Development of Interatomic Potentials with Uncertainty Quantification: Applications to Two-dimensional Materials

A DISSERTATION
SUBMITTED TO THE FACULTY OF THE GRADUATE SCHOOL
OF THE UNIVERSITY OF MINNESOTA
BY

Mingjian Wen

IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Ellad B. Tadmor, Advisor

July, 2019

Acknowledgements

I oftentimes just cannot help thinking of myself being a model not too different from an interatomic potential in the sense that both of us gain more insights by learning from a set of training data. While the largest portion of training data for my interatomic potentials come from expensive first-principles calculations, those for me in my Ph.D. years are provided generously for free by my advisor Prof. Ellad B. Tadmor. Not to mention his professional mentorship, tremendous patience, and continuous support on my research, I am significantly influenced by and truly adore his research appetite to address problems that would benefit a whole research community. Another large source of training data for me is Prof. Ryan S. Elliott. His rigorous mathematical thinking and extraordinary expertise in computer science among others are instrumental and invaluable to me. Hopefully, I have become an active learning–enabled model that can generate data by myself and for myself in my future scientific endeavors. Many thanks to them.

I would like to thank the other two members of my thesis committee Prof. Perry H. Leo and Prof. George Karypis for their comments and suggestions for the thesis. This thesis would not be possible without the efforts from the collaborators. Specifically, I would like to thank Prof. Efthimios Kaxiras for hosting me as a visiting student in his group and his encouragement to explore machine learning techniques to develop potentials, and Prof. James P. Sethna and Prof. Petr Plecháč for the collaborations and discussions on various projects. The members of the OpenKIM project and the MURI project should also be thanked.

I am also indebted to the members of Tadmor's group—Moon-Ki Choi, Jiadi Fan, Daniel S. Karls, Neil Khadilar, Subrahmanyam Pattamatta, Min Shi, Amit Singh, Stephen M. Whalen, and Kuan Zhang—for useful discussions and inspirations at our group meetings. Of course, I am grateful to my peer graduate students—Fan Feng, Hanlin Gu, Gunjan Pahlani, Ariel Ibarra Pino, Krishanu Sen, Andrew Vechart, and Eduardo Vitral—for general discussions on various topics in mechanics, materials, and many more.

Finally, my deepest gratitude goes to my family for their love, encouragement, understanding, and unconditional support.

To my wife *Min He*

Abstract

Atomistic simulation is a powerful computational tool to investigate materials on the microscopic scale and is widely employed to study a large variety of problems in science and engineering. Empirical interatomic potentials have proven to be an indispensable part of atomistic simulation due to their unrivaled computational efficiency in describing the interactions between atoms, which produce the forces governing atomic motion and deformation. Atomistic simulation with interatomic potentials, however, has historically been viewed as a tool limited to provide only qualitative insight. A key reason is that in such simulations there are many sources of uncertainty that are difficult to quantify, thus failing to give confidence interval on the obtained results. This thesis presents my research work on the development of interatomic potentials with the ability to quantify the uncertainty in simulation results. The methods to train interatomic potentials and quantify the uncertainty are demonstrated via two-dimensional materials and heterostructures throughout this thesis, whose low-dimensional nature makes them distinct from their three-dimensional counterparts in many aspects. Both physics-based and machine learning interatomic potentials are developed for MoS_2 and multilayer graphene structures. The new potentials accurately model the interactions in these systems, reproducing a number of structural, energetic, elastic, and thermal properties obtained from first-principles calculations and experiments. For physics-based potentials, a method based on Fisher information theory is used to analyze the parametric sensitivity and the uncertainty in material properties obtained from phase average. We show that the dropout technique can be applied to train neural network potentials and demonstrate how to obtain the predictions and the associated uncertainties of material properties practically and efficiently from such potentials. Putting all these ingredients of my research work together, we create an open-source fitting framework to train interatomic potentials and hope it can make the development and deployment of interatomic potentials easier and less error prone for other researchers.

Contents

List of Tables

List of Figures

Notations

$\delta(\cdot)$	Dirac delta
$\delta_{\cdot,\cdot}$	Kronecker delta
s^*	complex conjugate of a scalar s
$\boldsymbol{M}^\dagger$	complex conjugate transpose of a matrix $\boldsymbol{M}$
$\boldsymbol{M}^{\mathrm{T}}$	transpose of a matrix $\boldsymbol{M}$
$\mathbb{R}^n$	real space of n dimensions
$\mathcal{V}$	total interatomic potential energy of an atomic system
$\mathcal{L}(\boldsymbol{\theta})$	loss function with parameters $\boldsymbol{\theta}$
$\boldsymbol{x}$	a column vector of an observation of the input
$\boldsymbol{y}$	a column vector of an observation of the output
$\boldsymbol{X}$	a matrix of N inputs in a data set, $\boldsymbol{X} = [\boldsymbol{x}_1^{\mathrm{T}}; \boldsymbol{x}_2^{\mathrm{T}}; \cdots ; \boldsymbol{x}_N^{\mathrm{T}}]$
$\boldsymbol{Y}$	a matrix of N outputs in a data set, $\boldsymbol{Y} = [\boldsymbol{y}_1^{\mathrm{T}}; \boldsymbol{y}_2^{\mathrm{T}}; \cdots ; \boldsymbol{y}_N^{\mathrm{T}}]$
$\mathcal{D}$	a data set comprised of N observations, $\mathcal{D} = (\boldsymbol{X}, \boldsymbol{Y})$
$p(\boldsymbol{\theta})$	probability distribution of $\boldsymbol{\theta}$
$p(\boldsymbol{x} \mid \boldsymbol{y})$	conditional probability distribution of $\boldsymbol{x}$ given $\boldsymbol{y}$

