

The atomic density can be expanded by both representation α and its dimension q as

$$ {}_i\rho\left(\mathbf{r}\right) = \sum_{\alpha}\sum_{q} \left(\sum_{j\in{}_i\mathcal{N}} \overline{{}^{(\alpha)}_{i}\psi_q\left({}_j\mathbf{r}\right)} \right) {}^{(\alpha)}_{i}\psi_q\left(\mathbf{r}\right). \tag{3.2.19} $$

The summation of eigenfunctions over i-th neighbor list also referred to as an *order parameter of atom* i, ${}^{(\alpha)}_{i}a_q$, that is

$$ {}^{(\alpha)}_{i}a_q \overset{\text{def}}{=} \sum_{j\in{}_i\mathcal{N}} \overline{{}^{(\alpha)}_{i}\psi_q\left({}_j\mathbf{r}\right)} \tag{3.2.20} $$

$$ = \sum_{j\in{}_i\mathcal{N}} \overline{{}^{(\alpha)}\psi_q\left({}_j\mathbf{r} - {}_i\mathbf{r}\right)} = \sum_{j\in{}_i\mathcal{N}} \overline{{}^{(\alpha)}\psi_q\left({}_{ij}\mathbf{r}\right)}. \tag{3.2.21} $$

Now the atomic density can be expanded using the order parameters and bases for each irreducible representation as

$$ {}_i\rho\left(\mathbf{r}\right) = \sum_{\alpha}\sum_{q} {}^{(\alpha)}_{i}a_q \, {}^{(\alpha)}_{i}\psi_q\left(\mathbf{r}\right). \tag{3.2.22} $$

In practice, α and q should be truncated by balance between computational cost and accuracy.

Harmonic basis for spherically symmetric potential

If particles move under a spherically symmetric potential $V_{\text{sph}}\left(\mathbf{r}\right)$, the potential will be invariant to any rotational operation R around the origin,

$$ \left[RV_{\text{sph}}\right]\left(\mathbf{r}\right) = V_{\text{sph}}\left(R^{-1}\mathbf{r}\right) = V_{\text{sph}}\left(\mathbf{r}\right), \tag{3.2.23} $$

therefore the Hamiltonian of these particles

$$H = \frac{\boldsymbol{p}^2}{2m} + V_{\text{sph}}(\mathbf{r}) \tag{3.2.24}$$

will be invariant to any rotational operation around the origin,

$$RHR^{-1} = H. \tag{3.2.25}$$

Their eigenfunctions should be written in polar coordinates using radial functions P_{nl} and spherical harmonic functions ${}^{(l)}Y_m$ as

$${}^{(l)}\psi_{nm}(\mathbf{r}) = P_{nl}(r)\,{}^{(l)}Y_m(\theta,\phi). \tag{3.2.26}$$

The rotational operation will convert these eigenfunctions into a linear combination of spherical harmonic functions belonging to the same irreducible representation l as

$$\left[R\,{}^{(l)}\psi_{nm}\right](\mathbf{r}) = \left[RP_{nl}\,{}^{(l)}Y_m\right](r,\theta,\phi) \tag{3.2.27}$$

$$= P_{nl}(r)\left[R\,{}^{(l)}Y_m\right](\theta,\phi) \tag{3.2.28}$$

$$= \sum_{m'} P_{nl}(r)\,{}^{(l)}Y_{m'}(\theta,\phi)\,{}^{(l)}D^{m'}_{m}(R). \tag{3.2.29}$$

If the polar component of the eigenfunction is defined as irrelevant to the index of l, the l on P is omitted:

$${}^{(l)}\psi_{nm}(\mathbf{r}) = P_n(r)\,{}^{(l)}Y_m(\theta,\phi).$$

Local decomposition of many-body Hamiltonian

It is assumed that a many-body Hamiltonian, H, can be composed of local hamiltonians centering each atom i.

Proposition 3.1 (Local decomposition of many-body Hamiltonian)**.** *If the many-body Hamiltonian, H, can be decomposed into local hamiltonians, the local hamiltonian is a summation of the local kinetic part and local potential part centering atom i*

$$H = \sum_i {}_ih \tag{3.2.30}$$

$${}_ih = \sum_{j\in{}_i\mathcal{N}} \frac{{}_j\boldsymbol{p}^2}{2\,{}_jm} + {}_iV\left({}_j\mathbf{r}\right). \tag{3.2.31}$$

If the local potentials are invariant to any rotational operation around each atom i:

$$[{}_iR\,{}_iV](\mathbf{r}) = {}_iV\left({}_iR^{-1}\mathbf{r}\right) = {}_iV(\mathbf{r}), \tag{3.2.32}$$

the eigenfunctions for local hamiltonians also should be written in local polar coordinates as

$${}^{(l)}_i\psi_{nm}(\mathbf{r}) = {}_iP_{nl}(r)\,{}^{(l)}_iY_m(\theta,\phi). \tag{3.2.33}$$

By Substituting (3.2.33) into (3.2.11) and (3.2.12), the density on each site is decomposed as

$${}_i\rho(\mathbf{r}) = \sum_l {}^{(l)}_ia^{nm}\,{}^{(l)}_i\psi_{nm}(\mathbf{r}), \tag{3.2.34}$$

$${}^{(l)}_ia_{nm} = \sum_{j\in{}_i\mathcal{N}} \overline{{}^{(l)}_i\psi_{nm}\left({}_j\mathbf{r}\right)} \tag{3.2.35}$$

$$= \sum_{j\in{}_i\mathcal{N}} \overline{{}_iP_{nl}\left({}_jr\right)\,{}^{(l)}_iY_m\left({}_j\theta,{}_j\phi\right)}. \tag{3.2.36}$$

Note that the radial functions, ${}_iP_{nl}(r)$, are real. The order parameter of atom i, ${}^{(l)}_ia_{nm}$, is a superposition of decomposition coefficients of an atomic density over its neighbor list, ${}_i\mathcal{N}$.

Structural Feature

Definition

A structural feature is a summation of products of the order parameters adjusted using coefficients so that it is invariant to any rotational operations in $\mathcal{G}$.

Definition 3.1 (Structural Feature)**.**

$${}^{(l_1\cdots l_p),\sigma}_id_n = \sum_{m_1\cdots m_p} {}^{(l_1)}_ia_{nm_1}\cdots{}^{(l_p)}_ia_{nm_p}\,{}^{(l_1\cdots l_p),\sigma}c_{m_1\cdots m_p} \tag{3.2.37}$$

Note that this is a combination of the order parameters that are the Fourier coefficients of the atomic density, but not the Fourier coefficients of the atomic density that are expanded on a product basis.

There are two known methods for calculating the coefficient, ${}^{(l_1\cdots l_p),\sigma}c_{m_1\cdots m_p}$. One is to combine two order parameters sequentially [79], and the other is to find it as an eigenvalue problem with a projection operator to the identity irreducible representation [23]. Here, the latter method by the projection operator is explained.

Definition 3.2 (Projection operator)**.** The projection operator, ${}^{(\beta)}P_{l(m)}$, is an operator

that gets the lth component of a particular irreducible representation, ${}^{(\beta)}D$, from a vector:

$$ {}^{(\beta)}P_{l(m)} \stackrel{\text{def}}{=} \frac{d_\beta}{g} \sum_{R \in \mathcal{G}} \overline{{}^{(\beta)}D^l_m(R)} R. \tag{3.2.38} $$

The projection operator to a particular irreducible representation, ${}^{(\beta)}P$, is an operator that gets the all components of the representation from a vector:

$$ {}^{(\beta)}P \stackrel{\text{def}}{=} \sum_l {}^{(\beta)}P_{l(l)} \tag{3.2.39} $$

$$ = \frac{d_\beta}{g} \sum_{R \in \mathcal{G}} \overline{{}^{(\beta)}\chi(R)} R. \tag{3.2.40} $$

In the following manner, the projection operator, ${}^{(\beta)}P$, extracts only the components of a specific irreducible representation, ${}^{(\beta)}D$.

Theorem 3.3 (Projection onto an Irreducible Representation)**.** *A component of a specific irreducible representation ${}^{(\beta)}D$ extracted by a projection operator ${}^{(\beta)}P$ is*

$$ {}^{(\beta)}Pf = \sum_m {}^{(\beta)}c_m \, {}^{(\beta)}\phi_m, \tag{3.2.41} $$

where function f is expanded by irreducible representations with their bases as $f = \sum_\alpha {}^{(\alpha)}c^m \, {}^{(\alpha)}\phi_m$.

Proof. Operate the projection on the arbitrary vector f to get the lth component of the representation ${}^{(\beta)}D$ as

$$ {}^{(\beta)}P_{l(m)}f = \frac{d_\beta}{g} \sum_{R \in \mathcal{G}} \overline{{}^{(\beta)}D^l_m(R)} R \sum_\alpha \sum_n {}^{(\alpha)}c_n \, {}^{(\alpha)}\phi_n \tag{3.2.42} $$

$$ = \frac{d_\beta}{g} \sum_\alpha \sum_n {}^{(\alpha)}c_n \, {}^{(\alpha)}\phi_{n'} \sum_{R \in \mathcal{G}} \overline{{}^{(\beta)}D^l_m(R)} \, {}^{(\alpha)}D^{n'}_n(R) \tag{3.2.43} $$

$$ = \sum_\alpha \sum_n {}^{(\alpha)}c_n \, {}^{(\alpha)}\phi_{n'} \delta^{\beta\alpha} \delta^{ln'} \delta_{mn} \tag{3.2.44} $$

$$ = {}^{(\beta)}c_m \, {}^{(\beta)}\phi_l. \tag{3.2.45} $$

As a result of taking the sum of l with $l = m$, (3.2.41) is obtained. □

The reduction coefficient of arbitrary reducible representation thus can be calculated by utilizing Theorem 3.3 of the projection operator.

To get the rotation-invariant bases, it is necessary to reduce a given reducible representation (X) to the identity representation (Γ_1); the symbol (Γ_1) is the Bethe's notation.

The projection operator onto the identity representation is

$$^{(\Gamma_1)}P = \frac{1}{g}\sum_{R\in\mathcal{G}} R, \tag{3.2.46}$$

since their characters are 1. Assuming the invariant relation on a basis function

$$^{(\Gamma_1)}P\,^{(\Gamma_1)}\Psi = {}^{(\Gamma_1)}\Psi, \tag{3.2.47}$$

where the basis is expanded by the bases of the given representation X

$$^{(\Gamma_1)}\Psi = \sum_m u_m\,^{(X)}\phi_m. \tag{3.2.48}$$

The expansion coefficients, $\{u_m\}_{m=1}^{d_X}$, are obtained by solving the eigenvalue problem for the matrix expression of the identity representation in the reducible representation X.

Theorem 3.4 (Reduction Coefficient)**.** *The reduction coefficients from a given reducible representation X to identity representation Γ_1 are calculated through the eigenvalue problem for the matrix expression of the projection operator*

$$^{(\Gamma_1|X)}Pu = u, \tag{3.2.49}$$

where

$$^{(\Gamma_1|X)}P = \frac{1}{g}\sum_{R\in\mathcal{G}} {}^{(X)}D\,(R) \tag{3.2.50}$$

is a matrix expression of the projection operator to the identity representation Γ_1 from the reducible representation X.

Proof. Expand the basis $^{(\Gamma_1)}\Psi$ by the bases of the given reducible representation X as

$$^{(\Gamma_1)}\Psi = \sum_m u_m\,^{(X)}\phi_m. \tag{3.2.51}$$

Assume the basis is invariant under the projection onto the identity representation, it gives the followings

$$^{(\Gamma_1)}P\,^{(\Gamma_1)}\Psi = {}^{(\Gamma_1)}P\left(\sum_m u_m\,^{(X)}\phi_m\right) \tag{3.2.52}$$

$$= \sum_m u_m\left(^{(\Gamma_1)}P\,^{(X)}\phi_m\right) \tag{3.2.53}$$

$$= \sum_m u_m\left(\frac{1}{g}\sum_{R\in\mathcal{G}} R\,^{(X)}\phi_m\right) \tag{3.2.54}$$

$$= \sum_m u_m \left(\frac{1}{g} \sum_{R \in \mathcal{G}} {}^{(X)}\phi_{m'} \, {}^{(X)}D_m^{m'}(R) \right) \tag{3.2.55}$$

$$= \sum_m u_m \, {}^{(X)}\phi_{m'} \left(\frac{1}{g} \sum_{R \in \mathcal{G}} {}^{(X)}D_m^{m'}(R) \right). \tag{3.2.56}$$

Rewrite (3.2.51) and (3.2.56) in a matrix form as

$${}^{(X)}\Phi^\top U = {}^{(X)}\Phi^\top \, {}^{(\Gamma_1|X)}PU. \tag{3.2.57}$$

Sweep out the vector of basis functions, $\Phi^\top$, from each term. As a result the eigenvalue problem for ${}^{(\Gamma_1|X)}P$ is derived. □

Example 3.1 (Product of two representations)**.** If the identity representation, Γ_1, is obtained from a reduction of a product representation, $l_1 \times l_2 = \Gamma_1 + \cdots$, the basis of identity representation can be written as

$${}^{(\Gamma_1)}\Psi = \sum_{m_1 m_2} {}^{(l_1)}\psi_{m_1} \, {}^{(l_2)}\psi_{m_2} \, {}^{(l_1 \times l_2|\Gamma_1)}C_1^{m_1 m_2} \tag{3.2.58}$$

$$\stackrel{\text{def}}{=} \sum_{m_1 m_2} {}^{(l_1)}\psi_{m_1} \, {}^{(l_2)}\psi_{m_2} \, {}^{(l_1 \times l_2)}c^{m_1 m_2}, \tag{3.2.59}$$

where ${}^{(l_1 l_2|\Gamma_1)}C_1^{m_1 m_2}$ is a Clebsh-Gordan (CG) coefficient for reducing the product representation into the identity representation. Because the identity representation is one-dimensional one, there is only one basis. Thus it is reasonable to omit the symbol for destination element of the CG coefficient, ${}^{(l_1 l_2|\Gamma_1)}C_1^{m_1 m_2} \to {}^{(l_1 l_2)}c^{m_1 m_2}$. The CG coefficient for the identity representation in (3.2.59) can be calculated through the eigenvalue problem of (3.2.49). Because of the unitarity of the CG coefficient, the identity CG coefficient is also a unitary tensor

$$\overline{{}^{(l_1 \times l_2)}c^{m_1 m_2}} = {}^{(l_1 \times l_2)}c_{m_1 m_2}. \tag{3.2.60}$$

This definition can be generalized straightforward:

Corollary 3.3 (Reduction to identity representation)**.** *The reduction of any product representation into the identity representation is*

$${}^{(\Gamma_1),\sigma}\Psi = \sum_{m_1 \cdots m_p} {}^{(l_1)}\psi_{m_1} \cdots {}^{(l_p)}\psi_{m_p} \, {}^{(l_1 \ldots l_p),\sigma}c^{m_1 \ldots m_p}, \tag{3.2.61}$$

$$\overline{{}^{(l_1 \ldots l_p),\sigma}c^{m_1 \ldots m_p}} = {}^{(l_1 \ldots l_p),\sigma}c_{m_1 \ldots m_p}, \tag{3.2.62}$$

where the direct product symbol, $\times$, is omitted for simplicity. The CG coefficients, ${}^{(l_1 \ldots l_p),\sigma}c^{m_1 \ldots m_p}$, are the answers of the eigenvalue problem (3.2.49)*. There can be more than one eigenvectors whose eigenvalues are unity when $p \geq 4$, and an index σ is added to distinguish these coefficients.*

The structural feature is a coefficients of an identity representation that is obtained from the reduction of order parameters of the product representation,

$$ {}^{(l_1\cdots l_p),\sigma}_{\ \ i}d_n = \sum_{m_1\cdots m_p} {}^{(l_1)}_{i}a_{nm_1}\cdots{}^{(l_p)}_{i}a_{nm_p}\,{}^{(l_1\cdots l_p),\sigma}c_{m_1\cdots m_p}. \tag{3.2.63} $$

It is expected to be invariant to any rotational operation, but it is not trivial and will be proved in the next section.

Invariance of structural feature under ordinal rotational operations

Lemma 3.2 (Transformation of the order parameter)**.** *The rotational operation on the order parameter of atom i follows the transformation law of a set of irreducible representations with which the atomic density function centered on atom i is expanded as,*

$$ R\,{}^{(l)}_{i}a_{nm} = \sum_{m'} {}^{(l)}_{i}a_{nm'}\overline{{}^{(l)}D^{m'}_{m}(R)}. \tag{3.2.64} $$

Proof. By treating a_{nm} as a function in linear space, the following is immediately found;

$$ R\,{}^{(l)}_{i}a_{nm} = \left[R\sum_{j\in{}_i\mathcal{N}}\overline{{}_iP_n\,{}^{(l)}_{i}Y_m}\right]\left({}_j\mathbf{r}\right) \tag{3.2.65} $$

$$ = \left[\sum_{j\in{}_i\mathcal{N}}\overline{(R\,{}_iP_n)}\overline{\left(R\,{}^{(l)}_{i}Y_m\right)}\right]\left({}_j\mathbf{r}\right) \tag{3.2.66} $$

$$ = \sum_{m'}\left[\sum_{j\in{}_i\mathcal{N}}\overline{{}_iP_n}\,\overline{{}^{(l)}_{i}Y_{m'}}\right]\left({}_j\mathbf{r}\right)\overline{{}^{(l)}D^{m'}_{m}(R)} \tag{3.2.67} $$

$$ = \sum_{m'}{}^{(l)}_{i}a_{nm'}\overline{{}^{(l)}D^{m'}_{m}(R)}. \tag{3.2.68} $$

□

Theorem 3.5. *The structural feature is invariant under any ordinal rotational operations,*

$$ R\,{}^{(l_1\cdots l_p),\sigma}_{\ \ i}d_n = {}^{(l_1\cdots l_p),\sigma}_{\ \ i}d_n. \tag{3.2.69} $$

Proof. Apply a rotational operation R on the structural feature,

$$ R\,{}^{(l_1\cdots l_p),\sigma}_{\ \ i}d_n = R\sum_{m_1\cdots m_p}{}^{(l_1)}_{i}a_{nm_1}\cdots{}^{(l_p)}_{i}a_{nm_p}\,{}^{(l_1\cdots l_p),\sigma}c_{m_1\cdots m_p} \tag{3.2.70} $$

$$ = R\sum_{m_1\cdots m_p}{}^{(l_1)}_{i}a_{nm_1}\cdots{}^{(l_p)}_{i}a_{nm_p}\overline{{}^{(l_1\cdots l_p|\Gamma_1),\sigma}C_1^{m_1\cdots m_p}}. \tag{3.2.71} $$

Because the product of the order parameters follow the same transformation rule as the basis of the product representation, each order parameter can be transformed by the rotational operation from Lemma 3.2 as

$$R \sum_{m_1 \cdots m_p} {}^{(l_1)}_{i}a_{nm_1} \cdots {}^{(l_p)}_{i}a_{nm_p} \overline{{}^{(l_1 \cdots l_p|\Gamma_1),\sigma}C_1^{m_1 \cdots m_p}}$$

$$= \sum_{m_1 \cdots m_p} \left(R\, {}^{(l_1)}_{i}a_{nm_1} \right) \cdots \left(R\, {}^{(l_p)}_{i}a_{nm_p} \right) \overline{{}^{(l_1 \cdots l_p|\Gamma_1),\sigma}C_1^{m_1 \cdots m_p}} \tag{3.2.72}$$

$$= \sum_{m_1 \cdots m_p} \left(\sum_{m_{1'}} {}^{(l_1)}_{i}a_{nm_1} \overline{{}^{(l_1)}D^{m_{1'}}_{m_1}(R)} \right) \cdots \left(\sum_{m_{p'}} {}^{(l_p)}_{i}a_{nm_p} \overline{{}^{(l_p)}D^{m_{p'}}_{m_p}(R)} \right) \overline{{}^{(l_1 \cdots l_p|\Gamma_1),\sigma}C_1^{m_1 \cdots m_p}} \tag{3.2.73}$$

$$= \sum_{m_1 \cdots m_p} \sum_{m_{1'} \cdots m_{p'}} {}^{(l_1)}_{i}a_{nm_{1'}} \cdots {}^{(l_p)}_{i}a_{nm_{p'}} \overline{{}^{(l_1)}D^{m_{1'}}_{m_1}(R)} \cdots \overline{{}^{(l_p)}D^{m_{p'}}_{m_p}(R)} \cdot \overline{{}^{(l_1 \cdots l_p|\Gamma_1),\sigma}C_1^{m_1 \cdots m_p}} \tag{3.2.74}$$

$$= \sum_{m_1 \cdots m_p} \sum_{m_{1'} \cdots m_{p'}} {}^{(l_1)}_{i}a_{nm_{1'}} \cdots {}^{(l_p)}_{i}a_{nm_{p'}} \overline{{}^{(l_1 \cdots l_p)}D^{m_{1'} \cdots m_{p'}}_{m_1 \cdots m_p}(R)} \cdot \overline{{}^{(l_1 \cdots l_p|\Gamma_1),\sigma}C_1^{m_1 \cdots m_p}}. \tag{3.2.75}$$