List of Abbreviations

1D	one-dimensional
2D	two-dimensional
3D	three-dimensional
4D	four-dimensional
AIMD	*ab initio* molecular dynamics
AIREBO	adaptive intermolecular reactive empirical bond order
API	application programming interface
BFGS	Broyden–Fletcher–Goldfarb–Shanno minimization algorithm
BOA	Born–Oppenheimer approximation
DFT	density functional theory
DNNIP	dropout neural network interatomic potential
DRIP	dihedral-angle-corrected registry-dependent interlayer potential
EAM	embedded atom method
FIM	Fisher information matrix
GAP	Gaussian approximation potential
GGA	generalized gradient approximation
GP	Gaussian process
IP	interatomic potential
KC	Kolmogorov–Cresp
KIM	knowledgebase of interatomic models
KL	Kullback–Leibler
KLIFF	KIM-based learning-integrated fitting framework
KRR	kernel ridge regression

LAMMPS	large-scale atomic/molecular massively parallel simulator
LCBOP	long-range carbon bond order potential
LDA	local density approximation
LJ	Lennard–Jones
LM	Levenberg–Marquardt
MC	Monte Carlo
MCMC	Markov chain Monte Carlo
MD	molecular dynamics
MEAM	modified embedded atom method
NN	neural network
NNIP	neural network interatomic potential
OpenKIM	open knowledgebase of interatomic models
ReaxFF	reactivate force field
REBO	reactive empirical bond order
RMSE	root-mean-square error
SGD	stochastic gradient descent
SW	Stillinger–Weber
TB	tight-binding
TMCMC	transitional Markov chain Monte Carlo
UQ+P	uncertainty quantification and propagation
VASP	Vienna ab initio simulation package
vdW	van der Waals

Chapter 1

Introduction

A review of the history of human civilization makes it clear that major advancements in how we live, work, travel, communicate, etc. are more than often driven by fundamental development in materials and the relevant technological processes, manifested by the naming of ages of civilizations—from the Stone Age through the Bronze and Iron Ages to the Silicon Age (Information Age) [1–3]. It is believed now we may be on the edge of another great materials revolution powered by nanotechnology, preluding the "Age of Nano"[1] [4, 5]. Nanotechnology[2] refers to the study, manipulation, and engineering of materials with at least one dimension sized from 1 to 100 nanometers [6].

The study of nano materials requires a resolution on the atomistic level. Experimentally, this is impossible until the advent of scanning tunneling microscope (STM) [7] and atomic force microscope (AFM) [8], which enable us to see and control individual atoms, respectively. Even with these tools, it can still be difficult and/or expensive to conduct materials research in experimental settings [9, 10]. As an alternative, atomistic simulation (e.g. Monte Carlo (MC), molecular dynamics (MD), and lattice dynamics methods) is a powerful computational tool to investigate materials problems on the nano scale and is widely employed in academia and industry to study a large variety

[1] Considering the rise of artificial intelligence (AI) in the last decade, especially deep learning, and its promising applications in many scientific, engineering, and technological areas, some may argue that the next age should be named the "Age of AI" instead of the "Age of Nano". Maybe, this can be answered by an AI-enabled robot made of nano materials in the future.

[2] The ideas and concepts behind nanotechnology was first brought up by physicist Richard Feynman at an American Physical Society meeting in 1959, long before the term nanotechnology was coined by Norio Taniguchi at the University of Tokyo in his explorations of ultraprecision machining in 1974 [6].

of problems, ranging from phase transitions [11], chemical reaction processes [12], to protein foldings [13].

At the core of any atomistic simulation lies a description of the interactions between atoms, which produces the forces governing atomic motion and deformation. First-principles approaches that involve solving the Schrödinger equation are most accurate, but due to hardware and algorithmic limitations, these approaches are typically limited to studying small molecular systems and crystalline materials characterized by compact unit cells with an upper limit on the number of atoms in the range of $\sim 10^3$. This difficulty can be overcome with the aid of empirical interatomic potentials (IPs) (also known as force fields). IPs strive to capture the influence of the electrons on the nuclei in an effective manner without explicitly simulating them, so they are computationally far less expensive than first-principles methods and can therefore be used to compute static and dynamic properties that are inaccessible to quantum calculations. Historically, atomistic simulation with IPs is viewed as a tool limited to provide only qualitative insight. A key reason is that in such simulations there are many sources of uncertainty that are difficult to quantify, thus failing to give confidence interval on the result of a simulation [14].