By the definition of the CG coefficient, the representation tensor transforms the coordinate of the CG coefficient from $1 \cdots p$ to $1' \cdots p'$ as

$$\sum_{m_1 \cdots m_p} \sum_{m_{1'} \cdots m_{p'}} {}^{(l_1)}_{i}a_{nm_{1'}} \cdots {}^{(l_p)}_{i}a_{nm_{p'}} \overline{{}^{(l_1 \cdots l_p)}D^{m_{1'} \cdots m_{p'}}_{m_1 \cdots m_p}(R)} \cdot \overline{{}^{(l_1 \cdots l_p|\Gamma_1),\sigma}C_1^{m_1 \cdots m_p}}$$

$$= \sum_{m_{1'} \cdots m_{p'}} {}^{(l_1)}_{i}a_{nm_{1'}} \cdots {}^{(l_p)}_{i}a_{nm_{p'}} \overline{{}^{(l_1 \cdots l_p|\Gamma_1),\sigma}C_1^{m_{1'} \cdots m_{p'}}} \tag{3.2.76}$$

$$= {}^{(l_1 \cdots l_p),\sigma}_{i}d_n. \tag{3.2.77}$$

Therefore the structural factor is invariant for any rotational operations. □

Projection Matrix for $SO(3)$

When using a representation matrix $D(R)$ of the rotation group, which is a unitary matrix corresponding to a rotation action R, the transformation of the basis of the representation can be defined as

$$R\, {}^{(J)}\phi_M = \sum_{M'} {}^{(J)}\phi_{M'}\, {}^{(J)}D^{M'}_{M}(R). \tag{3.2.78}$$

The matrix ${}^{(J)}D(R)$ is the Wigner D-Matrix with integer J for $SO(3)$. Theorem 3.3 gives a projection matrix to the identity representation,

$${}^{(\Gamma_1|X)}P = \frac{1}{g} \sum_{R \in \mathcal{G}} {}^{(X)}D(R) \tag{3.2.79}$$

$$= \frac{1}{8\pi^2} \int_0^{2\pi} \mathrm{d}\alpha \int_0^{\pi} \mathrm{d}\beta \sin\beta \int_0^{2\pi} \mathrm{d}\gamma \, {}^{(X)}D\left(\alpha, \beta, \gamma\right), \tag{3.2.80}$$

where the summation over the group $\mathcal{G}$ is substituted by the integration over the unit sphere since $SO\left(3\right)$ is a continuous group, and ${}^{(X)}D\left(R\right)$ is a representation matrix for a product representation

$$X = l_1 \times l_2 \times \cdots \times l_p. \tag{3.2.81}$$

The product representation is by definition

$$\left[{}^{(X)}D\left(\alpha, \beta, \gamma\right)\right]^{m_1 m_2 \cdots m_p}_{m_{1'} m_{2'} \cdots m_{p'}} = {}^{(l_1)}D^{m_1}_{m_{1'}}\left(\alpha, \beta, \gamma\right) {}^{(l_2)}D^{m_2}_{m_{2'}}\left(\alpha, \beta, \gamma\right) \cdots {}^{(l_p)}D^{m_p}_{m_{p'}}\left(\alpha, \beta, \gamma\right), \tag{3.2.82}$$

giving a projection matrix element

$$\begin{aligned}
{}^{(\Gamma_1 | l_1 \times \cdots \times l_p)}P_{m_1 \cdots m_p \left(m_{1'} \cdots m_{p'}\right)} &= \frac{1}{g} \sum_{R \in \mathcal{G}} {}^{(l_1 \times \cdots \times l_p)}D^{m_1 \cdots m_p}_{m_{1'} \cdots m_{p'}}\left(R\right) && (3.2.83)\\
&= \frac{1}{8\pi^2} \int_0^{2\pi} \mathrm{d}\alpha \int_0^{\pi} \mathrm{d}\beta \sin\beta \int_0^{2\pi} \mathrm{d}\gamma \, {}^{(l_1 \times \cdots \times l_p)}D^{m_1 \cdots m_p}_{m_{1'} \cdots m_{p'}}\left(\alpha, \beta, \gamma\right) && (3.2.84)\\
&= \frac{1}{8\pi^2} \int_0^{2\pi} \mathrm{d}\alpha \int_0^{\pi} \mathrm{d}\beta \sin\beta \int_0^{2\pi} \mathrm{d}\gamma \\
&\quad \times {}^{(l_1)}D^{m_1}_{m_{1'}}\left(\alpha, \beta, \gamma\right) \cdots {}^{(l_p)}D^{m_p}_{m_{p'}}\left(\alpha, \beta, \gamma\right). && (3.2.85)
\end{aligned}$$

By means of this explicit expression of the projection matrix and Theorem 3.4, the generalized Clebsh-Gordan coefficient can be calculated, which is needed for the structural feature in Definition 3.1. For more details, see Seko et al. [23]. Note that the projection matrix for $SO\left(3\right)$ is not invariant for a spatial inversion, azimuth quantum numbers should be restricted to make the structural feature invariant to any ordinal rotation, which is described later in 3.2.2.

Energy

Rotational invariance of Functional

The total energy functional, $\mathcal{E}$, is a map from the total atomic density function, $\rho \in \mathcal{X}_\sharp$, to the potential energy of the whole system, where $\mathcal{X}_\sharp$ refers to a whole set of functions that is expressed as $\sum_i \delta\left(\mathbf{r} - \mathbf{r}_i\right)$ with a periodicity in all dimensions. Note that the density function is a real linear function, but this energy functional $\mathcal{E}$ is not always linear.

Let the true atomic energy functional ${}_i\mathcal{E}$. Since operators in the Banach space, T and S, can make a sum of them defined on a domain of $\mathcal{D}\left(T\right) \cap \mathcal{D}\left(S\right)$, the total energy

functional can be expanded by the atomic energies as

$$\mathcal{E}[\rho] = \left(\sum_{i=1}^{N} {}_i\mathcal{E} \right)[\rho] = \sum_{i=1}^{N} {}_i\mathcal{E}[\rho], \tag{3.2.86}$$

$$\mathcal{D}(\mathcal{E}) = \bigcap_{N} \mathcal{X}_\sharp = \mathcal{X}_\sharp. \tag{3.2.87}$$

Assume that the true atomic energy functional can be estimated by a linear functional ${}_i\mathcal{F}$ on $\mathcal{X}_\sharp$,

$${}_i\mathcal{E}[\rho] \approx {}_i\mathcal{F}[\rho], \quad \rho \in \mathcal{X}_\sharp, {}_i\mathcal{F} \in \mathcal{X}_\sharp^* = B(\mathcal{X}_\sharp, \mathbb{R}), \tag{3.2.88}$$

thus the total energy also can be estimated as

$$\mathcal{E}[\rho] \approx \sum_{i=1}^{N} {}_i\mathcal{F}[\rho]. \tag{3.2.89}$$

Figure 3.2.1 shows the relation between the energy functional and its rotation. When the density function is rotated by an operation R,

$$R : \mathcal{X}_\sharp \ni \rho \mapsto R\rho \in \mathcal{X}_\sharp,$$

the energy functional then can be defined through a relation

$$[R^*\mathcal{E}][\rho] = \mathcal{E}[R\rho] \quad , \forall \rho \in \mathcal{X}_\sharp, \tag{3.2.90}$$

where $R^* : \mathcal{X}_\sharp^* \to \mathcal{X}_\sharp^*$ is an adjoint of the rotation operator $R : \mathcal{X}_\sharp \to \mathcal{X}_\sharp$. The vector space, $\mathcal{X}_\sharp$, and its adjoint space, $\mathcal{X}_\sharp^*$, is connected with the Riesz correspondence, $J_{\mathcal{X}_\sharp}$. The energy functional is required to be invariant under any rotation operation in $O(3)$,

$$[R^*\mathcal{E}][\rho] = \mathcal{E}[\rho], \tag{3.2.91}$$

thereby

$$\mathcal{E}[R\rho] = \mathcal{E}[\rho]. \tag{3.2.92}$$

The assumed linearity of the total energy gives the rotational invariance of each atomic energy functional,

$${}_i\mathcal{E}[R\rho] = {}_i\mathcal{E}[\rho]. \tag{3.2.93}$$

Therefore the invariance of the energy functional under rotation operations can be regarded as the invariance of the atomic energy functional under the operation on the density function.

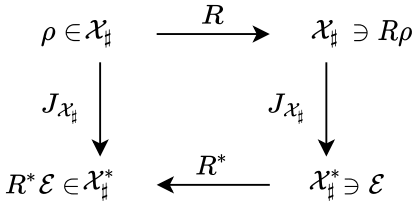


Figure 3.2.1: The relation between the energy functional $\mathcal{E}$ and its rotation $R^* : \mathcal{E} \mapsto R^*\mathcal{E}$. $J_{\mathcal{X}_\sharp}$ is the Riesz correspondence $\mathcal{X}_\sharp \to \mathcal{X}_\sharp^*$ between vector space and its adjoint space.

Rotational invariance of the approximated atomic potential energy

The atomic energy should be invariant under any rotation operation, so should the approximated linear functional ${}_i\mathcal{F}$. The rotational invariance of the approximated energy should be imposed by expanding the energy using the structural features, which are rotationally invariant, that is

$$\begin{aligned} {}_i\mathcal{F}[\rho] &\approx \text{Polynomial of structural features} \\ &\stackrel{\text{def}}{=} F\left(\left\{{}^{(l_1\cdots l_p),\sigma}_{\ \ i}d_n\right\}\right). \end{aligned} \tag{3.2.94}$$

Note that the atom index i is shifted from the approximate energy functional to the structural feature. The types of the polynomial models and the concrete examples are described in detail in [23, 29, 24].

Force

The forces acting on atoms are derived from the derivatives of the total potential energy U with respect to the Cartesian coordinates attached on each atom. Let the Lagrangian $L\left(q^1, \ldots, q^N, v_1, \ldots v_N, t\right) = T - U \in TC \times \mathbb{R}$, where q^αs are labels for the configurations in the manifold C, v_αs are labels for the tangent vectors to C, i.e. TC, and $U \in C^\infty(M)$. The generalized force can be written as

$$F_\alpha = \dot{p_\alpha} = \frac{\partial L}{\partial q^\alpha} = -\frac{\partial U}{\partial q^\alpha}, \tag{3.2.95}$$

where $\alpha = \{1 \ldots 3N\}$ for the N body system. If these indices are grouped by each body, the force on a body i in dimension α can be written as

$${}_iF_\alpha = -\frac{\partial U}{\partial\, {}_iq^\alpha}, \tag{3.2.96}$$

where body number $i = \{1 \ldots N\}$ and dimension $\alpha = \{1, 2, 3\}$. Since the atomic potential energy of atom i, ${}_iE$, is defined as a polynomial of a set of polynomial invariants, ${}_iD =$

${}_iD_{\text{pair}} + {}_iD_2 + \cdots$, the atomic potential energy is written explicitly as

$$
\begin{aligned}
{}_iE\left({}_iD\right) &= \left[f_1 + f_2 + \cdots\right]\left({}_iD\right) && (3.2.97)\\
&= \left[\sum_m f_m\right]\left({}_iD\right) && (3.2.98)\\
&\overset{\text{def}}{=} \mathcal{F}\left({}_iD\right), && (3.2.99)
\end{aligned}
$$

where $\mathcal{F}$ is an aggregated polynomial of f_m.

From (3.2.96) the force acting on body k from bodies in the neighbor is

$${}_kF_\alpha = -\frac{\partial}{\partial\,{}_kq^\alpha}\left(\sum_{i\in{}_k\mathcal{N}} {}_iE\left({}_iD\right)\right), \tag{3.2.100}$$

thus partial derivatives of the atomic potential energy is needed. The atomic potential energy can be expressed as a composite function. First function is a polynomial, $\mathcal{F}:\mathbb{R}^m \ni {}_iD \mapsto E \in \mathbb{R}$, which is a map from a set of polynomial invariants, ${}_iD = \left\{{}_i^{(l_1\times\cdots l_p),\sigma}d_n\right\}$, to an energy of real value. Second function, $\mathcal{D}$, is a map from a set of order parameters, $\left\{{}_i^{(l)}a_{nm}\right\}_{l=l_1}^{l_p}$, to a structural feature, ${}_i^{(l_1\times\cdots l_p),\sigma}d_n$. Third function, $\mathcal{A}$, is a map from a set of coordinates of neighbor atoms, ${}_i\mathcal{N}$, to a set of order parameters, $\left\{{}_i^{(l)}a_{n,-l},\ldots,{}_i^{(l)}a_{n,l}\right\}$.

$$
\begin{aligned}
&\mathcal{F}:\left\{\left\{{}_id_n\right\},\left\{{}_i^{(l\times l)}d_n\right\},\ldots,\left\{{}_i^{(l_1\times\cdots l_p),\sigma}d_n\right\}\right\}\mapsto E && (3.2.101)\\
&\mathcal{D}:\left\{\left\{{}_i^{(l_1)}a_{nm_1}\right\}_{m_1=-l_1}^{l_1},\ldots,\left\{{}_i^{(l_p)}a_{nm_p}\right\}_{m_p=-l_p}^{l_p}\right\}\mapsto {}_i^{(l_1\times\cdots l_p),\sigma}d_n && (3.2.102)\\
&\mathcal{A}:\left(\left\{{}_j\boldsymbol{q}\right\}_{j=1}^{N};i,l,\ldots\right)\mapsto\left\{{}_i^{(l)}a_{n,-l},\ldots,{}_i^{(l)}a_{n,l}\right\} && (3.2.103)
\end{aligned}
$$

where parameters not related to the discussion on the derivative are omitted. Now the energy can be written as a composite function form as

$${}_iE = \mathcal{F}\circ\left(\mathcal{D}\circ\mathcal{A}\right)\left(\left\{{}_j\boldsymbol{q}\right\}_{j=1}^{N};i,l,\ldots\right). \tag{3.2.104}$$

By applying of Chain Rule, the derivative of the atomic potential energy of atom i is written in Leibnitz notation as

$$\frac{\partial\,{}_iE}{\partial\,{}_kq^\alpha} = \frac{\partial\mathcal{F}}{\partial\mathcal{D}}\frac{\partial\mathcal{D}}{\partial\mathcal{A}}\frac{\partial\mathcal{A}}{\partial\,{}_kq^\alpha}. \tag{3.2.105}$$

For example, for the linear model

$$\mathcal{F}: f_1\left(\left\{{}_i^{(\boldsymbol{l}),\sigma}d^n\right\}_{\boldsymbol{l}=l}^{l_1\times\cdots l_p}\right) = \sum_{n\boldsymbol{l}\sigma}{}^{(\boldsymbol{l}),\sigma}w_n\,{}_i^{(\boldsymbol{l}),\sigma}d_n, \tag{3.2.106}$$

let

$$\mathbf{A} := \left\{ \left\{ {}^{(l_1)}_{i}a_{nm_1} \right\}_{m_1=-l_1}^{l_1}, \ldots, \left\{ {}^{(l_p)}_{i}a_{nm_p} \right\}_{m_p=-l_p}^{l_p} \right\}, \tag{3.2.107}$$

thereby

$$\frac{\partial\, {}_iE}{\partial\, {}_kq^\alpha} = \sum_{n\boldsymbol{l}\sigma}{}^{(\boldsymbol{l}),\sigma}w_n \left\langle \frac{\partial\, {}^{(\boldsymbol{l}),\sigma}_{i}d_n}{\partial \mathbf{A}}, \frac{\partial \mathbf{A}}{\partial\, {}_kq^\alpha} \right\rangle \tag{3.2.108}$$

$$= \sum_{n\boldsymbol{l}\sigma} \sum_{L\in\{1\cdots|\boldsymbol{l}|\}} {}^{(\boldsymbol{l}),\sigma}w_n \left({}^{(l_1)}_{i}a_{nm_1} \cdots {}^{(\check{L})}_{i}a_{nm_L} \cdots {}^{(l_p)}_{i}a_{nm_p} \right) \frac{\partial\, {}^{(L)}_{i}a_{nm_L}}{\partial\, {}_kq^\alpha}\, {}^{(l_1\times\cdots l_p),\sigma}c_{m_1\cdots m_p}. \tag{3.2.109}$$

By Summing up these derivatives of atomic energy for the atoms in the neighbor of the atom k, the total force acting on the atom k is derived as

$${}_kF_\alpha = -\frac{\partial}{\partial\, {}_kq^\alpha} \left(\sum_{i=1}^{N} {}_iE\left({}_iD\right) \right). \tag{3.2.110}$$

$$= -\sum_{i\in{}_k\mathcal{N}} \frac{\partial\, {}_iE}{\partial\, {}_kq^\alpha} \tag{3.2.111}$$

$$= -\sum_{i={}_k\mathcal{N}} \left(\sum_{n\boldsymbol{l}\sigma} \sum_{L\in\{1\cdots|\boldsymbol{l}|\}} {}^{(\boldsymbol{l}),\sigma}w_n \left({}^{(l_1)}_{i}a_{nm_1} \cdots {}^{(\check{L})}_{i}a_{nm_L} \cdots {}^{(l_p)}_{i}a_{nm_p} \right) \times \frac{\partial\, {}^{(L)}_{i}a_{nm_L}}{\partial\, {}_kq^\alpha}\, {}^{(l_1\times\cdots l_p),\sigma}c_{m_1\cdots m_p} \right). \tag{3.2.112}$$

The derivatives of the order parameter is given by the well-known derivatives of spherical harmonics.

Pair-wise Force

A generalized force on a direction α of generalized coordinate is

$$F_\alpha = \dot{p_\alpha} = \frac{\partial L}{\partial q^\alpha} = -\frac{\partial U}{\partial q^\alpha}. \tag{3.2.113}$$

When coordinates are grouped by each atom, (3.2.113) is also grouped by each atom as

$${}_iF_\alpha = -\frac{\partial U}{\partial\, {}_iq^\alpha}. \tag{3.2.114}$$

If the potential energy U of a set of N-particles can be approximated by a linear function of a set of ordered particle pairs,

$$U\left(\{1,2,3,\ldots,N\}\right) \approx U_{\mathrm{a}}\left(\{(1,2),\cdots,(N,N-1)\}\right) \qquad \text{approximation}$$

$$
\begin{aligned}
&= U_{\mathrm{a}}\left(s \cdot {}_1\mathcal{N} \cup {}_2\mathcal{N} \cup \cdots \cup {}_N\mathcal{N}\right) && \text{grouping} \\
&= s\left(U_{\mathrm{a}}\left({}_1\mathcal{N}\right) + U_{\mathrm{a}}\left({}_2\mathcal{N}\right) + \cdots + U_{\mathrm{a}}\left({}_N\mathcal{N}\right)\right) && \text{linearity,}
\end{aligned}
$$

where U_{a} is an approximation of a many-body potential by means of a set of unordered particle pairs for all atoms, ${}_i\mathcal{N}$ is a neighbor list centering an atom i, and s is a coefficient of over-count for the neighbor style. The set of pairs around atom i is equivalent to a set of vectors from atom i,

$$
{}_i\mathcal{N} \sim {}_i\mathbf{N} = \left\{{}_jq - {}_iq\right\}_{j \in {}_i\mathcal{N}} \overset{\text{def}}{=} \left\{{}_{ij}q\right\}_{j \in {}_i\mathcal{N}}, \tag{3.2.115}
$$

where ${}_{ij}q$ means a vector from atom i to j.