Since the discovery of graphene [15], two-dimensional (2D) materials and heterostructures have been shown to possess remarkable electronic, mechanical, thermal, and optical properties that their three-dimensional (3D) bulk counterparts do not have. Such unprecedented properties open the door to a host of innovative nanotechnology applications, such as semiconductors, ultrasensitive sensors, and medical devices to name a few. To accelerate this revolution, highly accurate atomistic simulation techniques are required to better understand the basic science of 2D materials and heterostructures, systematically design new devices, and improve manufacturing processes.

The main goal of this thesis is to develop accurate IPs that have the ability to quantify the uncertainty in atomistic simulations and then apply these IPs to study 2D materials and heterostructures.

1.1 Atomistic modelling

Quantum mechanics is a fundamental theory in physics which describes nature at the smallest scales of energy levels of atoms and subatomic particles. In the quantum mechanics treatment, both the nuclei (protons and neutrons) and the orbiting electrons are directly modeled. The mass of an individual proton/neutron is roughly 1836 times of that of an individual electron, and thus the nuclei can be modeled as classical Newtonian particles, leaving the electrons to be considered as wave-like particles. The wave-like behavior of a total number of N_{el} electrons can be represented using a *wave function* $\chi(\boldsymbol{r}_1, \ldots, \boldsymbol{r}_{N_{\text{el}}}, t)$, where $\boldsymbol{r}_n$ $(n = 1, 2, \ldots N_{\text{el}})$ is the spacial position of electron n and t denotes time. The evolution of this wave function is governed by the *time-dependent* Schrödinger equation [16]

$$\left(u(\boldsymbol{r}_1, \ldots, \boldsymbol{r}_{N_{\text{el}}}, t) - \frac{\hbar^2}{2m_{\text{el}}} \sum_{n=1}^{N_{\text{el}}} \nabla_n^2 \right) \chi(\boldsymbol{r}_1, \ldots, \boldsymbol{r}_{N_{\text{el}}}, t) = i\hbar \frac{\partial}{\partial t} \chi(\boldsymbol{r}_1, \ldots, \boldsymbol{r}_{N_{\text{el}}}, t), \quad (1.1)$$

where $u(\boldsymbol{r}_1, \ldots, \boldsymbol{r}_{N_{\text{el}}}, t)$ is an external potential energy filed the system of electrons is subjected to, ∇_n^2 is the Laplacian operator applied to electron n, m_{el} is the mass of an electron, $\hbar$ is the Plank's constant, and i is the imaginary unit.

As mentioned above, the mass of an individual proton/neutron is far greater than that of an individual electron, so the magnitude of the acceleration of the nucleus due to the Coulomb interaction is smaller than that of the electron. Consequently, we can assume that as the nuclei move, the electrons find the appropriate ground state configuration by responding instantaneously to the gradual evolution of the nuclei positions. This assumption is known as the Born–Oppenheimer approximation (BOA) [17]. When modeling materials, the primary origin of the external potential is the interaction between the electrons and the nuclei in the solid. Therefore, a mathematical consequence of the BOA assumption is that the external potential becomes time-independent, $u(\boldsymbol{r}_1, , \ldots, \boldsymbol{r}_{N_{\text{el}}})$. In this case, equation (1.1) can be solved by the method of separation of variables, and we look for solutions of the wave function in the form

$$\chi(\boldsymbol{r}_1, \ldots, \boldsymbol{r}_{N_{\text{el}}}, t) = \psi(\boldsymbol{r}_1, \ldots, \boldsymbol{r}_{N_{\text{el}}}) \varphi(t), \quad (1.2)$$

where ψ is a function of positions of electrons only and φ is a function of time only. Inserting this into equation (1.1) and rearranging the terms, we get

$$i\hbar\frac{1}{\varphi}\frac{\mathrm{d}\varphi}{\mathrm{d}t} = -\frac{\hbar^2}{2m_{\mathrm{el}}}\frac{1}{\psi}\sum_{n=1}^{N_{\mathrm{el}}}\nabla_n^2\psi + u, \tag{1.3}$$

of which the left-hand side only depends on time t whereas the right-hand side only depends on the positions of elections. This is valid only when both sides of the equation are equal to a constant; otherwise, for example, by varying t, the left side changes without touching the right side, and the equality can not hold any longer. Let's call this constant ϵ. The time-dependent part can be easily solved as

$$\varphi(t) = \exp(-i\omega t), \tag{1.4}$$

where $\omega = \epsilon/\hbar$, which is also the reason why ϵ is treated as energy. The positional part of the wave function is known as the *time-independent* Schrödinger equation [16]:

$$\left(u(\boldsymbol{r}_1,\ldots,\boldsymbol{r}_{N_{\mathrm{el}}}) - \frac{\hbar^2}{2m_{\mathrm{el}}}\sum_{n=1}^{N_{\mathrm{el}}}\nabla_n^2\right)\psi(\boldsymbol{r}_1,\ldots,\boldsymbol{r}_{N_{\mathrm{el}}}) = \epsilon\psi(\boldsymbol{r}_1,\ldots,\boldsymbol{r}_{N_{\mathrm{el}}}). \tag{1.5}$$