The force acting on atom i can be calculated as a summation of derivatives of the partial potential energies[1]

$$
\begin{aligned}
{}_iF_\alpha &\approx -\frac{\partial}{\partial\,{}_iq^\alpha} s\left(U_{\mathrm{a}}\left({}_1\mathbf{N}\right) + U_{\mathrm{a}}\left({}_2\mathbf{N}\right) + \cdots + U_{\mathrm{a}}\left({}_N\mathbf{N}\right)\right) && (3.2.116) \\
&= -s\left(\frac{\partial}{\partial\,{}_iq^\alpha} U_{\mathrm{a}}\left({}_1\mathbf{N}\right) + \frac{\partial}{\partial\,{}_iq^\alpha} U_{\mathrm{a}}\left({}_2\mathbf{N}\right) + \cdots + \frac{\partial}{\partial\,{}_iq^\alpha} U_{\mathrm{a}}\left({}_N\mathbf{N}\right)\right) && (3.2.117) \\
&= -s\left(\left\langle \frac{\partial U_{\mathrm{a}}}{\partial\,{}_1\mathbf{N}}, \frac{\partial\,{}_1\mathbf{N}}{\partial\,{}_iq^\alpha} \right\rangle + \left\langle \frac{\partial U_{\mathrm{a}}}{\partial\,{}_2\mathbf{N}}, \frac{\partial\,{}_2\mathbf{N}}{\partial\,{}_iq^\alpha} \right\rangle + \cdots + \left\langle \frac{\partial U_{\mathrm{a}}}{\partial\,{}_N\mathbf{N}}, \frac{\partial\,{}_N\mathbf{N}}{\partial\,{}_iq^\alpha} \right\rangle\right), && (3.2.118)
\end{aligned}
$$

where

$$
\begin{aligned}
\left\langle \frac{\partial U_{\mathrm{a}}}{\partial\,{}_j\mathbf{N}}, \frac{\partial\,{}_j\mathbf{N}}{\partial\,{}_iq^\alpha} \right\rangle &= \sum_{k \in {}_j\mathcal{N}} \sum_{\beta=1,2,3} \left.\frac{\partial U_{\mathrm{a}}}{\partial\,{}_{jk}q^\beta}\right|_{{}_j\mathbf{N}} \left.\frac{\partial\,{}_{jk}q^\beta}{\partial\,{}_iq^\alpha}\right|_{{}_iq^\alpha} && (3.2.119) \\
&= \sum_{k \in {}_j\mathcal{N}} \sum_{\beta=1,2,3} \left.\frac{\partial U_{\mathrm{a}}}{\partial\,{}_{jk}q^\beta}\right|_{{}_j\mathbf{N}} \left.\frac{\partial\left({}_kq - {}_jq\right)^\beta}{\partial\,{}_iq^\alpha}\right|_{{}_kq^\alpha} && (3.2.120) \\
&= \sum_{k \in {}_j\mathcal{N}} \sum_{\beta=1,2,3} \left.\frac{\partial U_{\mathrm{a}}}{\partial\,{}_{jk}q^\beta}\right|_{{}_j\mathbf{N}} \left(\delta_{ki}\delta_\alpha^\beta - \delta_{ji}\delta_\alpha^\beta\right) && (3.2.121) \\
&= \sum_{k \in {}_j\mathcal{N}} \left.\frac{\partial U_{\mathrm{a}}}{\partial\,{}_{jk}q^\alpha}\right|_{{}_j\mathbf{N}} \left(\delta_{ki} - \delta_{ji}\right). && (3.2.122)
\end{aligned}
$$

[1]Here we used the chain rule of vector function:

$$
\begin{aligned}
\frac{\mathrm{d}}{\mathrm{d}x} f\left(g\left(x\right), h\left(x\right)\right) &= \lim_{t \to 0} \frac{f\left(g\left(x+t\right), h\left(x+t\right)\right) - f\left(g\left(x\right), h\left(x\right)\right)}{t} \\
&= \lim_{t \to 0} \frac{f\left(g\left(x+t\right), h\left(x+t\right)\right) - f\left(g\left(x\right), h\left(x+t\right)\right)}{t} \\
&\quad + \lim_{t \to 0} \frac{f\left(g\left(x\right), h\left(x+t\right)\right) - f\left(g\left(x\right), h\left(x\right)\right)}{t} \\
&= \left.\frac{\mathrm{d}f}{\mathrm{d}g}\right|_{g(x),h(x)} \left.\frac{\mathrm{d}g}{\mathrm{d}x}\right|_x + \left.\frac{\mathrm{d}f}{\mathrm{d}h}\right|_{g(x),h(x)} \left.\frac{\mathrm{d}h}{\mathrm{d}x}\right|_x
\end{aligned}
$$

Consequently

$$ {}_iF_\alpha = -s \sum_{j=1}^{N} \sum_{k \in {}_j\mathcal{N}} \left. \frac{\partial U_\mathrm{a}}{\partial\, {}_{jk}q^\alpha} \right|_{{}_j\mathbf{N}} (\delta_{ki} - \delta_{ji}) \tag{3.2.123} $$

$$ = -s \sum_{j=1}^{N} \left(\left. \frac{\partial U_\mathrm{a}}{\partial\, {}_{ji}q^\alpha} \right|_{{}_j\mathbf{N}} \delta_{i \in {}_j\mathcal{N}} - \sum_{k \in {}_j\mathcal{N}} \left. \frac{\partial U_\mathrm{a}}{\partial\, {}_{jk}q^\alpha} \right|_{{}_j\mathbf{N}} \delta_{ji} \right) \tag{3.2.124} $$

$$ = -s \left(\sum_{j \in \left\{ j \,\middle|\, i \in {}_j\mathcal{N} \right\}} \left. \frac{\partial U_\mathrm{a}}{\partial\, {}_{ji}q^\alpha} \right|_{{}_j\mathbf{N}} - \sum_{j \in {}_i\mathcal{N}} \left. \frac{\partial U_\mathrm{a}}{\partial\, {}_{ij}q^\alpha} \right|_{{}_i\mathbf{N}} \right). \tag{3.2.125} $$

The first and second term in (3.2.125) refer to the reaction acting on atom i from atoms in the neighbor and the action from the neighbor of the atom i, respectively. In the last line the index of neighbor vector list is changed from ${}_j\mathbf{N}$ to ${}_i\mathbf{N}$ using the Kronecker delta with the summation on j.

For the *full* neighbor style, a set of atoms whose neighbor list contains an atom i is identical to the neighbor list of the atom i, i.e. $\left\{ j \,\middle|\, i \in {}_j\mathcal{N} \right\} = {}_i\mathcal{N}$. It is because the atom j will be in the neighbor list of the atom i when an atom i is in the neighbor list of an atom j, simultaneously. The coefficient of overcount s is $1/2$ for the full neighbor style, so

$$ {}_iF_\alpha = -\frac{1}{2} \sum_{j \in {}_i\mathcal{N}} \left(\left. \frac{\partial U_\mathrm{a}}{\partial\, {}_{ji}q^\alpha} \right|_{{}_j\mathbf{N}} - \left. \frac{\partial U_\mathrm{a}}{\partial\, {}_{ij}q^\alpha} \right|_{{}_i\mathbf{N}} \right). \tag{3.2.126} $$

For the *half* neighbor style, there is no common atom in the neighbor list of an atom i and a set of atoms whose neighbor list contains the atom i, i.e. ${}_i\mathcal{N} \cap \left\{ j \,\middle|\, i \in {}_j\mathcal{N} \right\} = \varnothing$. In other words, the union of the neighbor list and the set of atoms corresponds to the complete components of a force acting on the atom i. The coefficient of overcount s is 1 for the half neighbor style, thus

$$ {}_iF_\alpha = -\sum_{j \in \left\{ j \,\middle|\, i \in {}_j\mathcal{N} \right\}} \left. \frac{\partial U_\mathrm{a}}{\partial\, {}_{ji}q^\alpha} \right|_{{}_j\mathbf{N}} + \sum_{j \in {}_i\mathcal{N}} \left. \frac{\partial U_\mathrm{a}}{\partial\, {}_{ij}q^\alpha} \right|_{{}_i\mathbf{N}}. \tag{3.2.127} $$

At last the derivative of the atomic energy by the relative position vectors gives a pairwise force ${}_{ij}F_\alpha$ acting on atom i from atom j as

$$ {}_{ij}F_\alpha = \left(- \left. \frac{\partial U_\mathrm{a}}{\partial\, {}_{ji}q^\alpha} \right|_{{}_j\mathbf{N}} + \left. \frac{\partial U_\mathrm{a}}{\partial\, {}_{ij}q^\alpha} \right|_{{}_i\mathbf{N}} \right) \delta_{j \in {}_i\mathcal{N}}, \tag{3.2.128} $$

for the full neighbor style and

$$ {}_{ij}F_\alpha = -\left.\frac{\partial U_\mathrm{a}}{\partial\, {}_{ji}q^\alpha}\right|_{{}_j\mathbf{N}} \delta_{j\in\{j\,|\,i\in{}_j\mathcal{N}\}} + \left.\frac{\partial U_\mathrm{a}}{\partial\, {}_{ij}q^\alpha}\right|_{{}_i\mathbf{N}} \delta_{j\in{}_i\mathcal{N}}, \tag{3.2.129} $$

for the half neighbor style. A summation of the pairwise forces over atoms other than the atom i gives a force acting on the atom i

$$ {}_iF_\alpha = s\sum_j {}_{ij}F_\alpha. \tag{3.2.130} $$

Note that Kronecker delta in (3.2.128) and (3.2.129) is sometimes omitted when the neighbor list of the relative vectors is stored somewhere and used as the index for the sum.

Linearity of parameters

Order parameter

The order parameter of an atom i is the Fourier expansion of the atomic density around the atom. Since the Fourier expansion is linear, so is the order parameter.

Theorem 3.6. *The order parameter of an atom i is linear for its neighbor list,*

$$ {}^{(l)}_i a_{nm}\left[{}_i\mathcal{N}\right] = {}^{(l)}_i a_{nm}\left[{}_i\mathcal{N}\setminus X\right] + {}^{(l)}_i a_{nm}\left[X\right]. \tag{3.2.131} $$

Proof. Dividing the neighbor list into two parts, getting

$$ \begin{aligned} {}^{(l)}_i a_{nm}\left[{}_i\mathcal{N}\right] &= {}^{(l)}_i a_{nm}\left[{}_i\mathcal{N}\setminus X + X\right] && (3.2.132)\\ &= \sum_{j\in({}_i\mathcal{N}\setminus X)+X} {}_iP_n\left({}_jr\right)\overline{{}^{(l)}_iY_m\left({}_j\theta,{}_j\phi\right)} && (3.2.133)\\ &= \left[\sum_{j\in{}_i\mathcal{N}\setminus X} + \sum_{j\in X}\right] {}_iP_n\left({}_jr\right)\overline{{}^{(l)}_iY_m\left({}_j\theta,{}_j\phi\right)} && (3.2.134)\\ &= {}^{(l)}_i a_{nm}\left[{}_i\mathcal{N}\setminus X\right] + {}^{(l)}_i a_{nm}\left[X\right]. && (3.2.135) \end{aligned} $$

□

Structural feature

The structural feature around an atom i is by Definition 3.1

$$ {}^{(l_1\ldots l_p),\sigma}_i d_n = \sum_{m_1\cdots m_p} {}^{(l_1)}_i a_{nm_1}\cdots{}^{(l_p)}_i a_{nm_p}\,{}^{(l_1\ldots l_p),\sigma}c_{m_1\ldots m_p}. \tag{3.2.136} $$

Lemma 3.3. *For the simplest case, $p = 1$, the structural feature is linear,*

$$ {}^{(l_1)}_{i}d_n\left[{}_i\mathcal{N}\right] = {}^{(l_1)}_{i}d_n\left[{}_i\mathcal{N}\backslash X\right] + {}^{(l_1)}_{i}d_n\left[X\right]. \tag{3.2.137} $$

Proof. (3.2.131) gives □

$$ \begin{aligned} {}^{(l_1)}_{i}d_n\left[{}_i\mathcal{N}\right] &= \sum_{m_1} {}^{(l_1)}_{i}a_{nm_1}\left[{}_i\mathcal{N}\right] && (3.2.138)\\ &= \sum_{m_1} {}^{(l_1)}_{i}a_{nm_1}\left[{}_i\mathcal{N}\backslash X + X\right] && (3.2.139)\\ &= \sum_{m_1} {}^{(l_1)}_{i}a_{nm_1}\left[{}_i\mathcal{N}\backslash X\right] + \sum_{m_1} {}^{(l_1)}_{i}a_{nm_1}\left[X\right] && (3.2.140)\\ &= {}^{(l_1)}_{i}d_n\left[{}_i\mathcal{N}\backslash X\right] + {}^{(l_1)}_{i}d_n\left[X\right]. && (3.2.141) \end{aligned} $$

Lemma 3.4. *For $p = 2$, the structural feature is not linear.*

Proof. By definition,

$$ {}^{(l_1l_2)}_{i}d_n = \sum_{m_1m_2} {}^{(l_1)}_{i}a_{nm_1}\,{}^{(l_2)}_{i}a_{nm_2}\,{}^{(l_1l_2)}c_{m_1m_2} \tag{3.2.142} $$

and explicitly expanding it gives

$$ \begin{aligned} {}^{(l_1l_2)}_{i}d_n\left[{}_i\mathcal{N}\right] &= \sum_{m_1m_2} {}^{(l_1)}_{i}a_{nm_1}\left[{}_i\mathcal{N}\right]{}^{(l_2)}_{i}a_{nm_2}\left[{}_i\mathcal{N}\right]{}^{(l_1l_2)}c_{m_1m_2} && (3.2.143)\\ &= \sum_{m_1m_2} {}^{(l_1)}_{i}a_{nm_1}\left[{}_i\mathcal{N}\setminus X + X\right]{}^{(l_2)}_{i}a_{nm_2}\left[{}_i\mathcal{N}\setminus X + X\right]{}^{(l_1l_2)}c_{m_1m_2} && (3.2.144)\\ &= \sum_{m_1m_2} \Big({}^{(l_1)}_{i}a_{nm_1}\left[{}_i\mathcal{N}\setminus X\right]{}^{(l_2)}_{i}a_{nm_2}\left[{}_i\mathcal{N}\setminus X\right] + {}^{(l_1)}_{i}a_{nm_1}\left[{}_i\mathcal{N}\setminus X\right]{}^{(l_2)}_{i}a_{nm_2}\left[X\right]\\ &\quad + {}^{(l_1)}_{i}a_{nm_1}\left[X\right]{}^{(l_2)}_{i}a_{nm_2}\left[{}_i\mathcal{N}\setminus X\right] + {}^{(l_1)}_{i}a_{nm_1}\left[X\right]{}^{(l_2)}_{i}a_{nm_2}\left[X\right]\Big){}^{(l_1l_2)}c_{m_1m_2} && (3.2.145)\\ &= {}^{(l_1l_2)}_{i}d_n\left[{}_i\mathcal{N}\setminus X\right] + {}^{(l_1l_2)}_{i}d_n\left[X\right]\\ &\quad + \sum_{m_1m_2} \Big({}^{(l_1)}_{i}a_{nm_1}\left[{}_i\mathcal{N}\setminus X\right]{}^{(l_2)}_{i}a_{nm_2}\left[X\right]\\ &\qquad + {}^{(l_1)}_{i}a_{nm_1}\left[X\right]{}^{(l_2)}_{i}a_{nm_2}\left[{}_i\mathcal{N}\setminus X\right]\Big){}^{(l_1l_2)}c_{m_1m_2} && (3.2.146)\\ &\neq {}^{(l_1l_2)}_{i}d_n\left[{}_i\mathcal{N}\setminus X\right] + {}^{(l_1l_2)}_{i}d_n\left[X\right]. && (3.2.147) \end{aligned} $$

The inequality in the last line shows the nonlinearity of the structural feature. □

Lemma 3.5. *Structural features of the higher order can be obtained by combining a structural feature of the lower order and a set of order parameters sequentially, i.e. there*

exists a map f such that

$$f : \left\{ {}^{(l_1 \cdots l_{p-1})}_{i}d_n, \left\{ {}^{(l_p)}_{i}a_{nm} \right\} \right\} \mapsto {}^{(l_1 \cdots l_p)}_{i}d. \tag{3.2.148}$$

Proof. Let the structural feature of pth order explicitly indexed with the destination of the reduction

$${}^{(l_1 \cdots l_p|\Gamma_1)}_{i}d_n = \sum_{m_1 \cdots m_p} {}^{(l_1)}_{i}a_{nm_1} \cdots {}^{(l_p)}_{i}a_{nm_p} \overline{{}^{(l_1 \cdots l_p|\Gamma_1),\sigma}C_1^{m_1 \cdots m_p}}. \tag{3.2.149}$$

For $p = 2$ by definition

$${}^{(l_1 l_2|\Gamma_1)}_{i}d_n = \sum_{m_1 m_2} {}^{(l_1)}_{i}a_{nm_1} {}^{(l_2)}_{i}a_{nm_2} \overline{{}^{(l_1 l_2|\Gamma_1)}C_1^{m_1 m_2}}, \tag{3.2.150}$$

then the structural feature of 3$^{\text{rd}}$ order can be calculated as

$$\begin{aligned}
{}^{(l_1 l_2 l_3|\Gamma_1)}_{i}d_n &= \sum_{m_3} {}^{(l_1 l_2|\Gamma_1)}_{i}d_n \, {}^{(l_3)}_{i}a_{nm_3} \overline{{}^{(\Gamma_1 l_3|\Gamma_1)}C_1^{1m_3}} && (3.2.151)\\
&= \sum_{m_3} \left(\sum_{m_1 m_2} {}^{(l_1)}_{i}a_{nm_1} {}^{(l_2)}_{i}a_{nm_2} \overline{{}^{(l_1 l_2|\Gamma_1)}C_1^{m_1 m_2}} \right) {}^{(l_3)}_{i}a_{nm_3} \overline{{}^{(\Gamma_1 l_3|\Gamma_1)}C_1^{1m_3}} && (3.2.152)\\
&= \sum_{m_1 m_2 m_3} {}^{(l_1)}_{i}a_{nm_1} {}^{(l_2)}_{i}a_{nm_2} {}^{(l_3)}_{i}a_{nm_3} \left(\overline{{}^{(l_1 l_2|\Gamma_1)}C_1^{m_1 m_2}} \cdot \overline{{}^{(\Gamma_1 l_3|\Gamma_1)}C_1^{1m_3}} \right) && (3.2.153)\\
&= \sum_{m_1 m_2 m_3} {}^{(l_1)}_{i}a_{nm_1} {}^{(l_2)}_{i}a_{nm_2} {}^{(l_3)}_{i}a_{nm_3} \overline{{}^{(l_1 l_2 l_3|\Gamma_1)}C_1^{m_1 m_2 m_3}}. && (3.2.154)
\end{aligned}$$

Assume that the synthesis rule holds up to the $p-1$th order. Then the structural feature of the pth order can be calculated as

$$\begin{aligned}
{}^{(l_1 \cdots l_p|\Gamma_1)}_{i}d_n &= \sum_{m_p} {}^{(l_1 \cdots l_{p-1}|\Gamma_1)}_{i}d_n \, {}^{(l_p)}_{i}a_{nm_p} \overline{{}^{(\Gamma_1 l_p|\Gamma_1),\sigma}C_1^{1m_p}} && (3.2.155)\\
&= \sum_{m_p} \left(\sum_{m_1 \cdots m_{p-1}} {}^{(l_1)}_{i}a_{nm_1} \cdots {}^{(l_{p-1})}_{i}a_{nm_{p-1}} \overline{{}^{(l_1 \cdots l_{p-1}|\Gamma_1),\sigma}C_1^{m_1 \cdots m_{p-1}}} \right)\\
&\qquad \times {}^{(l_p)}_{i}a_{nm_p} \overline{{}^{(\Gamma_1 l_p|\Gamma_1),\sigma}C_1^{1m_p}} && (3.2.156)\\
&= \sum_{m_1 \cdots m_p} {}^{(l_1)}_{i}a_{nm_1} \cdots {}^{(l_p)}_{i}a_{nm_p} \left(\overline{{}^{(l_1 \cdots l_{p-1}|\Gamma_1),\sigma}C_1^{m_1 \cdots m_{p-1}}} \cdot \overline{{}^{(\Gamma_1 l_p|\Gamma_1),\sigma}C_1^{1m_p}} \right) && (3.2.157)\\
&= \sum_{m_1 \cdots m_p} {}^{(l_1)}_{i}a_{nm_1} \cdots {}^{(l_p)}_{i}a_{nm_p} \overline{{}^{(l_1 \cdots l_p|\Gamma_1),\sigma}C_1^{m_1 \cdots m_p}}, && (3.2.158)
\end{aligned}$$

therefore the proposition is inductively proved. □

Theorem 3.7. *Structural features of the second or higher order are not linear.*

Proof. The nonlinearity of the structural feature for $p \geq 2$ is inductively derived from Lemma

3.4 and Lemma 3.5. □

Note that the linearity of the polynomial models, the atomic energy, and the generalized force follow the linearity of the structural features.