The objective then is to solve this time-independent Schrödinger equation for the electronic wave function, $\psi(\boldsymbol{r}_1,\ldots,\boldsymbol{r}_{N_{\mathrm{el}}})$. From the wave function, we can, in principle, know everything about the materials system, including the mechanism of the bonding formation when atoms are brought together and the motion of atoms when forces are applied to the system. We refer computational schema attempting to solve either equation (1.1) or equation (1.5) as *first-principles* or *ab inito* methods.

Unfortunately, it turns out that equation (1.5) can only be solved exactly for the case of a hydrogen atom with a single electron. To solve the hydrogen molecule (two protons and two electrons) problem, we need to make a number of further assumptions, let along materials made up of thousands of or even millions of atoms. To tackle this, a number of approximate methods have been put forward, among which the density functional theory (DFT) has become one of the most popular and versatile methods available in computational chemistry, condensed-matter physics, and materials science. The central idea of DFT is to recast the many-body problem involving multiple electrons to an

equivalent, but greatly simplified, problem of solving for a single electron wave function like the hydrogen atom problem, in which the electron density $\rho(\boldsymbol{r})$ is the fundamental quantity to be solved for. There are essentially three "steps" in solving equation (1.5) using DFT [17]: (i) replace the many-electron wave function by an electron density with an effective external potential; (ii) replace the multi-electron problem with an equivalent single-electron system; and (iii) solve for the wave function for the greatly simplified single-electron case.

DFT is actually an exact theory without any approximations being made, provided the *exchange-correlation* term in the effective potential is known exactly. However, the exchange-correlation term is an energy contribution difficult to compute, and typically approximations have to be made to carry out the computation. The local density approximation (LDA), and generalized gradient approximation (GGA), as well as the hybrid of them are widely used functionals to approximate the exchange-correlation term. Another approximation widely used in DFT is that instead of solving the Schrödinger equation for all electrons, we solve for the electronic structure of the valence electrons only. The argument behind this is that the core electrons are tightly bound to remain close to the nucleus and thus do not participate in any significant way in the bonding process. Of course, the effects of core electrons cannot be neglected completely. To this end, we replace the Coulomb potential in the DFT formulation with an effective *pseudopotential*, which, essentially, provides an overall description for all the core electrons, but individual description for each valence electron. The use of pseudopotential greatly reduces the degrees of freedom.

Although first-principles approaches such as DFT that involve solving the Schrödinger equation are very accurate, due to hardware and algorithmic limitations, these approaches are typically limited to studying small molecular systems and crystalline materials characterized by compact unit cells with an upper limit on the number of atoms in the range of $\sim 10^3$. To proceed, we make further approximations. The tight-binding (TB) method approximates the wave functions of a system by superposing the wave functions for isolated atoms located at each atomic site. Many integrals in DFT are parametrized into simple analytic forms. In spite of these simplifications, TB method still requires computationally intensive matrix inversion.

To step further boldly, empirical interatomic potentials (IPs) (also known as force

fields) treats atoms as classical particles without explicitly modeling the degrees of freedom of electrons. The BOA allows the electrons to be replaced by an effective potential [17], although the exact form of the effective potential is not known. So, the central task in developing IPs is to design a function—taking the positions of atoms[3] and their species as the arguments—to accurately approximate the energy of the electrons. Mathematically, we are seeking for IP function of the form

$$\mathcal{V} = \mathcal{V}(\boldsymbol{r}_1, \ldots, \boldsymbol{r}_{N_\mathrm{a}}, Z_1, \ldots, Z_{N_\mathrm{a}}; \boldsymbol{\theta}), \tag{1.6}$$

where $\boldsymbol{r}_1, \ldots, \boldsymbol{r}_{N_\mathrm{a}}$ and $Z_1, \ldots, Z_{N_\mathrm{a}}$ are the positions and atomic species of a system of N_a atoms, respectively, and $\boldsymbol{\theta}$ is a set of fitting parameters associated with the IP mathematical form. IPs are computationally far less costly than first-principles methods and can therefore be used to compute static and dynamic properties that are inaccessible to quantum calculations. IPs are the central topic of this thesis, and we will discuss more about the laws of physics an IP has to obey and how to design IPs in chapters 2 and 3.