As a result, it is found that only order parameters and structural features of 1st order can be calculated in parallel concerning a set of atoms. However, when atoms are spatially grouped based on the cutoff length of interatomic potential, any structural features restore the linearity for the groups of sets, so do energies, forces, and stresses. It leads to the basic concept of spatial decomposition discussed in 3.2.3.

3.2.2 Algorithmic Aspect of Machine Learning Potential

In this section, the algorithmic aspect of PolyMLP is explained to show how it is parallelized. Loops for inner variables in pseudocode can be accelerated by parallel calculation on each node. On the other hand, loops for atom indices can be accelerated by spatial decomposition over nodes. These two types of acceleration are combined to calculate large systems efficiently.

Definitions of Variables

In this section, the typical notation is used for atom indices and element type.

Atomic energy

The total energy can be expressed as the sum of the atomic energies of each atom,

$$E\left[\rho\right] = \left[\sum_{i=1}^{N} E^{(i)}\right]\left[\rho\right]. \tag{3.2.159}$$

Expand atomic density using basis functions with order parameter

The neighboring atomic density of elements around atom-i is

$$\rho^{(i,s)}\left(\mathbf{r}\right). \tag{3.2.160}$$

This density function can be expanded by the basis functions

$$\rho^{(i,s)}\left(\mathbf{r}\right) = \sum_{n=0}^{n_{\max}} \sum_{l=0}^{l_{\max}} \sum_{m=-l}^{l} a_{nlm}^{(i,s)} f_n\left(r\right) Y_l^m\left(\hat{\mathbf{r}}\right), \tag{3.2.161}$$

where $a_{nlm}^{(i,s)}$ is a generalized order parameter for the atom-i with the features, nlm and s, f_n is a Gaussian-type radial function combined with a cosine-based cutoff function f_c,

and Y_l^m is a spherical harmonic function. The order parameter is a inner product between the basis function and the neighboring atomic density. The basis function is

$$\phi_{nlm}(\mathbf{r}) = f_n(r) Y_l^m(\hat{\mathbf{r}}). \tag{3.2.162}$$

The radial function is

$$f_n(r) \stackrel{\text{def}}{=} \exp\left(-\beta_n (r - r_n)^2\right) f_c(r), \tag{3.2.163}$$

$$f_c(r) \stackrel{\text{def}}{=} \begin{cases} \frac{1}{2}\left(1 + \cos\left(\pi \frac{r}{r_c}\right)\right) & (r < r_c) \\ 0 & (r \geq r_c) \end{cases}, \tag{3.2.164}$$

$$= \chi_{\mathcal{B}_{(0,r_c)}}(r) \frac{1}{2}\left(1 + \cos\left(\pi \frac{r}{r_c}\right)\right) \tag{3.2.165}$$

where $\chi_{\mathcal{B}_{(0,r_c)}}$ is a characteristic function of an open-ball subset $\mathcal{B}_{(0,r_c)}$ of the space. To simplify the expression, we define the generalized order parameter as follows:

$$a_{nlm,\{s,s'\}}^{(i)} \stackrel{\text{def}}{=} \begin{cases} a_{nlm}^{(i,s'')} & \text{if } \{s_i, s''\} = \{s, s'\} \\ 0 & \text{otherwise} \end{cases}. \tag{3.2.166}$$

This means, for example, if the element of atom-i is A and the element of the other atom is B, then

$$a_{nlm}^{(i,B)} = a_{nlm,\{A,B\}}^{(i)} = a_{nlm,\{B,A\}}^{(i)}. \tag{3.2.167}$$

In practice, the order parameter is calculated as a sum in a neighbor list of given cutoff radius r_c

$$a_{nlm}^{(i,s)} \stackrel{\text{def}}{=} \sum_{j \in \mathcal{N}^{(i,s)}} f_n\left(r^{(ij)}\right) \overline{Y_l^m(\hat{\mathbf{r}}^{(ij)})}, \tag{3.2.168}$$

$$\mathcal{N}^{(i,s)} \stackrel{\text{def}}{=} \left\{ j \,\middle|\, r^{(ij)} < r_c, s^{(j)} = s, j \neq i \right\}, \tag{3.2.169}$$

$$\mathbf{r}^{(ij)} \stackrel{\text{def}}{=} \mathbf{r}^{(j)} - \mathbf{r}^{(i)}. \tag{3.2.170}$$

We denote the set of a multiset pair elements, which allows for multiple instances for each of its elements, as

$$\mathcal{S} \stackrel{\text{def}}{=} \{[A, A], [A, B], [B, B], \cdots\}. \tag{3.2.171}$$

Structural Feature

Consider an indexing set for a structural feature that is invariant to the $O(3)$ action with qth-order polynomial of $\left\{a_{nlm,t}^{(i)}\right\}$,

$$\mathcal{L}_p \stackrel{\text{def}}{=} \left\{ \left([l_1, \cdots, l_p], \sigma \right) \middle| l_k \in \{0, \cdots, l^p_{\max}\}, \mathbf{C}^{(l_1 \cdots l_p), \sigma} \neq \mathbf{0}, \sum_{\nu=1}^{p} l_\nu = \text{even} \right\}. \quad (3.2.172)$$

The l_ks are indices of the irreducible representations for each order parameter. The Clebsch-Gordan coefficient should not be zero. The final condition is imposed so that the structural feature is invariant with respect to spatial inversion. A family of these sets indexed by an indexing set $P = \{1, 2, \cdots, p_{\max}\}$ is

$$\{\mathcal{L}_p\}_{p \in P}. \quad (3.2.173)$$

Each element of the union of these family

$$\mathcal{L} = \bigcup_p \mathcal{L}_p \quad (3.2.174)$$

has a one-to-one correspondence with the angular momentum term of the entire polynomial.

An indexing set for types of the atoms is defined as follows,

$$\mathcal{T}_p \stackrel{\text{def}}{=} \{[t_1, \cdots, t_p] \,|\, t_k \in \mathcal{S}, t_1 \cap \cdots \cap t_p \neq \varnothing\}. \quad (3.2.175)$$

The intersection of the type pairs of all orders cannot be an empty set to prevent the structural feature from becoming zero.

The indexing set for types, $\mathcal{T}_p$, can be combined with the $\mathcal{L}_p$ as a direct product of these indexing sets,

$$\mathcal{K}_p \stackrel{\text{def}}{=} \mathcal{L}_p \times \mathcal{T}_p \quad (3.2.176)$$

$$= \{([l_1, \cdots, l_p], [t_1, \cdots, t_p], \sigma) \,|\, ([l_1, \cdots, l_p], \sigma) \in \mathcal{L}_p, [t_1, \cdots, t_p] \in \mathcal{T}_p\}. \quad (3.2.177)$$

A family of these sets indexed by the indexing set P is

$$\{\mathcal{K}_p\}_{p \in P}. \quad (3.2.178)$$

The union of these families,

$$\mathcal{K} = \bigcup_p \mathcal{K}_p, \quad (3.2.179)$$

is an indexing set of all possible polynomials with angular-element terms.

The structural features are composed of the product of the order parameters

$$d_{n\mathbf{k}}^{(i)} = \sum_{m_1=-l_1}^{l_1} \cdots \sum_{m_p=-l_p}^{l_p} C_{m_1,\cdots,m_p}^{\mathbf{l}} a_{nl_1m_1,t_1}^{(i)} \cdots a_{nl_pm_p,t_p}^{(i)}, \tag{3.2.180}$$

where the index is an element of the indexing set for the pth-order angular-element structural feature $\mathcal{K}_p$,

$$\mathbf{k} = \left([l_1, \cdots, l_p], [t_1, \cdots, t_p], \sigma\right) \in \mathcal{K}_p. \tag{3.2.181}$$

Note that the condition mentioned above for the $\mathcal{L}_p$, $\sum_{\nu=1}^{p} l_\nu =$ even, can be obtained from the invariance of the structural feature, $Id_{n\mathbf{k}}^{(i)} = d_{n\mathbf{k}}^{(i)}$. From the conversion rule of the order parameter, we have $Ia_{nlm,t}^{(i)} = (-)^l a_{nlm,t}^{(i)}$ and requiring $(-)^{\sum_{\nu=1}^{p} l_\nu} = 1$ gives $\sum_{\nu=1}^{p} l_\nu =$ even. Since the Wigner functions used for obtaining the CG coefficients are the irreducible representations for the $SO(3)$ group, the projection consisted of the Wigner function doesn't reflect the spatial inversion. Thus we must add this constraint for the structural feature to be invariant for any group action of the $O(3)$ group.

As shown above, a structural feature for an atom-i can be uniquely specified by an index set of radial hyperparameter n and a polynomial index $\mathbf{k}$. Therefore, the tuple $(n, \mathbf{k})$ are referred to as a *feature index*.

Polynomial Feature

The atomic energy functional is approximated by several polynomials for the structural features. Consider an indexing set, $\mathcal{P}$, that specifies the combination of polynomials. A polynomial consisting of $q_{\max}$ linear independent terms can be specified as a union of $q_{\max}$ families of indexing sets, $\{\mathcal{P}_q\}_{q\in Q}$, where the polynomial indexing set is $Q = \{1, 2, \cdots, q_{\max}\}$,

$$\mathcal{P} = \bigcup_{q\in Q} \mathcal{P}_q. \tag{3.2.182}$$

Each $\mathcal{P}_q$ refers to an indexing set for a qth-order polynomial feature from structural features

$$d_{[(n_1,\mathbf{k}_1),\cdots,(n_q,\mathbf{k}_q)]}^{(i)} \stackrel{\text{def}}{=} \prod_{\nu=1}^{q} d_{n_\nu\mathbf{k}_\nu}^{(i)}, \tag{3.2.183}$$

where

$$[(n_1, \mathbf{k}_1), \cdots, (n_q, \mathbf{k}_q)] \in \mathcal{P}_q, \tag{3.2.184}$$

$$\mathcal{P}_q \stackrel{\text{def}}{=} \left\{ \left[(n_1, \mathbf{k}_1), \cdots, (n_q, \mathbf{k}_q) \right] \middle| \forall \nu \in Q, \mathbf{k}_\nu \in \mathcal{K}; \bigcap_{\nu \in Q} e\left(\mathbf{k}_\nu\right) \neq \varnothing \right\}, \tag{3.2.185}$$

$$e\left(\mathbf{k}_p\right) \stackrel{\text{def}}{=} t_1 \cap \cdots \cap t_p. \tag{3.2.186}$$

A family of sets of structural features indexed by the indexing set $\mathcal{P}_q$ is

$$\Delta_q \stackrel{\text{def}}{=} \left\{ d^{(i)}_{[(n_1,\mathbf{k}_1),\cdots,(n_q,\mathbf{k}_q)]} \right\}_{[(n_1,\mathbf{k}_1),\cdots,(n_q,\mathbf{k}_q)]\in\mathcal{P}_q}, \tag{3.2.187}$$

and its union

$$\Delta \stackrel{\text{def}}{=} \bigcup_{q \in Q} \Delta_q \tag{3.2.188}$$

can be used as a complete set for describing an energy polynomial. The union $\mathcal{P}$ and each $\mathcal{P}_q$ refer to the whole polynomial and each polynomial term, respectively.

Potential Energy Calculation

The atomic energy of atom-i is assumed to be approximated by a certain function of structural features

$$E^{(i)}\left[\rho\right] = F\left(\Delta\right) \tag{3.2.189}$$

$$\stackrel{\text{def}}{=} \sum_{q=1}^{q_{\max}} \sum_{\mathbf{f} \in \mathcal{P}_q} w_{\mathbf{f}} d^{(i)}_{\mathbf{f}}. \tag{3.2.190}$$

This equation is shown in the pseudo code 1.

Algorithm 1 Compute energy E

Require: $\{d^{(i)}_{n,\mathbf{k}}\}_{i\in\{1,2,\cdots,N\};\ n\in\{0,1,\cdots n_{\max}\};\ \mathbf{k}\in\mathcal{K}}$ ▷ Get structural features
for $i \in \{1, 2, \cdots, N\}$ **do** ▷ For all atoms
  $E_i \leftarrow 0$ ▷ Initialize atomic potential
  for $\mathbf{f} = [(n_1, \mathbf{k}_1), \ldots, (n_p, \mathbf{k}_p)] \in \mathcal{P}$ **do** ▷ For all feature indices
    $E_i \leftarrow E_i + w_{\mathbf{f}} d^{(i)}_{\mathbf{f}}$ ▷ Update atomic potential (3.2.183),(3.2.190)
  end for
end for
$E \leftarrow 0 \ (\forall i)$ ▷ Initialize total potential energy
for $i \in \{1, 2, \cdots, N\}$ **do** ▷ For all atoms
  $E \leftarrow E + E_i$ ▷ Accumulate atomic energies (3.2.159)
end for
return E

Force and Stress Tensor Calculation

A force acting on an atom-i is a partial derivative of the total potential energy with respect to the position of the atom-i,

$$\mathbf{F}^{(i)} \stackrel{\text{def}}{=} -\frac{\partial E}{\partial \mathbf{r}^{(i)}} \tag{3.2.191}$$

$$= -\frac{\partial}{\partial \mathbf{r}^{(i)}} \sum_{j \in \mathcal{N}_i} E^{(j)} \tag{3.2.192}$$

$$= -\sum_{j \in \mathcal{N}_i} \frac{\partial E^{(j)}}{\partial \mathbf{r}^{(i)}}. \tag{3.2.193}$$

This generalized force on the atom-i is obtained as the sum of the pairwise forces acting from atoms included in the neighbor list of the atom-i,

$$\mathbf{F}^{(i)} = \sum_{j \in \mathcal{N}_i} \mathbf{F}^{(ij)}, \tag{3.2.194}$$

where the pairwise force is

$$\mathbf{F}^{(ij)} \stackrel{\text{def}}{=} \frac{\partial E^{(i)}}{\partial \mathbf{r}^{(ij)}} - \frac{\partial E^{(j)}}{\partial \mathbf{r}^{(ji)}}. \tag{3.2.195}$$

The pairwise force holds the Newton's third law,

$$\mathbf{F}^{(ij)} = -\mathbf{F}^{(ji)}. \tag{3.2.196}$$

Partial Derivative of Atomic Energy

The above pairwise force (3.2.195) contains partial differentiation of atomic energy, which is explicitly expressed as follows using the partial differentiation of the structural feature.

$$\frac{\partial E^{(i)}}{\partial \mathbf{r}^{(ij)}} = \frac{\partial}{\partial \mathbf{r}^{(ij)}} \sum_{q=1}^{q_{\max}} \sum_{\mathbf{f} \in \mathcal{P}_q} w_{\mathbf{f}} d_{\mathbf{f}}^{(i)} \tag{3.2.197}$$

$$= \sum_{q=1}^{q_{\max}} \sum_{\mathbf{f} \in \mathcal{P}_q} w_{\mathbf{f}} \frac{\partial d_{\mathbf{f}}^{(i)}}{\partial \mathbf{r}^{(ij)}} \tag{3.2.198}$$

$$= \sum_{q=1}^{q_{\max}} \sum_{[(n_1,\mathbf{k}_1)\cdots(n_q,\mathbf{k}_q)] \in \mathcal{P}_q} w_{[(n_1,\mathbf{k}_1)\cdots(n_q,\mathbf{k}_q)]} \frac{\partial}{\partial \mathbf{r}^{(ij)}} \left(\prod_{\nu=1}^{q} d_{n_\nu \mathbf{k}_\nu}^{(i)} \right) \tag{3.2.199}$$

$$= \sum_{q=1}^{q_{\max}} \sum_{[(n_1,\mathbf{k}_1)\cdots(n_q,\mathbf{k}_q)] \in \mathcal{P}_q} w_{[(n_1,\mathbf{k}_1)\cdots(n_q,\mathbf{k}_q)]} \sum_{\nu=1}^{q} \frac{\partial d_{n_\nu \mathbf{k}_\nu}^{(i)}}{\partial \mathbf{r}^{(ij)}} \prod_{\nu' \neq \nu} d_{n_{\nu'} \mathbf{k}_{\nu'}}^{(i)}. \tag{3.2.200}$$

Adjoint Method

In the force estimation, identical differentiation of both basis functions and polynomials repeatedly appears when PolyMLP is implemented straightforwardly. The force estimation can be accelerated by reducing these redundant differentiations. The number of differentiations can be reduced at the expense of memory preserving the calculated results. This method is referred to as the adjoint method [19, 79].

Polynomial Adjoint

The terms irrelevant to j are grouped in advance into a coefficient D such that satisfies the following,

$$\frac{\partial E^{(i)}}{\partial \mathbf{r}^{(ij)}} =: \sum_{n=0}^{n_{\max}} \sum_{\mathbf{k}\in\mathcal{K}} \frac{\partial d_{n\mathbf{k}}^{(i)}}{\partial \mathbf{r}^{(ij)}} D_{n\mathbf{k}}^{(i)}, \tag{3.2.201}$$

where

$$D_{n\mathbf{k}}^{(i)} \overset{\text{def}}{=} \sum_{q=1}^{q_{\max}} \sum_{[(n_1,\mathbf{k}_1)\cdots(n_q,\mathbf{k}_q)]\in\mathcal{P}_q} w_{[(n_1,\mathbf{k}_1)\cdots(n_q,\mathbf{k}_q)]} \sum_{\nu=1}^{q} \delta_{nn_\nu}\delta_{\mathbf{k}\mathbf{k}_\nu} \prod_{\nu'\neq\nu} d_{n_{\nu'}\mathbf{k}_{\nu'}}^{(i)}. \tag{3.2.202}$$

This D is indexed by an atom number i, a radial index n, and angular-element index $\mathbf{k}$,

$$\left\{ D_{n\mathbf{k}}^{(i)} \right\}_{i\in\{1,2,\cdots,N\},n\in\{0,1,\cdots,n_{\max}\},\mathbf{k}\in\mathcal{K}}. \tag{3.2.203}$$

In Equation (3.2.201), the part not related to the j derivative can be reused, so that the amount of calculation is reduced. This polynomial adjoint is implemented as pseudo code 2.

Algorithm 2 Compute Polynomial Adjoint $\{D^{(i)}_{n,\mathbf{k}}\}_{i\in\{1,2,\cdots,N\};\ n\le n_{\max};\ \mathbf{k}\in\mathcal{K}}$

Require: $\{d^{(i)}_{n,\mathbf{k}}\}_{i\in[N];n\in[n_{\max}];\mathbf{k}\in\mathcal{K}}$ ▷ Get structural features
for $i \in \{1, 2, \cdots, N\}$ **do** ▷ for all atoms
  for $n \in \{0, 1, \cdots, n_{\max}\}, \mathbf{k} \in \mathcal{K}$ **do** ▷ for all possible polynomials
    $D^{(i)}_{n,\mathbf{k}} \leftarrow 0$ ▷ Initialize polynomial adjoint
  end for
end for
for $i \in \{1, 2, \cdots, N\}$ **do** ▷ for all atoms
  for $[(n_1, \mathbf{k}_1), \ldots, (n_q, \mathbf{k}_q)] \in \mathcal{P}$ **do** ▷ for all combinations of polynomials
    for $\nu \in \{1, 2, \cdots, q\}$ **do** ▷ for all orders of polynomials
      $D^{(i)}_{n_\nu,\mathbf{k}_\nu} \leftarrow D^{(i)}_{n_\nu,\mathbf{k}_\nu} + w_{[(n_1,\mathbf{k}_1),\ldots,(n_q,\mathbf{k}_q)]} \prod_{\nu'\neq\nu} d^{(i)}_{n_{\nu'}\mathbf{k}_{\nu'}}$ ▷ P-adjoint:(3.2.202)
    end for
  end for
end for
return $\{D^{(i)}_{n,\mathbf{k}}\}_{i\in\{1,2,\cdots,N\};\ n\le n_{\max};\ \mathbf{k}\in\mathcal{K}}$

Basis Function Adjoint

To calculate the derivative of the structural feature, we need to recreate an adjoint with respect to the product of basis functions. That is, derivative of Equation (3.2.201) gives

$$\frac{\partial E^{(i)}}{\partial \mathbf{r}^{(ij)}} = \sum_{n=0}^{n_{\max}} \sum_{\mathbf{k}\in\mathcal{K}} \frac{\partial d^{(i)}_{n\mathbf{k}}}{\partial \mathbf{r}^{(ij)}} D^{(i)}_{n\mathbf{k}} \tag{3.2.204}$$

$$= \sum_{n} \sum_{p} \sum_{l_1\cdots l_p} \sum_{t_1\cdots t_p} \sum_{\sigma} D^{(i)}_{n,[l_1,\cdots,l_p],[t_1,\cdots,t_p],\sigma} \sum_{m_1\cdots m_p} C^{(l_1\cdots l_p),\sigma}_{m_1\cdots m_p} \times \sum_{\mu=1}^{p} \delta_{t_\mu,\left\{s^{(i)},s^{(j)}\right\}} \frac{\partial \phi_{nl_\mu m_\mu}}{\partial \mathbf{r}} \left(\mathbf{r}^{(ij)}\right) \prod_{\mu'\neq\mu} a^{(i)}_{nl_{\mu'}m_{\mu'},t_{\mu'}} \tag{3.2.205}$$

$$=: \sum_{n=0}^{n_{\max}} \sum_{l=0}^{l_{\max}} \sum_{m=-l}^{l} \frac{\partial \phi_{nlm}}{\partial \mathbf{r}} \left(\mathbf{r}^{(ij)}\right) \Phi_{nlm,\left\{s^{(i)},s^{(j)}\right\}}, \tag{3.2.206}$$

where the last line is an implicit definition of the basis function adjoint $\Phi_{nlm,t}$. Similarly to Eq (3.2.201), we can reuse the part not related to the position of atom-j and the amount of calculation is reduced. The basis function adjoint can be written explicitly as

$$\Phi_{nlm,\left\{s^{(i)},s^{(j)}\right\}} \overset{\text{def}}{=} \sum_{p=1}^{|\mathbf{l}|} \sum_{(\mathbf{l},\mathbf{t},\sigma)\in\mathcal{K}_p} D^{(i)}_{n\mathbf{l}\mathbf{t},\sigma} \sum_{m_1\cdots m_p} C^{(\mathbf{l}),\sigma}_{m_1\cdots m_p} \times \sum_{\mu=1}^{p} \delta_{t_\mu,\left\{s^{(i)},s^{(j)}\right\}} \delta_{ll_\mu} \delta_{mm_\mu} \prod_{\mu'\neq\mu} a^{(i)}_{nl_{\mu'}m_{\mu'},t_{\mu'}}. \tag{3.2.207}$$

This basis function adjoint can be obtained from pseudo code (3).

Algorithm 3 Compute Basis Function adjoint $\{\Phi^{(i)}_{nlm,t}\}_{i\in\{1,\cdots,N\};n\in\{0,\cdots,n_{\max}\};l\leq l_{\max};|m|\leq l;t\in\mathcal{S}}$

Require: $\{D^{(i)}_{n,\mathbf{k}}\}_{i\in\{1,2,\cdots,N\};\ n\leq n_{\max};\ \mathbf{k}\in\mathcal{K}}$

for $i \in \{1, 2, \cdots, N\}$ **do**
  for $t \in \mathcal{S}$ **do** ▷ for all pairs of elements
    for $n \in \{0, 1, \cdots, n_{\max}\}$ **do**
      for $l \leq l_{\max}, |m| \leq l$ **do**
        $\Phi^{(i)}_{nlm,t} \leftarrow 0$ ▷ Initialize basis function adjoint
      end for
    end for
  end for
end for
for $i \in \{1, 2, \cdots, N\}$ **do**
  for $(\mathbf{l}, \mathbf{t}, \sigma) \in \mathcal{K}$ **do** ▷ for all possible polynomials with angular-element terms
    if $s_i \notin e(\mathbf{t})$ **then** ▷ Skip if not related type (3.2.186)
      continue
    end if
    $p \leftarrow |\mathbf{l}|$
    for $n \in \{0, 1, \cdots, n_{\max}\}$ **do**
      for $\mathbf{m} \in \{m_1, m_2, \cdots, m_p\}$ **do**
        for $\mu \in \{1, 2, \cdots, p\}$ **do**
          $\Phi^{(i)}_{nl_\mu m_\mu, t_\mu} \leftarrow \Phi^{(i)}_{nl_\mu m_\mu, t_\mu} + D^{(i)}_{n,(\mathbf{l},\mathbf{t},\sigma)} C^{\mathbf{l},\sigma}_{\mathbf{m}} \prod_{\mu'\neq\mu} a^{(i)}_{nl_{\mu'}m_{\mu'},t_{\mu'}}$ ▷ (3.2.207)
        end for
      end for
    end for
  end for
end for
return $\{\Phi^{(i)}_{nlm,t}\}_{i\in\{1,\cdots,N\};\ n\in\{0,\cdots,n_{\max}\};\ l\leq l_{\max};\ |m|\leq l;\ t\in\mathcal{S}}$

We also define the derivative of $E^{(j)}$ here. The $r^{(ij)}$ is replaced by $r^{(ji)}$, which corresponds to the spatial inversion, and the spatial inversion of the spherical harmonics Y is given by the azimuthal quantum number

$$Y^m_l\left(-\hat{\mathbf{r}}\right) = (-)^l Y^m_l\left(\hat{\mathbf{r}}\right). \tag{3.2.208}$$

Substituting (3.2.208) into equation (3.2.206) yields:

$$\frac{\partial E^{(j)}}{\partial \mathbf{r}^{(ji)}} = -\sum_{n=0}^{n_{\max}}\sum_{l=0}^{l_{\max}}\sum_{m=-l}^{l}\frac{\partial \phi_{nlm}}{\partial \mathbf{r}}\left(\mathbf{r}^{(ji)}\right)\Phi_{nlm,\left\{s^{(j)},s^{(i)}\right\}} \tag{3.2.209}$$

$$= \sum_{n=0}^{n_{\max}}\sum_{l=0}^{l_{\max}}(-)^{l+1}\sum_{m=-l}^{l}\frac{\partial \phi_{nlm}}{\partial \mathbf{r}}\left(\mathbf{r}^{(ij)}\right)\Phi_{nlm,\left\{s^{(i)},s^{(j)}\right\}} \tag{3.2.210}$$

Force calculation

Equations (3.2.195), (3.2.206) and (3.2.210) give an expression of the pairwise force with two basis adjoints

$$\mathbf{F}^{(ij)} = \frac{\partial E^{(i)}}{\partial \mathbf{r}^{(ij)}} - \frac{\partial E^{(j)}}{\partial \mathbf{r}^{(ji)}} \tag{3.2.211}$$

$$= \sum_{n=0}^{n_{\max}} \sum_{l=0}^{l_{\max}} \left(\sum_{m=-l}^{l} \frac{\partial \phi_{nlm}}{\partial \mathbf{r}} \left(\mathbf{r}^{(ij)}\right) \Phi_{nlm,\left\{s^{(i)},s^{(j)}\right\}} \right.$$
$$\left. - (-)^{l+1} \sum_{m=-l}^{l} \frac{\partial \phi_{nlm}}{\partial \mathbf{r}} \left(\mathbf{r}^{(ij)}\right) \Phi_{nlm,\left\{s^{(i)},s^{(j)}\right\}} \right) \tag{3.2.212}$$

$$= \sum_{n=0}^{n_{\max}} \sum_{l=0}^{l_{\max}} \left(\frac{\partial \phi_{nl0}}{\partial \mathbf{r}} \left(\mathbf{r}^{(ij)}\right) \Phi_{nl0,\left\{s^{(i)},s^{(j)}\right\}} \right.$$
$$\left. + 2\mathrm{Re}\left[\sum_{m=1}^{l} \frac{\partial \phi_{nlm}}{\partial \mathbf{r}} \left(\mathbf{r}^{(ij)}\right) \Phi_{nlm,\left\{s^{(i)},s^{(j)}\right\}} \right] \right). \tag{3.2.213}$$

The formula transformation in the last line halves the amount of calculation by separating the m sum into 0 and others. This derivation will be shown below.

From the nature of the spherical harmonic function, the following relationship is self-evident

$$\frac{\partial \phi_{nl-m}}{\partial \mathbf{r}} = (-)^m \overline{\frac{\partial \phi_{nlm}}{\partial \mathbf{r}}}. \tag{3.2.214}$$

This relation gives the m-inversion of the basis function adjoint

$$\Phi_{nl-m,\left\{s^{(i)},s^{(j)}\right\}} = \sum_{p=1}^{|\mathbf{l}|} \sum_{\mathbf{k}\in\mathcal{K}_p} D^{(i)}_{n\mathbf{k}} \sum_{\mathbf{m}_p} C^{(\mathbf{l}),\sigma}_{\mathbf{m}_p} \sum_{\mu=1}^{p} \delta_{t_\mu,\left\{s^{(i)},s^{(j)}\right\}} \delta_{ll_\mu} \delta_{-mm_\mu} \prod_{\mu'\neq\mu} a^{(i)}_{nl_{\mu'}m_{\mu'},t_{\mu'}} \tag{3.2.215}$$

$$= \sum_{p=1}^{|\mathbf{l}|} \sum_{\mathbf{k}\in\mathcal{K}_p} D^{(i)}_{n\mathbf{k}} \sum_{\mathbf{m}_p} C^{(\mathbf{l}),\sigma}_{-\mathbf{m}_p} \sum_{\mu=1}^{p} \delta_{t_\mu,\left\{s^{(i)},s^{(j)}\right\}} \delta_{ll_\mu} \delta_{m-m_\mu} \prod_{\mu'\neq\mu} a^{(i)}_{nl_{\mu'}-m_{\mu'},t_{\mu'}} \tag{3.2.216}$$

$$= \sum_{p=1}^{|\mathbf{l}|} \sum_{\mathbf{k}\in\mathcal{K}_p} D^{(i)}_{n\mathbf{k}} \sum_{\mathbf{m}_p} \overline{C^{(\mathbf{l}),\sigma}_{\mathbf{m}_p}} \sum_{\mu=1}^{p} \delta_{t_\mu,\left\{s^{(i)},s^{(j)}\right\}} \delta_{ll_\mu} \delta_{m-m_\mu}$$
$$\times (-)^{\sum_{\mu'\neq\mu} m_{\mu'}} \prod_{\mu'\neq\mu} \overline{a^{(i)}_{nl_{\mu'}m_{\mu'},t_{\mu'}}} \tag{3.2.217}$$

$$= \sum_{p=1}^{|\mathbf{l}|} \sum_{\mathbf{k}\in\mathcal{K}_p} D^{(i)}_{n\mathbf{k}} \sum_{\mathbf{m}_p} \overline{C^{(\mathbf{l}),\sigma}_{\mathbf{m}_p}} \sum_{\mu=1}^{p} \delta_{t_\mu,\left\{s^{(i)},s^{(j)}\right\}} \delta_{ll_\mu} \delta_{m-m_\mu}$$
$$\times (-)^{m_\mu} \prod_{\mu'\neq\mu} \overline{a^{(i)}_{nl_{\mu'}m_{\mu'},t_{\mu'}}} \tag{3.2.218}$$

$$= \overline{\Phi_{nlm,\left\{s^{(i)},s^{(j)}\right\}}} \left(-\right)^{m}. \tag{3.2.219}$$

In the transformation from (3.2.216) to (3.2.217), we use a relation

$$C^{(\mathbf{l}),\sigma}_{-\mathbf{m}_p} = \overline{C^{(\mathbf{l}),\sigma}_{\mathbf{m}_p}}. \tag{3.2.220}$$

This relationship between CG coefficients is due to the fact that irreducible representations of any finite group can be equivalently transformed into unitary representations. Therefore, the solution of the eigenvalue problem of the projection operator can be transformed into a unitary representation, and if it is selected as the coefficient $C^{(\mathbf{l}),\sigma}_{\mathbf{m}_p}$, the above relation is satisfied. From (3.2.214) and (3.2.219), we have for $m > 0$

$$\frac{\partial \phi_{nl-m}}{\partial \mathbf{r}}\left(\mathbf{r}^{(ij)}\right) \Phi_{nl-m,\left\{s^{(i)},s^{(j)}\right\}} = \left(-\right)^{2m} \overline{\frac{\partial \phi_{nlm}}{\partial \mathbf{r}}\left(\mathbf{r}^{(ij)}\right) \Phi_{nlm,\left\{s^{(i)},s^{(j)}\right\}}}, \tag{3.2.221}$$

and finally obtain

$$\begin{aligned} &\frac{\partial \phi_{nlm}}{\partial \mathbf{r}}\left(\mathbf{r}^{(ij)}\right) \Phi_{nlm,\left\{s^{(i)},s^{(j)}\right\}} + \frac{\partial \phi_{nl-m}}{\partial \mathbf{r}}\left(\mathbf{r}^{(ij)}\right) \Phi_{nl-m,\left\{s^{(i)},s^{(j)}\right\}} \\ &\qquad = 2\mathrm{Re}\left[\sum_{m=1}^{l} \frac{\partial \phi_{nlm}}{\partial \mathbf{r}}\left(\mathbf{r}^{(ij)}\right) \Phi_{nlm,\left\{s^{(i)},s^{(j)}\right\}}\right]. \end{aligned} \tag{3.2.222}$$

Virial Contribution to Stress Tensor

The virial contribution to the stress tensor is given as

$$\boldsymbol{\sigma} \overset{\mathrm{def}}{=} \sum_{i=1}^{N} \mathbf{r}^{(i)} \otimes \mathbf{F}^{(i)} \tag{3.2.223}$$

$$= \sum_{i=1}^{N} \mathbf{r}^{(i)} \otimes \sum_{j \in \mathcal{N}_i} \mathbf{F}^{(ij)} \tag{3.2.224}$$

$$= \sum_{i=1}^{N} \sum_{j \in \mathcal{N}_i} \mathbf{r}^{(i)} \otimes \mathbf{F}^{(ij)} \tag{3.2.225}$$

$$= \frac{1}{2} \sum_{i=1}^{N} \sum_{j \in \mathcal{N}_i} \left(\mathbf{r}^{(i)} \otimes \mathbf{F}^{(ij)} + \mathbf{r}^{(j)} \otimes \mathbf{F}^{(ji)}\right) \tag{3.2.226}$$

$$= \frac{1}{2} \sum_{i=1}^{N} \sum_{j \in \mathcal{N}_i} \left(\mathbf{r}^{(i)} \otimes \mathbf{F}^{(ij)} - \mathbf{r}^{(j)} \otimes \mathbf{F}^{(ij)}\right) \tag{3.2.227}$$

$$= -\frac{1}{2} \sum_{i=1}^{N} \sum_{j \in \mathcal{N}_i} \left(\left(\mathbf{r}^{(j)} - \mathbf{r}^{(i)}\right) \otimes \mathbf{F}^{(ij)}\right) \tag{3.2.228}$$

$$= -\frac{1}{2}\sum_{i=1}^{N}\sum_{j\in\mathcal{N}_i}\mathbf{r}^{(ij)}\otimes\mathbf{F}^{(ij)}. \tag{3.2.229}$$

Therefore the stress tensor also can be obtained by the pair positional vector and the pairwise force.

Algorithm 4 Compute forces $\{\mathbf{F}^{(i)}\}_{i\in\{1,2\cdots,N\}}$ and stresses $\boldsymbol{\sigma}$

Require: $\{d^{(i)}_{n,\mathbf{k}}\}_{i\in\{1,\cdots,N\};n\in\{0,\cdots,n_{\max}\};\mathbf{k}\in\mathcal{K}}$

$\mathbf{F}^{(i)} \leftarrow \mathbf{0}\ (\forall i)$

$\sigma \leftarrow \mathbf{0}$

compute $\{\Phi^{(i)}_{n,\mathbf{k}}\}_{i\in\{1,\cdots,N\};n\in\{0,\cdots,n_{\max}\};\mathbf{k}\in\mathcal{K}}$ ▷ Store to reuse adjoints

for $i \in \{1,\cdots,N\}$ **do** ▷ Loop for real atoms

  for $j \in \mathcal{N}_i$ **do** ▷ Loop for neighbor

    $\partial E^{(ij)} \leftarrow 0, \partial E^{(ji)} \leftarrow 0$ ▷ Initialize pairwise derivatives

    compute $\left\{\frac{\partial\phi_{nlm}}{\partial\mathbf{r}}(\mathbf{r}^{(ij)})\right\}_{n,l,m\geq 0}$ ▷ Store to reuse derivatives

    for $n \in \{0,\cdots,n_{\max}\}$ **do**

      for $l \leq l_{\max}$ **do**

        $\partial E^{(ij)} \leftarrow \partial E^{(ij)} + \frac{\partial\phi_{nl0}}{\partial\mathbf{r}}\left(\mathbf{r}^{(ij)}\right)\Phi^{(i)}_{nl0,\left\{s^{(i)},s^{(j)}\right\}}$ ▷ (3.2.206)(3.2.222)

        $+2\sum_{m>0}\mathrm{Re}\left[\frac{\partial\phi_{nlm}}{\partial\mathbf{r}}\left(\mathbf{r}^{(ij)}\right)\Phi^{(i)}_{nlm,\left\{s^{(i)},s^{(j)}\right\}}\right]$

        $\partial E^{(ji)} \leftarrow \partial E^{(ji)} + (-)^{l+1}\left(\frac{\partial\phi_{nl0}}{\partial\mathbf{r}}\left(\mathbf{r}^{(ij)}\right)\Phi^{(j)}_{nl0,\left\{s^{(i)},s^{(j)}\right\}}\right.$ ▷ (3.2.210)(3.2.222)

        $\left.+2\sum_{m>0}\mathrm{Re}\left[\frac{\partial\phi_{nlm}}{\partial\mathbf{r}}\left(\mathbf{r}^{(ij)}\right)\Phi^{(j)}_{nlm,\left\{s^{(i)},s^{(j)}\right\}}\right]\right)$

      end for

    end for

    $\mathbf{F}^{(ij)} \leftarrow \partial E^{(ij)} - \partial E^{(ji)}$ ▷ Get pairwise force (3.2.211)

    $\mathbf{F}^{(i)} \leftarrow \mathbf{F}^{(i)} + \mathbf{F}^{(ij)}$ ▷ Add j contribution

    $\mathbf{F}^{(j)} \leftarrow \mathbf{F}^{(j)} - \mathbf{F}^{(ij)}$ ▷ Newton's 3rd law

    $\boldsymbol{\sigma} \leftarrow \boldsymbol{\sigma} - \mathbf{r}^{(ij)}\otimes\mathbf{F}^{(ij)}$ ▷ Update stress tensor

  end for

end for

return $\{\mathbf{F}^{(i)}\}_{i\in\{1,\cdots,N\}}$ and $\boldsymbol{\sigma}$

Performant Calculation of Spherical Harmonics

Spherical harmonics are key calculations in various fields, including quantum chemistry. Significant effort has been made since the 1960s and various algorithms have been developed for efficient calculation. The standard approach is to make use of a recurrence relation, which is noticeably effective for improving computational efficiency. However, for some recursive relations round-off error grows rapidly and the results can be incorrect. In this section, an efficient and accurate algorithm by Limpanuparb [98] is reviewed and extended to a recurrence relation of derivatives for force calculations.