1.2 Two-dimensional materials and heterostructures

The methods to train interatomic potentials (IPs) and quantify their uncertainty developed in this thesis are general and can be applied to any materials system. The example materials systems studied throughout this thesis are two-dimensional (2D) materials and heterostructures, whose low-dimensional nature makes them distinct from their three-dimensional (3D) counterparts in many aspects. In this section, we give a brief review of 2D materials and heterostructures to provide some background information.

Graphene is a one-atom thick crystalline sheet consisting of carbon atoms arranged in a hexagonal lattice (see figure 1.1). Each unit cell consists of two carbon atoms, and the distance between neighboring atoms is 1.42 Å (i.e. the lattice parameter is $a = 2.46$ Å). In graphene, one s orbital and two p orbitals of each atom hybridize to form three equivalent sp^2 orbitals, arranging themselves in a triangular planar configuration. The remaining unhybridized p orbital arranges itself to be as far apart from the sp^2 orbitals

[3] Hereafter, "atom" refers to a classical particle without distinguishing the protons, neutrons, and electrons, unless otherwise stated.

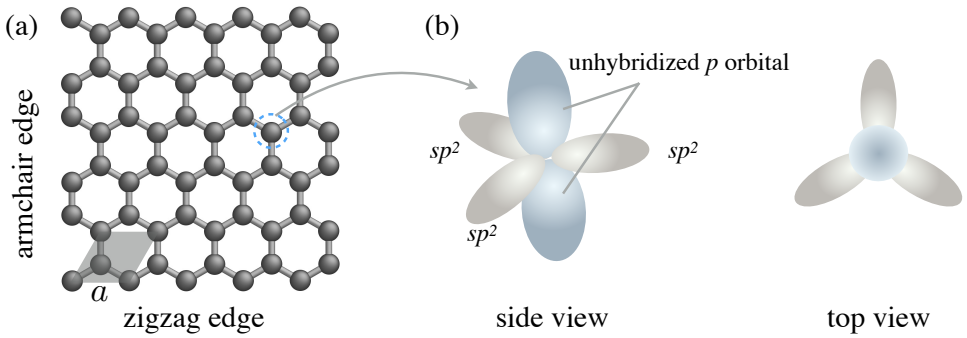


Figure 1.1: Crystal structure of graphene. (a) Top view of graphene. A unit cell (the shaded region) consists of two atoms, with a lattice parameter of $a = 2.46$ Å. (b) sp^2 hybridization in graphene.

as possible. As a result, it is positioned perpendicular to the plane where the sp^2 orbitals reside, with one lobe above the plane and the other below the plane. The sp^2-hybridized orbitals of different atoms come head to head with each other, thus forming strong covalent σ bonds; whereas the remaining unhybridized p orbitals make up the weak π bonds, and side-by-side π bonds of different atoms hybridize together to form the π bands. The π bands are half-filled that permit free-moving electrons, responsible for most of graphene's notable electronic properties [18].

Graphene, once presumed unstable and impossible to exist due to the formation of curved structures [19], was successfully extracted from bulk graphite by a mechanical exfoliation approach in air using adhesive tape by Geim and Novoselov in 2004[4] [15]. Ever since, other methods such as liquid-phase exfoliation [21], growth on SiC [22], growth on metals by precipitation [23], chemical vapor deposition [24] etc. have also been applied to successfully obtain graphene. Graphene has attracted significant interest due to its uncommon properties never seen in 3D bulk materials. It is a semiconductor with zero electronic band gap and has unusual 2D Dirac-like electronic excitation [25]. The Dirac electrons behave uncommonly in many ways, such as the confinement [26] and the integer quantum Hall effect [27]. Graphene has an unexpected high opacity, absorbing $\sim 2.6\%$ [28] of green light, and $\sim 2.3\%$ of red light [29]. As a result, we can see graphene with naked eyes, although it is only one-atom thick. This unusual opacity of graphene is

[4]Geim and Novoselov won the the Nobel Prize in Physics in 2010 "for groundbreaking experiments regarding the two-dimensional material graphene" [20].

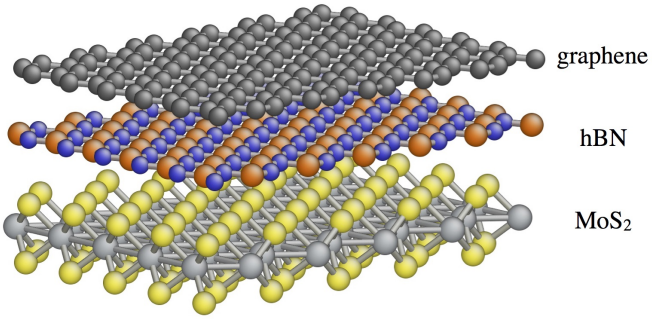


Figure 1.2: A two-dimensional heterostructures created by stacking graphene, hBN, and MoS_2 layer by layer.

defined solely by the *fine structure constant*, a fundamental physical constant independent of any material parameter [29, 30]. Graphene has superior thermal conductivity and high melting point. The thermal conductivity of suspended single layer graphene has been experimentally measured as 1500–2500 $\mathrm{W\,m^{-1}\,K^{-1}}$ [31–35]. The large spread can be attributed to measurement uncertainties as well as the variations in the quality of the graphene being measured. The melting point of graphene is around 4125 K [36]. As a comparison, copper has a thermal conductivity of $\sim 401\,\mathrm{W\,m^{-1}\,K^{-1}}$ [37], and the sun's surface have an effective temperature of 5777 K [36]. Mechanically, as the strongest material ever tested, graphene has a tensile strength of 130.5 GPa and a Young's modulus of 1 TPa [38]. Again, as a comparison, the A36 steel (a common structural steel) has a tensile strength of 400–550 MPa and a Young's modulus of 200 GPa [39].