The solutions of the associated Legendre differential equation

$$\left(1-x^2\right)\frac{\mathrm{d}^2y}{\mathrm{d}x^2}-2x\frac{\mathrm{d}y}{\mathrm{d}x}+\left\{l\left(l+1\right)-\frac{m^2}{1-x^2}\right\}y=0 \tag{3.2.230}$$

are linear combination of a Legendre function of the first kind $P_l^m\left(x\right)$ and a Legendre function of second kind $Q_l^m\left(x\right)$ by Ferrer's form for $|x|<1$ and $m\geq 0$

$$\begin{cases} P_l^m\left(x\right) & =\left(1-x^2\right)^{m/2}\frac{\mathrm{d}^m P_l(x)}{\mathrm{d}x^m} \\ Q_l^m\left(x\right) & =\left(1-x^2\right)^{m/2}\frac{\mathrm{d}^m Q_l(x)}{\mathrm{d}x^m}. \end{cases} \tag{3.2.231}$$

Since the Laplace's differential equation in spherical coordinates is equivalent to the Legendre differential equation, the Legendre function of the first kind P_l^m of degree l and order $m\geq 0$ is closely related to the spherical harmonics

$$Y_l^m\left(\theta,\phi\right)=\sqrt{\frac{2l+1}{4\pi}\frac{(l-m)!}{(l+m)!}}P_l^m\left(\cos\theta\right)e^{im\phi}. \tag{3.2.232}$$

They are sometimes separated into their real and imaginary parts,

$$\begin{cases} Y_l^{ms}\left(\theta,\phi\right) & =\sqrt{\frac{2l+1}{4\pi}\frac{(l-m)!}{(l+m)!}}P_l^m\left(\cos\theta\right)\sin\left(m\phi\right) \\ Y_l^{mc}\left(\theta,\phi\right) & =\sqrt{\frac{2l+1}{4\pi}\frac{(l-m)!}{(l+m)!}}P_l^m\left(\cos\theta\right)\cos\left(m\phi\right). \end{cases} \tag{3.2.233}$$

Note that only P_l^m is employed due to the continuity at $\theta=0$. It follows that the essential part for calculation of the spherical harmonics is the calculation of the Legendre function of the first kind.

Using the Lanczos principle of the orthogonal function, the Legendre polynomials obey the following ternary recurrence relations for $l\geq 2$ and $0\leq m\leq l-2$

$$P_l^m\left(\mu\right)=\frac{2l-1}{l-m}\mu P_{l-1}^m\left(\mu\right)-\frac{l-1+m}{l-m}P_{l-2}^m\left(\mu\right), \tag{3.2.234}$$

where $\cos\theta$ is denoted as μ.

Taking the normalized associated Legendre polynomial for $0\leq\theta\leq\pi$ as

$$\bar{P}_l^m\left(\cos\theta\right)=\sqrt{\frac{2l+1}{2\pi}\frac{(l-m)!}{(l+m)!}}P_l^m\left(\cos\theta\right), \tag{3.2.235}$$

(3.2.232) gives

$$Y_l^m\left(\theta,\phi\right)=\sqrt{\frac{2l+1}{4\pi}\frac{(l-m)!}{(l+m)!}}P_l^m\left(\cos\theta\right)e^{im\phi} \tag{3.2.236}$$

$$= \sqrt{\frac{2l+1}{4\pi}\frac{(l-m)!}{(l+m)!}}\sqrt{\frac{2l+1}{2\pi}\frac{(l-m)!}{(l+m)!}}^{-1} \bar{P}_l^m(\cos\theta)\, e^{im\phi} \tag{3.2.237}$$

$$= \frac{1}{\sqrt{2}}\bar{P}_l^m(\cos\theta)\, e^{im\phi}. \tag{3.2.238}$$

Transforming the normalized associated Legendre polynomial (3.2.235) using the ternary recurrence formula (3.2.234) gives

$$\bar{P}_l^m(\mu) = \sqrt{\frac{2l+1}{2}\frac{(l-m)!}{(l+m)!}}P_l^m(\mu) \tag{3.2.239}$$

$$= \sqrt{\frac{2l+1}{2}\frac{(l-m)!}{(l+m)!}}\left(\frac{2l-1}{l-m}\mu\sqrt{\frac{2(l-1)+1}{2\pi}\frac{((l-1)-m)!}{((l-1)+m)!}}^{-1}\bar{P}_{l-1}^m(\mu)\right.$$
$$\left. -\frac{(l-1)+m}{l-m}\sqrt{\frac{2(l-2)-1}{2\pi}\frac{((l-2)-m)!}{((l-2)+m)!}}^{-1}\bar{P}_{l-2}^m(\mu)\right) \tag{3.2.240}$$

$$= \frac{2l-1}{l-m}\mu\sqrt{\frac{2l+1}{2l-1}\frac{(l-m)!\,((l-1)+m)!}{(l+m)!\,((l-1)-m)!}}\bar{P}_{l-1}^m(\mu)$$
$$- \frac{(l-1)+m}{l-m}\sqrt{\frac{2l+1}{2(l-2)+1}\frac{(l-m)!\,((l-2)+m)!}{(l+m)!\,((l-2)-m)!}}\bar{P}_{l-2}^m(\mu) \tag{3.2.241}$$

$$= \mu\sqrt{\left(\frac{2l-1}{l-m}\right)^2\frac{2l+1}{2l-1}\frac{(l-m)}{(l+m)}}\bar{P}_{l-1}^m(\mu)$$
$$- \sqrt{\left(\frac{(l-1)+m}{l-m}\right)^2\frac{2l+1}{2(l-2)+1}\frac{(l-m)\,((l-1)-m)}{(l+m)\,((l-1)+m)}}\bar{P}_{l-2}^m(\mu) \tag{3.2.242}$$

$$= \mu\sqrt{\left(\frac{2l-1}{l-m}\right)^2\frac{2l+1}{2l-1}\frac{(l-m)}{(l+m)}}\bar{P}_{l-1}^m(\mu)$$
$$- \sqrt{\frac{(l-1)^2-m^2}{l^2-m^2}\frac{2l+1}{2(l-2)+1}}\bar{P}_{l-2}^m(\mu) \tag{3.2.243}$$

$$= \mu\sqrt{\frac{4l^2-1}{l^2-m^2}}\bar{P}_{l-1}^m(\mu) - \sqrt{\frac{4l^2-1}{l^2-m^2}}\sqrt{\frac{(l-1)^2-m^2}{4l^2-1}\frac{2l+1}{2(l-2)-1}}\bar{P}_{l-2}^m(\mu) \tag{3.2.244}$$

$$= \sqrt{\frac{4l^2-1}{l^2-m^2}}\left(\mu\bar{P}_{l-1}^m(\mu) - \sqrt{\frac{(l-1)^2-m^2}{2l-1}\frac{1}{(2(l-2)-1)}}\bar{P}_{l-2}^m(\mu)\right) \tag{3.2.245}$$

$$= \sqrt{\frac{4l^2-1}{l^2-m^2}}\left(\mu\bar{P}_{l-1}^m(\mu) - \sqrt{\frac{(l-1)^2-m^2}{4(l-1)^2-1}}\bar{P}_{l-2}^m(\mu)\right) \tag{3.2.246}$$

$$= a_l^m\left(\mu\bar{P}_{l-1}^m(\mu) + b_l^m\bar{P}_{l-2}^m(\mu)\right). \tag{3.2.247}$$

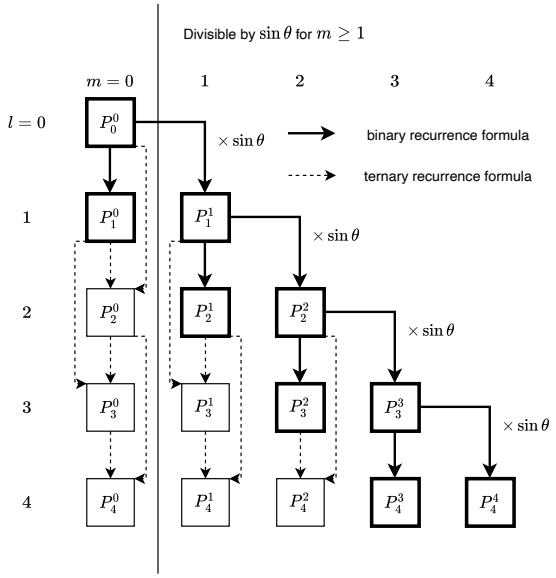


Figure 3.2.2: Recursion relation for computing the associated Legendre polynomials. The ternary and binary recurrence formulas are used in the azimuth quantum number direction and the magnetic quantum number direction, respectively. Since the binary recurrence formula for the first term of each magnetic quantum number multiplies $\sin\theta$, the Legendre polynomials of $m \geq 1$ avoid divergence for small θ.

Thus we have the ternary recurrence formula (3.2.234) for the normalized Legendre polynomial as

$$\bar{P}_l^m(\mu) = a_l^m \left(\mu \bar{P}_{l-1}^m(\mu) + b_l^m \bar{P}_{l-2}^m(\mu)\right), \tag{3.2.248}$$

where a_l^m and b_l^m are defined as

$$\begin{cases} a_l^m &= \sqrt{\frac{4l^2-1}{l^2-m^2}}, \\ b_l^m &= -\sqrt{\frac{(l-1)^2-m^2}{4(l-1)^2-1}}. \end{cases} \tag{3.2.249}$$

Combining (3.2.235) with two binary recurrence relations

$$\bar{P}_m^m(\cos\theta) = -\sqrt{1+\frac{1}{2m}}\sin\theta \bar{P}_{m-1}^{m-1}(\cos\theta) \tag{3.2.250}$$

$$\bar{P}_{m+1}^m(\cos\theta) = \sqrt{2m+3}\cos\theta \bar{P}_m^m(\cos\theta) \tag{3.2.251}$$

and

$$\bar{P}_0^0 = \sqrt{\frac{1}{2\pi}} \tag{3.2.252}$$

we obtain the spherical harmonic functions required for the order parameter sequentially.

The derivatives of the spherical harmonic functions are given in [22, 99] as

$$\nabla Y_l^m(\theta,\phi) = \frac{im\bar{P}_l^m(\mu)\,e^{im\phi}}{\sqrt{2}r\sin\theta}\begin{pmatrix}-\sin\phi\\ \cos\phi\\ 0\end{pmatrix} + \frac{\left(\frac{\mathrm{d}}{\mathrm{d}\theta}\bar{P}_l^m(\mu)\right)e^{im\phi}}{\sqrt{2}r}\begin{pmatrix}\cos\phi\cos\theta\\ \sin\phi\cos\theta\\ -\sin\theta\end{pmatrix}. \quad (3.2.253)$$

To evaluate this gradient numerically stable, the singularity of $(\sin\theta)^{-1}$ at the pole must be removed. If $m = 0$, the singularity is readily removed since the spherical harmonics does not depend on the azimuth ϕ. Otherwise, (3.2.250) indicates all $\bar{P}_l^m$ can be divisible by $\sin\theta$ for $m \neq 0$, using $\bar{P}_l^m/\sin\theta$ instead stabilize the evaluation. The analytic form of (3.2.253) can be transformed into a numerically stable form as

$$\nabla Y_l^m(\theta,\phi) = \begin{cases} \frac{\left(\frac{\mathrm{d}}{\mathrm{d}\theta}\bar{P}_l^0(\mu)\right)}{\sqrt{2}r}\begin{pmatrix}\cos\phi\cos\theta\\ \sin\phi\cos\theta\\ -\sin\theta\end{pmatrix}, & (\text{if } m=0)\\ \frac{im\bar{R}_l^m(\theta)e^{im\phi}}{\sqrt{2}r}\begin{pmatrix}-\sin\phi\\ \cos\phi\\ 0\end{pmatrix} + \frac{\left(\frac{\mathrm{d}}{\mathrm{d}\theta}\bar{P}_l^m(\mu)\right)e^{im\phi}}{\sqrt{2}r}\begin{pmatrix}\cos\phi\cos\theta\\ \sin\phi\cos\theta\\ -\sin\theta\end{pmatrix}, & (\text{otherwise}) \end{cases} \quad (3.2.254)$$

where

$$\bar{R}_l^m(\theta) \overset{\text{def}}{=} \frac{\bar{P}_l^m(\cos\theta)}{\sin\theta}. \quad (3.2.255)$$

$\bar{R}_l^m$ follows the same recurrence formula to $\bar{P}_l^m$ for $m \geq 1$

$$\bar{R}_l^m(\theta) = a_l^m\left(\cos\theta\bar{R}_{l-1}^m(\theta) + b_l^m\bar{R}_{l-2}^m(\theta)\right) \quad (3.2.256)$$

$$\bar{R}_m^m(\theta) = -\sqrt{1+\frac{1}{2m}}\sin\theta\bar{R}_{m-1}^{m-1}(\theta) \quad (3.2.257)$$

$$\bar{R}_{m+1}^m(\theta) = \sqrt{2m+3}\cos\theta\bar{R}_m^m(\theta) \quad (3.2.258)$$

starting from

$$\bar{R}_1^1 = -\sqrt{\frac{3}{4\pi}}. \quad (3.2.259)$$

Using the derivative of the associated Legendre polynomial gives

$$\nabla Y_l^m(\theta,\phi) = \begin{cases} \frac{1}{\sqrt{2}r}\left(\sqrt{l(l+1)}\bar{P}_l^1(\cos\theta)\right)\begin{pmatrix}\cos\phi\cos\theta\\ \sin\phi\cos\theta\\ -\sin\theta\end{pmatrix}, & (\text{if } m=0) \\ \frac{e^{im\phi}}{\sqrt{2}r}\left(im\bar{R}_l^m(\theta)\begin{pmatrix}-\sin\phi\\ \cos\phi\\ 0\end{pmatrix} + \left(m\cos\theta\bar{R}_l^m(\theta)\right.\right. & \\ \left.\left. +\sqrt{(l-m)(l+m+1)}\bar{P}_l^{m+1}(\cos\theta)\right)\begin{pmatrix}\cos\phi\cos\theta\\ \sin\phi\cos\theta\\ -\sin\theta\end{pmatrix}\right). & (\text{otherwise}) \end{cases} \tag{3.2.260}$$

3.2.3 Spatial Decomposition and MPI Parallelization

The molecular dynamics itself is inherently parallel, in other words, positions and velocities for all particles can be updated simultaneously. When simulating a system with many particles, it is effective to decompose the simulation box and allocate processors or a compute node to each decomposed part. There are three classes of algorithms for parallelizing the molecular dynamics calculation [3], that is splitting and replicating particles (atom decomposition), partitioning forces (force decomposition), and geometrically splitting of simulation box (spatial decomposition).

LAMMPS takes the last one, spatial decomposition, where each processor has information of both its atoms and some of those belonging to adjacent processors. These particles refer to real atoms and ghost atoms, respectively. Figure 3.2.3 shows an example configuration of real and ghost atoms near a processor border. When a system is decomposed spatially, each partial system acts as if it has complete information for calculating the energy and forces using real and ghost atoms. Accumulating calculation results for each partial system should reproduce the whole system through inter-processor MPI communication.

By following the basic MD timesteps with the spatial decomposition shown in Fig. 3.2.4, the information of the total system is distributed, calculated, and reproduced. Real atoms and ghost atoms communicate their properties to each other via MPI forward and reverse communication over processors. The MPI forward communication copies the properties of the ghost atoms from the corresponding real atoms of the adjacent processors before computing forces. After computing forces, the MPI reverse communication transfers the information accumulated on these ghost atoms into the corresponding real atoms of the adjacent processors.

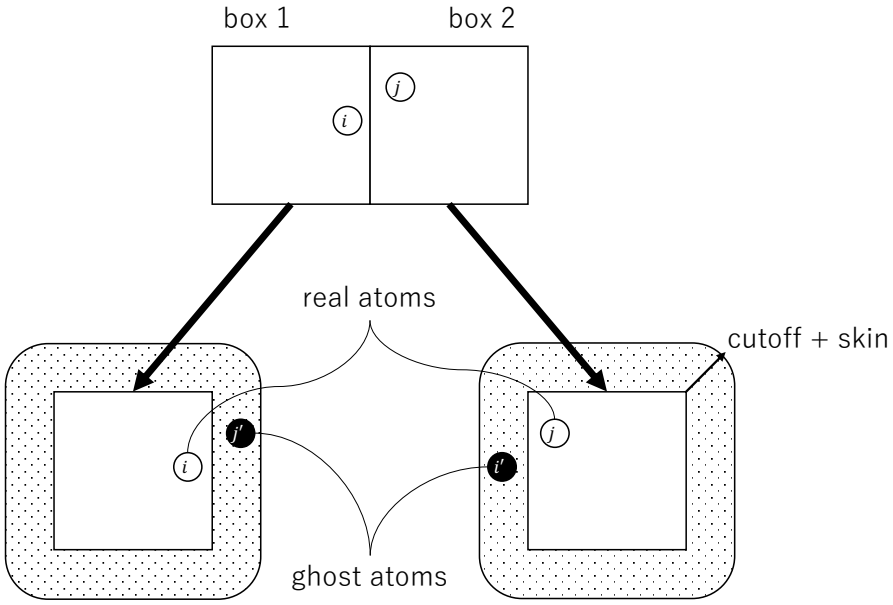


Figure 3.2.3: Spatial decomposition of simulation box. Each box has a shell with a thickness of cutoff + skin. Ghost atoms are atoms that are included not in each box but in the shell around each box.

LAMMPS provides four types of spatial decomposition, which consist of combinations of two neighbor styles, which is half or full, and two communication methods, which is Newton's third law enabled or disabled. With the half neighbor style, each pair of atoms is stored only once and a pairwise interaction acting on the other atom is assumed to be equal in magnitude and opposite in direction. On the other hand, using the full neighbor style, each pair is counted twice and pairwise interactions act on each atom individually. The half neighbor list reduces the number of force evaluations and requires more communication between processors, on the contrary, the full neighbor list doubles the number of force evaluations and reduces the communication. The flag for Newton's third law manages inter-processor communication of atoms near the border of each decomposed box.

For almost interatomic potentials, the half neighbor style with Newton's third law enabled and the full neighbor style with Newton's third law disabled is chosen for CPU and GPU, respectively. However, PolyMLP is so complicated that it needs special care to calculate the forces, whereas the energies can be simply calculated like other potentials, as shown in Fig. 3.2.5. For PolyMLP, a combination of the full-neighbor style and Newton's third law enabled is one of the best to calculate forces, which eventually coincides with the existing LAMMPS implementations of other MLIPs e. g. HDNNP, SNAP, PACE, or RANN.

When calculating the forces acting on the real atoms near the processor border, the atomic energy of the ghost atoms needs to be differentiated. However, this differentiation requires inner variables of the ghost atoms thus their hundreds of inner variables must be

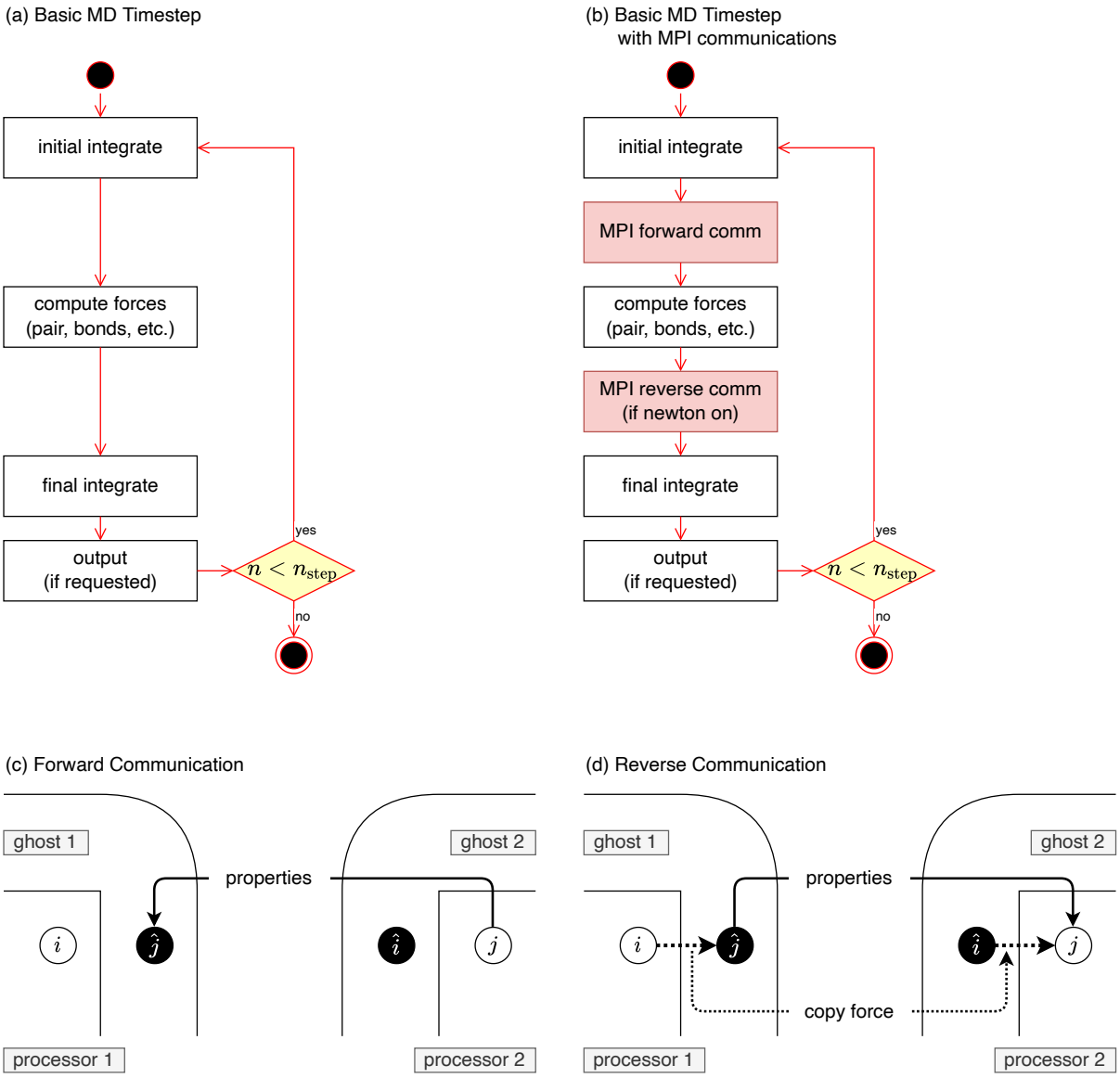


Figure 3.2.4: Simplified outline diagrams of MD timestep in LAMMPS. (a) The basic MD operations on a single simulation box. (b) The basic MD operations with spatial parallelization. The red-shaded box gives MPI forward- and reverse-communications, which transfer information between processors assigned to each decomposed box. (c) The MPI forward communication before computing forces. Properties of ghost atoms are updated from the corresponding real atoms in the adjacent processors. (d) The MPI reverse communication after computing forces. This communication transfers properties accumulated on the ghost atoms and forces acting on the ghost atoms to the corresponding real atoms.

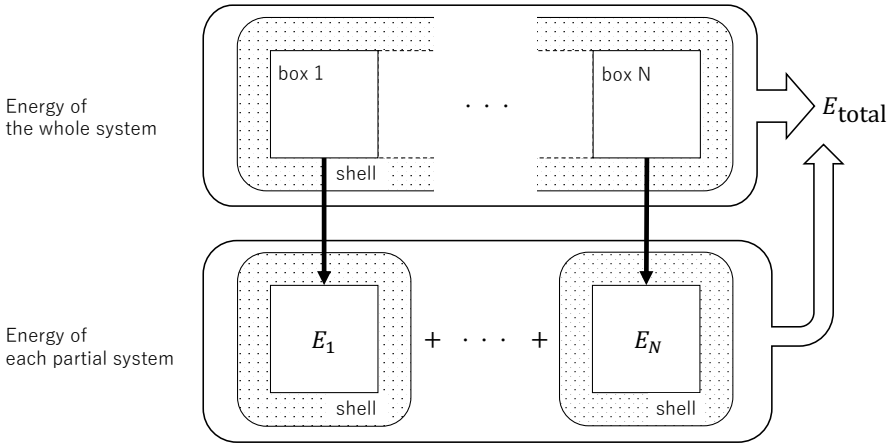


Figure 3.2.5: Energy reproduction through spatial decomposition. The total energy of the system is reproduced by simply adding each energy of the real atoms of partial system, even when the energy is estimated via MLIPs that use the atomic environment of ghost atoms in the shell. Only forward communication is used for this process.

copied from other processors. Even for a small system of hundreds of atoms, the number of ghost atoms can be thousands, thus the variables to be copied would become hundreds of thousands for each processor. Since communication between processors is usually less than one-tenth of the memory bandwidth, copying the internal variables of the ghost atoms causes a significant slowdown. Therefore, the full neighbor style should be used to avoid inter-processor communication of internal variables of the ghost atoms.

When using the full neighbor style, the Newton flag is typically disabled for simple interatomic potentials. Since these simple potentials calculate both pairwise forces acting on an atom only by relative positions stored in a single neighbor list, such potentials do not require inter-processor communication other than the position of ghost atoms. However, MLIPs use the derivatives of the atomic potential energy of ghost atoms to calculate forces, which cannot be calculated only by a single neighbor list but require atomic environments of all ghost atoms instead. If the newton flag is disabled, these environments of all ghost atoms should be copied between processors thus it consumes memory resources, moreover, the order parameter needs to be calculated redundantly. If these order parameters are communicated, it slows down the calculation, as already mentioned in the need for the full neighbor style. In addition, from a technical viewpoint, it is prohibitively difficult to communicate order parameters using personally developed code without following the LAMMPS framework. Therefore, it is the practical choice to enable the Newton flag to communicate the pairwise forces that are calculated on each processor.

The pairwise force calculation of the atoms near the border should be modified to calculate MLIPs efficiently as shown in Fig. 3.2.6. In typical cases, the pairwise force originating from the ghost atom's potential is calculated in the processor where the real atom exists, whereas, for MLIPs, this pairwise force is calculated in the adjacent processor

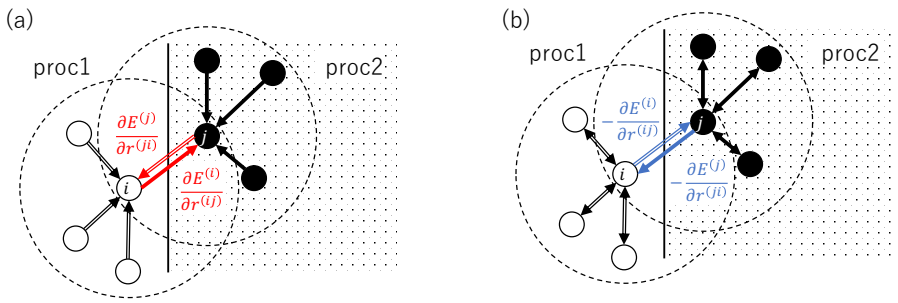


Figure 3.2.6: Modification of the pairwise force to calculate MLIPs. (a) A typical setting for calculating pairwise forces between atoms beyond processor border. It assumes that the atomic energy of the atom j on processor 2 should be differentiable on processor 1, however, this assumption cannot be applied to MLIPs. (b) The proposed setting for calculating pairwise forces of MLIPs. The reaction of the pairwise force is calculated on the processor where the atom exists.

where the ghost atom exists. The MPI reverse communication accumulates these forces to reproduce the whole system.

Although this pattern is practical enough, another implementation can be realized in theory, whose Newton flag is disabled. The implementation should reduce the MPI reverse communication when complete information about the neighbor list of the ghost atoms is copied via the MPI forward communication. In this case, the inner variables of each ghost atom are calculated on the processor where the real atom exists. However, it instead increases the MPI forward communication and causes redundant calculation of order parameters. As a result, this implementation ought to be slower than that practical implementation.

The spatial decomposition and communication method employed in the present research can be less efficient than a method by communicating the order parameters when the complexity of order parameters is dominant, for example, computing low-order interatomic potential models. As a practical matter, however, a highly accurate type of PolyMLP is usually selected for calculation whose amount of calculation is large, thus it is hard to realize a situation where the implementation by communicating order parameters would be more efficient.

Consequently, by considering the decomposition from both computational efficiency and technical difficulties, PolyMLP should use the full neighbor style with Newton's third law enabled. The computation of forces and stresses under spatial decomposition is implemented as pseudo code 5.

Algorithm 5 Compute forces $\{\mathbf{F}^{(i)}\}_{i\in\{1,2\cdots,N\}}$ and stresses $\boldsymbol{\sigma}$

Require: $\{d^{(i)}_{n,\mathbf{k}}\}_{i\in\{1,\cdots,N\};n\in\{0,\cdots,n_{\max}\};\mathbf{k}\in\mathcal{K}}$

$\mathbf{F}^{(i)} \leftarrow \mathbf{0}$ $(\forall i)$
$\sigma \leftarrow \mathbf{0}$
for $\xi \in \{1,\cdots,\#\text{ of nodes}\}$ **do** ▷ Loop for nodes
  for $i \in \mathcal{L}^{(\xi)}$ **do** ▷ Loop for real atoms in proc-ξ
    compute $\{\Phi^{(i)}_{n,\mathbf{k}}\}_{i\in\{1,\cdots,N\};n\in\{0,\cdots,n_{\max}\};\mathbf{k}\in\mathcal{K}}$ ▷ Store and reuse adjoints
    for $j \in \mathcal{N}^{(i)}$ **do** ▷ Loop for i's neighbor
      $\partial E^{(ij)} \leftarrow 0, \partial E^{(ji)} \leftarrow 0$ ▷ Initialize pairwise derivatives
      compute $\left\{\frac{\partial\phi_{nlm}}{\partial\mathbf{r}}(\mathbf{r}^{(ij)})\right\}_{n,l,m\geq 0}$ ▷ Store and reuse derivatives
      for $n \in \{0,\cdots,n_{\max}\}$ **do**
        for $l \leq l_{\max}$ **do**
          $\partial E^{(ij)} \leftarrow \partial E^{(ij)} + \frac{\partial\phi_{nl0}}{\partial\mathbf{r}}\left(\mathbf{r}^{(ij)}\right)\Phi^{(i)}_{nl0,\left\{s^{(i)},s^{(j)}\right\}}$
          $+2\sum_{m>0}\mathrm{Re}\left[\frac{\partial\phi_{nlm}}{\partial\mathbf{r}}\left(\mathbf{r}^{(ij)}\right)\Phi^{(i)}_{nlm,\left\{s^{(i)},s^{(j)}\right\}}\right]$
          if $j \in \mathcal{L}^{(\xi)}$ **then** ▷ if atom-j is in proc-ξ
            $\partial E^{(ji)} \leftarrow \partial E^{(ji)} + (-)^{l+1}\left(\frac{\partial\phi_{nl0}}{\partial\mathbf{r}}\left(\mathbf{r}^{(ij)}\right)\Phi^{(j)}_{nl0,\left\{s^{(i)},s^{(j)}\right\}}\right.$
            $\left.+2\sum_{m>0}\mathrm{Re}\left[\frac{\partial\phi_{nlm}}{\partial\mathbf{r}}\left(\mathbf{r}^{(ij)}\right)\Phi^{(j)}_{nlm,\left\{s^{(i)},s^{(j)}\right\}}\right]\right)$
          end if
        end for
      end for
      $\mathbf{F}^{(ij)} \leftarrow \partial E^{(ij)} - \partial E^{(ji)}$ ▷ NOTE: $\partial E^{(ji)}$ is zero when $j \notin \mathcal{L}^{(\xi)}$
      $\mathbf{F}^{(i)} \leftarrow \mathbf{F}^{(i)} + \mathbf{F}^{(ij)}$ ▷ Add j's contribution
      $\mathbf{F}^{(j)} \leftarrow \mathbf{F}^{(j)} - \mathbf{F}^{(ij)}$ ▷ Newton's 3rd law
      $\boldsymbol{\sigma} \leftarrow \boldsymbol{\sigma} - \mathbf{r}^{(ij)} \otimes \mathbf{F}^{(ij)}$ ▷ Update stress tensor
    end for
  end for
end for
return $\{\mathbf{F}^{(i)}\}_{i\in\{1,\cdots,N\}}$ and $\boldsymbol{\sigma}$

3.2.4 Performance Benchmark

The execution speed of programs that implement the above-mentioned high-speed technology was benchmarked using AI Bridging Cloud Infrastructure (ABCI), which is constructed and operated by National Institute of Advanced Industrial Science and Technology (AIST). ABCI is one of the largest-scale Open AI Computing Infrastructure with commonly accepted hardware and software for High-Performance Computing (HPC). ABCI consists of 120 Compute Nodes (A) that form in total 960 NVIDIA A100 GPU accelerators, 1,088 Compute Nodes (V) that form in total 4,352 NVIDIA GPU V100 accelerators. Compute Node (A) has eight NVIDIA A100 GPU accelerators, two In-

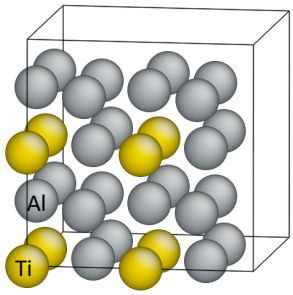


Figure 3.2.7: One of the sample structures, repeating base structure 2 by 2 by 2. It consists of 4 titanium and 28 aluminum atoms. Sample structures were generated by repeating the base structure of four atoms in three directions. Note that this is not the most stable state.

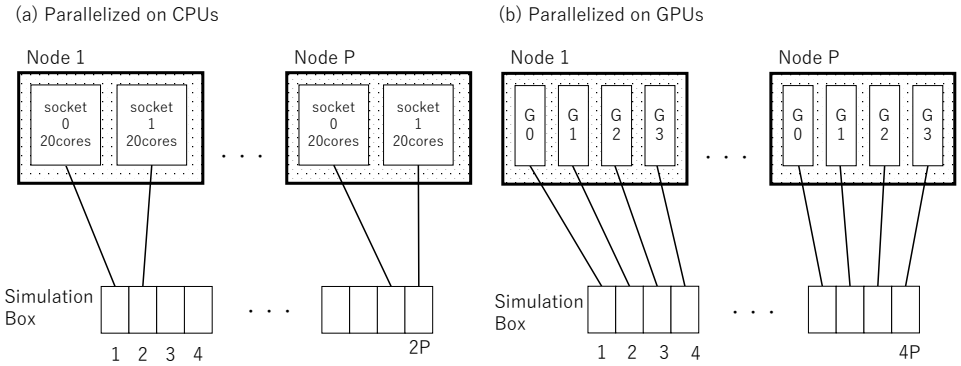
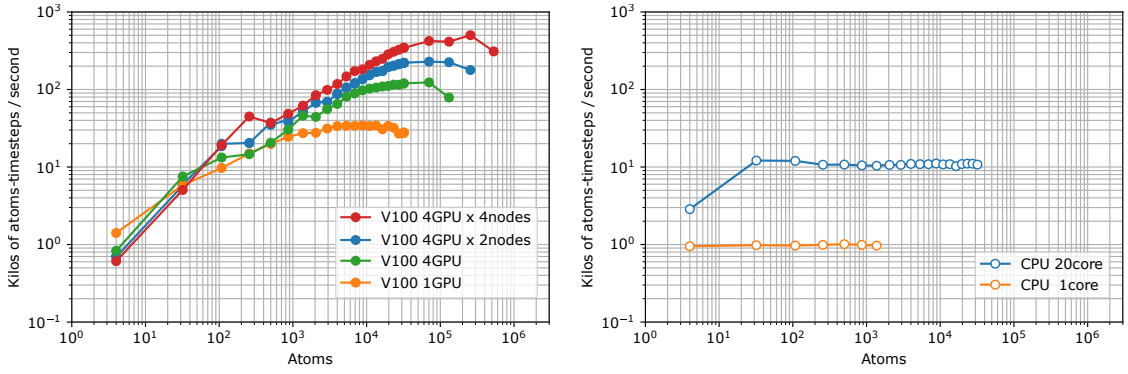


Figure 3.2.8: (a) The correspondence between decomposed simulation boxes and CPUs. The simulation box was decomposed into twice the number of nodes and each box was assigned to a CPU socket. (b) The correspondence between decomposed simulations and GPUs. The simulation box was decomposed into forth the number of nodes and each box was assigned to a GPU accelerator.

tel Xeon Platinum 8360Y, two NVMe SSDs, and four InfiniBand HDR (200Gbps each). Compute Node (V) has four NVIDIA V100 GPU accelerators, two Intel Xeon Gold 6148, one NVMe SSD, 384GiB memory, two InfiniBand EDR ports (100Gbps each). Here, the execution speed was measured by using up to 10 parallel nodes (V), that is, 40 GPUs or 20 CPUs.

The scaling simulations were performed for a binary alloy TiAl3, using a constant number of particles, volume, and energy (NVE ensemble). Sample structures are prepared by adding one or more translational copies of the base structure shown in Fig. 3.2.7 with 4, 32, 108, 256, 500, 864, 1,372, 2,048, 2,916, 4,000, 5,324, 6,912, 8,788, 10,976, 13,500, 16,384, 19,652, 23,328, 27,436, 32,000, 70,304, 131,072, 256,000, 530,604, and 1,048,576 atoms. An interatomic potential gtinv-197 was taken from the MLIP repository [29], which is one of the most accurate and computationally demanding MLPs in the repository. An MD timestep size of 1 fs was chosen, which is the default for the metal unit. The number of steps is 100, which is typically sufficient to measure averaged MD performance.



(a) (b)

Figure 3.3.1: The performance as a function of system size from 4 to 1,048,576 atoms. The vertical axis is system size by timesteps per second. A perfect scaling should be horizontal. (a) For GPUs with the spatial decomposition of up to 16 simulation boxes. (b) For CPUs using a single core and 20 cores on a CPU socket.

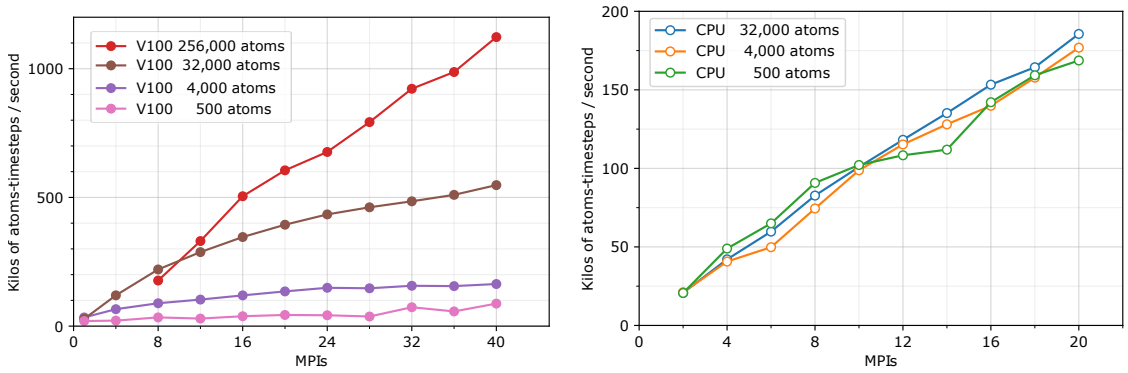
The simulation box was spatially decomposed and distributed to the processors over nodes as shown in Fig. 3.2.8. For CPUs, the simulation box was decomposed into twice the number of the nodes, and each decomposed box was assigned to a CPU socket with 20 cores. For GPUs, the simulation box was decomposed into forth the number of the nodes, and each decomposed box was assigned to a GPU accelerator. For example, with 32k atoms and 10 nodes, each CPU socket and GPU accelerator has 1.6k real atoms and 0.8k real atoms, respectively.

3.3 Results and Discussion

Two executables were benchmarked to determine the impact of parallelization on both CPUs and GPUs. These executables were generated by a implementation with two sets of compile options for the Kokkos library. The computational efficiency was assessed by a testing group of binary material selected and expanded from those in a preceding research [24].

Figure 3.3.1 shows the performance as a function of system size for GPUs and CPUs. The results for GPUs were calculated under spatial decomposition up to sixteen simulation boxes, assigning one GPU for each box. For GPUs, the efficiency in atom timestep per second improved as the system size increased. When the system was sufficiently large, the efficiency became approximately constant. The system size in which this rough flatness occurred was proportional to the number of parallelized GPUs. GPUs were not able to calculate more than 32,000, 131,072, 256,000, and 530,604 atoms at 1GPU, 4GPUs, 8GPUs, and 16GPUs, respectively, due to memory errors.

The results for CPUs were calculated on a processor socket with a single core or 20 core, without spatial decomposition. These CPU cores were parallelized using OpenMP



(a) (b)

Figure 3.3.2: The scaling measured as MD performance versus node count. Sample sizes were ranged from 500 to 256,000 atoms. (a) GPUs from 1 to 40 MPI processes. For 256,000 atoms, it was not possible to calculate with one or four GPUs due to insufficient memory capacity. (b) CPUs from 2 to 20 MPI processes.

threads in a CPU socket. Both calculation efficiencies were almost horizontal for the system size, which means CPUs scale perfectly in the observed system size. The benchmark was discontinued in the system size due to the execution time, when a trend was observed in calculations of up to 1,372 and 32,000 atoms for a single core and 20 cores, respectively.