Besides graphene, more than a dozen of different 2D crystals are found to be stable under ambient conditions. They can be broadly classified into three categories [40]: (i) **graphene family**, including graphene, hexagonal boron nitride (hBN), BCN, fluorographene, and graphene oxide; (ii) **transition metal dichalcogenide** of the type MX_2 with M a transitional metal atom (Mo, W, etc.) and X a chalcogen atom (S, Se, etc.); (iii) **2D oxides** such as micas and bismuth strontium calcium copper oxide (BSCCO). A high-throughput screening exploration of the Materials Project [41] together with density functional theory (DFT) calculations have predicted about 680 monolayers to be stable [42], although many remain to be synthesized and experimentally confirmed. Like graphene, these 2D materials (are expected to) also have unusual properties.

Completely new tunable materials can be created by stacking 2D materials layer by

layer to form 2D heterostructures (see figure 1.2 for an example). These materials have even more unusual and novel properties that their monolayer and 3D counterparts do not possess [40, 43]. For example, the electronic band gap of a graphene bilayer can be tuned by applying a variable external electric field, which allows great flexibility in the design and optimization of semiconductor devices such as p-n junctions and transistors [44]. A different manifestation of interesting behavior not found in the bulk is the superconductivity in intentionally misaligned (by a relative twist of $\sim 1.1^\circ$) graphene bilayers [45].

With these extraordinary properties, 2D materials and heterostructures are expected to be utilized in many applications such as semiconductors [46–48], light processing [49, 50], energy generation and storage [51–55], ultrasensitive sensors [56–59], medicine [60–64] and so forth. Although researches on fundamental properties of 2D materials and heterostructures and the attempt to apply them in a larger number of areas are intense, they are still in their infancy and are likely to remain one of the leading topics in condensed matter physics and materials science for many years [40].

1.3 Structure of thesis

The first part of this thesis will be concerned with the design and parameterization of both physics-based and machine learning interatomic potentials (IPs). Chapter 2 is mainly based on published work for physics-based IPs presented in [65–68]. However, this chapter will also provide an exposition of the symmetry requirements an IP (both physics-based and machine learning IPs) has to satisfy and the least-squares method to estimate the parameters in an IP. After introducing the general principles, two physics-based IPs are presented: a Stillinger–Weber (SW) potential for monolayer MoS_2 and a registry-dependent potential for the interlayer interactions in multilayer graphene. Chapter 3 is devoted to machine learning IPs. We first review the descriptors to represent atomic environments and some widely-used machine learning regression algorithms that have been applied to build IPs based on the descriptors. Then a neural network interatomic potential (NNIP) developed to model both the interlayer and intralayer interactions in multilayer graphene structures is introduced.

The second part (chapter 4) focuses on evaluating the quality of IPs. A review of

the approaches to quantify the uncertainty in IPs themselves and properties obtained using atomistic simulations with IPs is first provided. Then we discuss the use of Fisher information matrix (FIM) to measure the parametric sensitivity and the prediction uncertainty for the SW potential, as well as the application of dropout techniques to carry out uncertainty quantification and propagation (UQ+P) for an NNIP.

Finally, the KIM-based learning-integrated fitting framework (KLIFF) [69]—an open-source Python package for training both physics-based and machine learning IPs—is introduced in chapter 5. Its capabilities and implementation details are discussed briefly in this chapter, considering that the user manual [69] provides extensive information about this package. The thesis is concluded with a discussion of future work in chapter 6.

All the IPs developed in this thesis are implemented as KIM models archived in the open knowledgebase of interatomic models (OpenKIM) repository [70]. They can be used with major molecular simulation codes that conform to the knowledgebase of interatomic models (KIM) application programming interface (API), including LAMMPS [71], ASE [72], GULP [73], and DL_POLY [74] among others.

Chapter 2

Physics-based Potentials

The history of interatomic potentials (IPs) dates back to the 1920s, when Lennard–Jones (LJ) developed a pair potential for noble gases [75–77], $\phi(r) = 4\epsilon\left[(\sigma/r)^{12} - (\sigma/r)^{6}\right]$. The r^{-6} term in the LJ potential was proposed to model the van der Waals (vdW) forces based on theoretical derivations using two identical linear harmonic oscillators [78], whereas the r^{-12} term has no physical justification, chosen to model the Pauli repulsion at short ranges due to overlapping electron orbitals. Since then, a large number of IPs have been proposed, including

1. cluster potentials such as the three-body Stillinger–Weber (SW) potential [79] and four-body generalized pseudopotential theory [80];

2. pair functionals such as the embedded atom method (EAM) [81];

3. bond-order potentials such as the Tersoff [82], reactive empirical bond order (REBO) [83], long-range carbon bond order potential (LCBOP) [84], and reactivate force field (ReaxFF) [85] potentials.

Although these IPs become more and more complicated in terms of their mathematical form and the number of parameters, they can still be classified into the same category: physics-based IPs. A defining characteristic of such IPs is that their mathematical forms are devised to capture the underlying physics in the materials systems that they try to model. The physics-based IPs aim to subsume the complex chemistry of the Schrödinger equation into an effective form; therefore, they generally guarantee

better transferability (i.e. the ability to make predictions outside the properties an IP is trained to reproduce) compared with machine learning IPs that will be discussed in chapter 3.

In this chapter, we first discuss some of the requirements that IPs (both physics-based and machine learning) must satisfy and how to train IPs in section 2.1, and then present two physics-based IPs for two-dimensional (2D) materials and heterostructures: a SW potential for monolayer MoS_2 in section 2.2 and a registry-dependent potential for the interlayer interactions in graphene in section 2.3.

2.1 Background

A parametric interatomic potential (IP) takes the form

$$\mathcal{V} = \mathcal{V}(\boldsymbol{r}_1, \ldots, \boldsymbol{r}_{N_a}, Z_1, \ldots, Z_{N_\mathrm{a}}; \boldsymbol{\theta}), \tag{2.1}$$

where $\boldsymbol{r}_1, \ldots, \boldsymbol{r}_{N_\mathrm{a}}$ and $Z_1, \ldots, Z_{N_\mathrm{a}}$ are the coordinates and atomic species of a system of N_a atoms, respectively, and $\boldsymbol{\theta}$ denotes a set of fitting parameters. The IP energy defined above mush satisfy certain invariant requirements based on the nature of the laws of physics:

1. The *principle of material frame-indifference* states that the energy $\mathcal{V}$ of a system (and all quantities derived from it) must be invariant with respect to rigid-body translation and rotation [17]. Mathematically, we require

$$\mathcal{V}(\boldsymbol{Q}\boldsymbol{r}_1 + \boldsymbol{c}, \ldots, \boldsymbol{Q}\boldsymbol{r}_{N_a} + \boldsymbol{c}, Z_1, \ldots, Z_{N_\mathrm{a}}; \boldsymbol{\theta}) = \mathcal{V}(\boldsymbol{r}_1, \ldots, \boldsymbol{r}_{N_a}, Z_1, \ldots, Z_{N_\mathrm{a}}; \boldsymbol{\theta}), \tag{2.2}$$

   for all rotations $\boldsymbol{Q} \in SO(3)$ (proper orthogonal tensors) and vectors $\boldsymbol{c} \in \mathbb{R}^3$.

2. An IP should be invariant with respect to inversion, i.e.

$$\mathcal{V}(\boldsymbol{r}_1, \ldots, \boldsymbol{r}_{N_\mathrm{a}}, Z_1, \ldots, Z_{N_\mathrm{a}}; \boldsymbol{\theta}) = \mathcal{V}(-\boldsymbol{r}_1, \ldots, -\boldsymbol{r}_{N_\mathrm{a}}, Z_1, \ldots, Z_{N_\mathrm{a}}; \boldsymbol{\theta}). \tag{2.3}$$

   The above two invariance requirements imply that $\mathcal{V}$ can only be a function of the

distances between atoms [17], i.e.

$$\mathcal{V} = \mathcal{V}(r_{12}, r_{13}, \ldots, r_{1,N_\mathrm{a}}, r_{23}, \ldots, r_{N_\mathrm{a}-1,N_\mathrm{a}}, Z_1, \ldots, Z_{N_\mathrm{a}}; \boldsymbol{\theta}), \tag{2.4}$$

where r_{ij} is the Cartesian distance between atoms i and j.

3. An IP should be invariant with respect to permutation of the coordinates of atoms with the same species, i.e.

$$\mathcal{V}(\boldsymbol{r}_1, \ldots, \boldsymbol{r}_{N_\mathrm{a}}, Z_1, \ldots, Z_{N_\mathrm{a}}; \boldsymbol{\theta}) = \mathcal{V}(\pi[\boldsymbol{r}_1, \ldots, \boldsymbol{r}_{N_\mathrm{a}}], Z_1, \ldots, Z_{N_\mathrm{a}}; \boldsymbol{\theta}), \tag{2.5}$$

where $\pi[\cdot]$ denotes an arbitrary permutation honoring the atomic species requirement.