Compared to the result of a single CPU core, one GPU was faster through all system sizes. However, compared to the result of CPU 20 cores on a socket, the result of GPU was less or equally efficient, especially less than 100 atoms. Therefore, it is not necessary to use GPUs for problems of 100 atoms or less.

In GPUs, the calculation efficiency improved as the number of atoms increases. This tendency agrees with the known property of GPU, where the number of repetitions of simple calculations increases, the calculation greatly accelerates. The memory errors in the large systems may have been caused by segmentation faults when inner variables of PolyMLP exhausted the GPU memory of 16GiB. The slight slowdown just before segmentation faults may be due to the increase in page fault. This type of memory limitation has been noted in a study on GPU acceleration of the ReaxFF reactive force-field [100]. This result has revealed that the spatial parallelization of PolyMLP can expand the upper limit on the number of atoms.

Figure 3.3.2 shows the performance benchmark of parallel scaling on 1 to 10 nodes. GPUs and CPUs can allocate 4 and 2 MPIs per node, respectively, thus the simulation box was decomposed into 4 to 40 and 2 to 20 small boxes for GPUs and CPUs, respectively. For GPUs, the higher the number of atoms, the greater the inclination to the number of MPI processes. On the other hand, for CPUs, the inclination to the number of MPI processes was roughly constant regardless of the number of atoms.

Unexpectedly, the efficiency of CPU calculations was improved by the spatial decom-

position using MPI even for a relatively small number of atoms about five hundred. For typical cases with simple interactions, for example, EAM or LJ, the efficiency for a small system would get worse when the system is spatially decomposed. Therefore, the acceleration of PolyMLP should be caused by the extraordinary complexity of PolyMLP. This would be very useful, for example, when performing a structural search with high-precision calculations for medium-sized structures of about 1000 atoms.

While GPUs contributed greatly to accelerating large-scale calculations, they were known to be less effective for small-scale calculations. Therefore, the efficient calculation of the small to medium scale was one of the problems. Initially, this study was planned for large-scale calculations, but surprisingly, it was found that CPUs are useful for small to medium-sized calculations. By switching between CPU and GPU, this implementation enables high-precision MD to be performed at high speed, from small to large systems.

3.4 Conclusion

The performant implementation of the linearized machine learning potential has been developed using either CPUs or GPUs by modifying the spatial decomposition framework. A significant improvement has been made for GPUs when a sufficient number of atoms exists in the system. For small systems, GPUs did not increase computational efficiency so much. On the other hand, CPUs linearly increased in the efficiency to the number of spatial decompositions even for a relatively small system of 500 atoms. This study showed the possibility of performing molecular dynamics calculations at high speed with high accuracy for systems of arbitrary size by selecting either CPUs or GPUs according to the size of the system. It is expected that this research will make it possible to study problems in materials science that involve a medium- to large-scale and high-precision computations, which was prohibitively expensive in computational cost before.

Chapter 4

General Conclusion

In this thesis, the predictive power of the polynomial machine learning potentials (PolyMLPs) has been investigated and the performant implementation of PolyMLP has been discussed. PolyMLP is a machine learning interatomic potential based on results of first-principles calculations and developed in the group to which the author belongs. This thesis demonstrates the high accuracy and transferability of PolyMLP by evaluating the global search of grain boundary structures and discussed how to accelerate PolyMLP to access realistic problems of materials science using graphics processing units (GPUs).

In Chapter 2, the author examined the predictive power of PolyMLP for grain boundary properties by systematically evaluating angle dependence of grain boundary energy and grain boundary structure for $\langle 100 \rangle$ STGBs, $\langle 110 \rangle$ STGBs, and $\langle 100 \rangle$ pure-twist grain boundaries in the face-centered-cubic elemental metals of Ag, Al, Au, Cu, Pd, and Pt. The optimum PolyMLPs for the calculation of grain boundaries were determined through the verification of grain boundary energies of test structures for all Pareto-optimal MLPs. It was confirmed that PolyMLPs with a certain accuracy or higher reproduced the grain boundary energies of first-principles calculations. The most stable grain boundary structures were obtained by globally searching in the microscopic degrees of freedom for each grain boundary structure model defined by the macroscopic degrees of freedom. In all elemental metals, grain boundary energies computed using PolyMLPs were almost the same as those computed by first-principles calculations. Therefore, the consistency indicates that PolyMLPs have high predictive power even for defect structures not contained in the training data set used to develop PolyMLPs.

In Chapter 3, the author discussed the development of the performant implementation of PolyMLP by using GPUs based on the spatial decomposition of the simulation box. First, the iterative processes in the potential computation are accelerated using parallel computation in a single processor unit. Second, the author modified the boundary condition and reactions between atoms near processor borders to calculate the interatomic force over parallelized processor units using spatial decomposition. Calculations with multiple GPUs in parallel were realized based on spatial decomposition, and the exe-

cution efficiency of PolyMLP was more than 1000 times faster than that on one CPU core. The author succeeded in calculating PolyMLP by parallelizing multiple GPUs for a large-scale problem in which a single GPU would run out of memory and calculation would be impossible. By using the Kokkos library, the author created a program that can be executed on a variety of architectures, from personal computers to supercomputers on CPUs or GPUs. The author's implementation based on spatial decomposition can compute problems in a broad range of system size at a realistic speed using highly accurate PolyMLPs.

In this thesis, not only the high accuracy of PolyMLPs was demonstrated, but also the acceleration of PolyMLPs was realized. The author's performant implementation will enable large-scale, high-precision calculations that were not practically possible before. For example, it can be applied to defect structure search using global optimization, and high-precision calculation of thermodynamic quantities using molecular dynamics and thermodynamic integration. The author expects that it will greatly accelerate materials science research.

Acknowledgement

First of all, the author would like to express his gratitude to Professor Isao Tanaka at Kyoto University for continuous guidance and encouragement throughout this work. He would like to express his gratitude to Professor Hiroyuki Nakamura and Professor Hiroshi Okuda at Kyoto University for critical reading of this thesis. He would like to express his thanks to Associate Professor Atsuto Seko at Kyoto University for helpful discussions and advice throughout this work. He would like to thank Assistant Professor Hiroyuki Hayashi at Kyoto University for continuous encouragement.

The author is grateful to Mr. Shinohara for fruitful discussions and his kind cooperation through the development of the performant implementation. The author would like to express his thanks to the colleagues of Professor Tanaka's group for continuous encouragement and helpful discussions throughout this work.

Bibliography

[1] P. Hohenberg and W. Kohn, Phys. Rev. **136**, B864 (1964).

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

[3] S. Plimpton, J. Comput. Phys. **117**, 1 (1995).

[4] A. F. Voter, Physical Review B **57**, R13985 (1998).

[5] A. P. Bartók, M. C. Payne, R. Kondor, and G. Csányi, Phys. Rev. Lett. **104**, 136403 (2010).

[6] W. J. Szlachta, A. P. Bartók, and G. Csányi, Phys. Rev. B **90**, 104108 (2014).

[7] A. P. Bartók, J. Kermode, N. Bernstein, and G. Csányi, Phys. Rev. X **8**, 041048 (2018).

[8] Z. Li, J. R. Kermode, and A. De Vita, Phys. Rev. Lett. **114**, 096405 (2015).

[9] A. Glielmo, P. Sollich, and A. De Vita, Phys. Rev. B **95**, 214302 (2017).

[10] S. Lorenz, A. Groß, and M. Scheffler, Chem. Phys. Lett. **395**, 210 (2004).

[11] J. Behler and M. Parrinello, Phys. Rev. Lett. **98**, 146401 (2007).

[12] J. Behler, J. Chem. Phys. **134**, 074106 (2011).

[13] J. Han, L. Zhang, R. Car, and W. E, Commun. Comput. Phys. **23**, 629 (2018).

[14] N. Artrith and A. Urban, Comput. Mater. Sci. **114**, 135 (2016).

[15] N. Artrith, A. Urban, and G. Ceder, Phys. Rev. B **96**, 014112 (2017).

[16] A. Seko, A. Togo, and I. Tanaka, Phys. Rev. B **99**, 214108 (2019).

[17] A. Thompson, L. Swiler, C. Trott, S. Foiles, and G. Tucker, J. Comput. Phys. **285**, 316 (2015).

[18] R. Drautz, Phys. Rev. B **99**, 014104 (2019).

[19] R. Gayatri, S. Moore, E. Weinberg, N. Lubbers, S. Anderson, J. Deslippe, D. Perez, and A. P. Thompson, "Rapid exploration of optimization strategies on advanced architectures using TestSNAP and LAMMPS," (2020), arXiv:2011.12875 .

[20] K. Nguyen-Cong, J. T. Willman, S. G. Moore, A. B. Belonoshko, R. Gayatri, E. Weinberg, M. A. Wood, M. Gov, A. P. Thompson, and I. I. Oleynik, , 1 (2021).

[21] M. A. Wood and A. P. Thompson, J. Chem. Phys. **148**, 241721 (2018).

[22] G. Dusson, M. Bachmayr, G. Csanyi, R. Drautz, S. Etter, C. van der Oord, and C. Ortner, (2019), arXiv:1911.03550 .

[23] A. Seko, A. Togo, and I. Tanaka, Phys. Rev. B **99**, 214108 (2019).

[24] A. Seko, Phys. Rev. B **102**, 174104 (2020).

[25] A. P. Bartók, R. Kondor, and G. Csányi, Phys. Rev. B **87**, 184115 (2013).

[26] I. R. Kondor, *Group Theoretical Methods in Machine Learning* (Columbia University, 2008).

[27] P. J. Steinhardt, D. R. Nelson, and M. Ronchetti, Phys. Rev. B **28**, 784 (1983).

[28] A. V. Shapeev, Multiscale Model. Simul. **14**, 1153 (2016).

[29] A. Seko, arXiv:2007.14206 (2020), Machine Learning Potential Repository at Kyoto University, https://sekocha.github.io/repository/index-e.html.

[30] A. P. Sutton and R. W. Balluffi, *Interfaces in crystalline materials* (Oxford: Clarendon Press, 1995).

[31] J. Bishop and R. Hill, London, Edinburgh, Dublin Philos. Mag. J. Sci. **42**, 414 (1951).

[32] E. Kröner, Acta Metall. **9**, 155 (1961).

[33] Y. Mishin, M. Asta, and J. Li, Acta Mater. **58**, 1117 (2010).

[34] J. R. Greer and J. T. De Hosson, Prog. Mater. Sci. **56**, 654 (2011).

[35] G. C. Hasson and C. Goux, Scr. Metall. **5**, 889 (1971).

[36] D. Wolf, Scr. Metall. **23**, 1713 (1989).

[37] K. L. Merkle and D. Wolf, MRS Bull. **15**, 42 (1990).

[38] M. W. Finnis and J. E. Sinclair, Philos. Mag. A **50**, 45 (1984).

[39] M. S. Daw and M. I. Baskes, Phys. Rev. B **29**, 6443 (1984).

[40] J. W. Cahn, Y. Mishin, and A. Suzuki, Philos. Mag. **86**, 3965 (2006).

[41] S. von Alfthan, K. Kaski, and A. P. Sutton, Phys. Rev. B **74**, 134101 (2006).

[42] J. A. Brown and Y. Mishin, Phys. Rev. B **76**, 134118 (2007).

[43] M. Dao, L. Lu, R. J. Asaro, J. T. De Hosson, and E. Ma, Acta Mater. **55**, 4041 (2007).

[44] M. A. Tschopp and D. L. McDowell, Philos. Mag. **87**, 3147 (2007).

[45] D. L. Olmsted, S. M. Foiles, and E. A. Holm, Acta Mater. **57**, 3694 (2009).

[46] M. D. Sangid, H. Sehitoglu, H. J. Maier, and T. Niendorf, Mater. Sci. Eng. A **527**, 7115 (2010).

[47] M. A. Tschopp, K. N. Solanki, F. Gao, X. Sun, M. A. Khaleel, and M. F. Horstemeyer, Phys. Rev. B **85**, 064108 (2012).

[48] M. A. Tschopp, S. P. Coleman, and D. L. McDowell, Integr. Mater. Manuf. Innov. **4**, 176 (2015).

[49] S. Kiyohara, H. Oda, T. Miyata, and T. Mizoguchi, Sci. Adv. **2**, e1600746 (2016).

[50] G. S. Rohrer, E. A. Holm, A. D. Rollett, S. M. Foiles, J. Li, and D. L. Olmsted, Acta Mater. **58**, 5063 (2010).

[51] E. A. Holm, G. S. Rohrer, S. M. Foiles, A. D. Rollett, H. M. Miller, and D. L. Olmsted, Acta Mater. **59**, 5250 (2011).

[52] A. P. Bartók, J. Kermode, N. Bernstein, and G. Csányi, Phys. Rev. X **8**, 041048 (2018).

[53] A. Seko, A. Takahashi, and I. Tanaka, Phys. Rev. B **90**, 024101 (2014).

[54] A. Seko, A. Takahashi, and I. Tanaka, Phys. Rev. B **92**, 054113 (2015).

[55] A. Takahashi, A. Seko, and I. Tanaka, Phys. Rev. Mater. **1**, 063801 (2017).

[56] A. Thompson, L. Swiler, C. Trott, S. Foiles, and G. Tucker, J. Comput. Phys. **285**, 316 (2015).

[57] C. Chen, Z. Deng, R. Tran, H. Tang, I.-H. Chu, and S. P. Ong, Phys. Rev. Mater. **1**, 043603 (2017).

[58] A. V. Shapeev, Multiscale Model. Simul. **14**, 1153 (2016).

[59] V. L. Deringer, C. J. Pickard, and G. Csányi, Phys. Rev. Lett. **120**, 156001 (2018).

[60] E. V. Podryabinkin, E. V. Tikhonov, A. V. Shapeev, and A. R. Oganov, Phys. Rev. B **99**, 064114 (2019).

[61] K. Gubaev, E. V. Podryabinkin, G. L. Hart, and A. V. Shapeev, Comput. Mater. Sci. **156**, 148 (2019).

[62] T. Mueller, A. Hernandez, and C. Wang, J. Chem. Phys. **152**, 050902 (2020).

[63] S. P. Ong, W. D. Richards, A. Jain, G. Hautier, M. Kocher, S. Cholia, D. Gunter, V. L. Chevrier, K. A. Persson, and G. Ceder, Comput. Mater. Sci. **68**, 314 (2013).

[64] P. E. Blöchl, Phys. Rev. B **50**, 17953 (1994).

[65] J. P. Perdew, K. Burke, and M. Ernzerhof, Phys. Rev. Lett. **77**, 3865 (1996).

[66] G. Kresse and J. Hafner, Phys. Rev. B **47**, 558 (1993).

[67] G. Kresse and J. Furthmüller, Phys. Rev. B **54**, 11169 (1996).

[68] G. Kresse and D. Joubert, Phys. Rev. B **59**, 1758 (1999).

[69] The computational time is estimated using a single core of Intel Xeon E5-2695 v4 (2.10GHz).

[70] G. J. Ackland, G. Tichy, V. Vitek, and M. W. Finnis, Philos. Mag. A **56**, 735 (1987).

[71] P. L. Williams, Y. Mishin, and J. C. Hamilton, Model. Simul. Mater. Sci. Eng. **14**, 817 (2006).

[72] Y. Mishin, D. Farkas, M. J. Mehl, and D. A. Papaconstantopoulos, Phys. Rev. B **59**, 3393 (1999).

[73] X. W. Zhou, R. A. Johnson, and H. N. G. Wadley, Phys. Rev. B **69**, 144113 (2004).

[74] Y. Mishin, M. J. Mehl, D. A. Papaconstantopoulos, A. F. Voter, and J. D. Kress, Phys. Rev. B **63**, 224106 (2001).

[75] The computational cost of the EAM potential for Al is estimated to be 2×10^{-3} ms/atom/step.

[76] J. Li, Model. Simul. Mater. Sci. Eng. **11**, 173 (2003).

[77] W. T. Read and W. Shockley, Phys. Rev. **78**, 275 (1950).

[78] G. Simmons and H. Wang, *Single Crystal Elastic Constants and Calculated Aggregate Properties: A HANDBOOK*, 2nd ed. (The Massachusetts Institute of Technology, 1971).

[79] Y. Lysogorskiy, C. van der Oord, A. Bochkarev, S. Menon, M. Rinaldi, T. Hammerschmidt, M. Mrovec, A. Thompson, G. Csányi, C. Ortner, and R. Drautz, npj Comput. Mater. 2021 71 **7**, 1 (2021).

[80] A. Singraber, T. Morawietz, J. Behler, and C. Dellago, J. Chem. Theory Comput. **15**, 3075 (2019).

[81] A. Singraber, J. Behler, and C. Dellago, J. Chem. Theory Comput. **15**, 1827 (2019).

[82] D. Dickel, D. Francis, and C. Barrett, Comput. Mater. Sci. **171**, 109157 (2020).

[83] D. Dickel, M. Nitol, and C. Barrett, Comput. Mater. Sci. **196**, 110481 (2021).

[84] M. S. Nitol, D. E. Dickel, and C. D. Barrett, Comput. Mater. Sci. **188**, 110207 (2021).

[85] H. Yanxon, D. Zagaceta, B. C. Wood, and Q. Zhu, J. Chem. Phys. **153**, 054118 (2020).

[86] M. A. Cusentino, M. A. Wood, and A. P. Thompson, J. Phys. Chem. A **124**, 5456 (2020), arXiv:2003.11570 .

[87] H. Wang, L. Zhang, J. Han, and W. E, Comput. Phys. Commun. **228**, 178 (2018), arXiv:1712.03641 .

[88] M. Wen and E. B. Tadmor, Phys. Rev. B **100**, 195419 (2019), arXiv:1909.10134 .

[89] M. Wen and E. B. Tadmor, npj Comput. Mater. 2020 61 **6**, 1 (2020).

[90] R. Lot, F. Pellegrini, Y. Shaidu, and E. Küçükbenli, Comput. Phys. Commun. **256**, 107402 (2020), arXiv:1907.03055 .

[91] J. S. Smith, B. Nebgen, N. Mathew, J. Chen, N. Lubbers, L. Burakovsky, S. Tretiak, H. A. Nam, T. Germann, S. Fensin, and K. Barros, Nat. Commun. 2021 121 **12**, 1 (2021), arXiv:2003.04934 .

[92] V. Botu and R. Ramprasad, Int. J. Quantum Chem. **115**, 1074 (2015).

[93] H. Mori and T. Ozaki, Phys. Rev. Mater. **4**, 040601 (2020).

[94] G. P. Pun, R. Batra, R. Ramprasad, and Y. Mishin, Nat. Commun. 2019 101 **10**, 1 (2019), arXiv:1808.01696 .

[95] H. Carter Edwards, C. R. Trott, and D. Sunderland, J. Parallel Distrib. Comput. **74**, 3202 (2014).

[96] C. Trott, L. Berger-Vergiat, D. Poliakoff, S. Rajamanickam, D. Lebrun-Grandie, J. Madsen, N. Al Awar, M. Gligoric, G. Shipman, and G. Womeldorff, Comput. Sci. Eng. **23**, 10 (2021).

[97] C. R. Trott, D. Lebrun-Grandie, D. Arndt, J. Ciesko, V. Dang, N. Ellingwood, R. Gayatri, E. Harvey, D. S. Hollman, D. Ibanez, N. Liber, J. Madsen, J. Miles, D. Poliakoff, A. Powell, S. Rajamanickam, M. Simberg, D. Sunderland, B. Turcksin, and J. Wilke, IEEE Trans. Parallel Distrib. Syst. **33**, 805 (2022).

[98] T. Limpanuparb and J. Milthorpe, *Associated Legendre Polynomials and Spherical Harmonics Computation for Chemistry Applications*, Tech. Rep. (2014) arXiv:1410.1748v1 .

[99] D. A. Varshalovich, A. N. Moskalev, and V. K. Khersonskii, *Quantum Theory of Angular Momentum* (WORLD SCIENTIFIC, 1988).

[100] T. P. Senftle, S. Hong, M. M. Islam, S. B. Kylasa, Y. Zheng, Y. K. Shin, C. Junkermeier, R. Engel-Herbert, M. J. Janik, H. M. Aktulga, T. Verstraelen, A. Grama, and A. C. T. van Duin, npj Comput. Mater. 2016 21 **2**, 1 (2016).