For notational simplicity, we assume that the atomic species information is implicitly carried by the coordinates, and thus we exclude Z from the mathematical form to write equation (2.1) as

$$\mathcal{V} = \mathcal{V}(\boldsymbol{r}_1, \ldots, \boldsymbol{r}_{N_\mathrm{a}}; \boldsymbol{\theta}). \tag{2.6}$$

2.1.1 Mathematical form of potentials

The total potential energy of a system of N_a atoms can be decomposed to the contributions of individual atoms, taking the form

$$\mathcal{V} = \sum_{i=1}^{N_\mathrm{a}} E_i, \tag{2.7}$$

where E_i is the energy of atom i. While some IPs may be constructed from different concepts, nevertheless they can be expressed in this form by rewriting its mathematical form. For example, the total energy of the Lennard–Jones (LJ) potential [75–77] is constructed as the sum of the energy of individual bonds,

$$\mathcal{V} = \frac{1}{2} \sum_{i=1}^{N_\mathrm{a}} \sum_{\substack{j=1 \\ j \neq i}}^{N_\mathrm{a}} \phi(r_{ij}), \tag{2.8}$$

where the $\frac{1}{2}$ is used to avoid double counting. We can simply rearrange it to get

$$\mathcal{V} = \sum_{i=1}^{N_\mathrm{a}} \left[\frac{1}{2} \sum_{\substack{j=1 \\ j \neq i}}^{N_\mathrm{a}} \phi(r_{ij}) \right] = \sum_{i=1}^{N_\mathrm{a}} E_i, \tag{2.9}$$

where $E_i \coloneqq \frac{1}{2} \sum_{j=1, j\neq i}^{N_\mathrm{a}} \phi(r_{ij})$ can be regarded as the energy of atom i. However, the decomposition of the total energy into contributions of individual atoms is not unique, which is extremely true for many-body IPs. The decomposition could have an effect on some properties obtained from atomic simulations such as the thermal conductivity calculated using certain formulation of the heat current [86].

The art of designing a physics-based IP lies in finding the "correct" functions to express the potential energy of individual atoms or atom groups based on the developer's physical intuition and sometimes a bit of luck. The mathematical forms of most IPs are constructed by composing elementary functions such as polynomials, exponentials, and trigonometric functions among others.

In practice, the fitted mathematical forms of some IPs are often stored in a discretized tabulated format as a list of data points, and intermediate values are obtained by interpolation. For example, a pair potential function $\phi(r)$ would be stored as a set of values: $(r_1, \phi_1), (r_2, \phi_2), \ldots, (r_n, \phi_n)$, where $\phi_i = \phi(r_i)$ and n is the number of data points. In some cases, an analytic form for the function does not exist and the fitting procedure directly generates the IP in its discretized form with a small number (typically 10–30) of ϕ_i values used as the fitting parameters. Interpolation of tabulated data using polynomials or splines is computationally more efficient than calculating the total energy and forces directly from the analytic mathematical forms, especially when the analytic mathematical forms are complicated. Many popular IPs are routinely tabulated, including the embedded atom method (EAM) [81,87,88], Finnis-Sinclair (FS) model [89], effective medium theory (EMT) [90], modified embedded atom method (MEAM) [91], and angular-dependent potential (ADP) [92,93] among others.

Users of tabulated IPs typically consider the data file containing the discretized mathematical forms as *the* IP without considering the nature of the interpolation. The reasoning behind this is that if enough data points are used, the type of interpolation

will not affect the results. This argument is largely correct for Monte Carlo (MC) and molecular dynamics (MD) simulations, which, for finite temperatures, sample many of the interpolated data points and obtain a "smeared" result that is effectively independent of the interpolation type. However, the nature of the interpolation can strongly affect methodologies that use higher-order derivatives of the IP without sampling, such as lattice dynamics calculations. An example where a calculation can "go wrong" due to interpolation effects is the application of the vibrational self-consistent field (VSCF) method [94] to the second-generation reactive empirical bond order (REBO) potential [83]. VSCF requires derivatives of an IP up to fourth order, but REBO incorporates a cubic spline with knots (data points) at graphene and diamond geometries. This leads to discontinuities in the (numerically-computed) third- and fourth-order derivatives at the spline knots and, hence, to a breakdown of the VSCF approach [65].

We have systematically studied the effect of tabulation and interpolation on the predictions of IPs [65]. The simplest possible case of a one-dimensional (1D) chain of copper atoms interacting via a nearest-neighbor modified Morse potential [95] has been employed to compute quasi-harmonic predictions for the thermal expansion and finite-temperature elastic constants as well as MD predictions for these properties. Although simple, this example includes all of the features that are expected to play out in three-dimensional (3D) lattice dynamics calculations. We study five types of splines: natural cubic, cubic Hermite, clamped quartic, clamped quintic, and quintic Hermite. As a cautionary tale, we also include a sixth, "naïve quartic" spline generated using an algorithm that appears reasonable but leads to significant errors. The predictions obtained from the spline computations are compared to the same computations performed with the analytic IP. The results show a strong effect of the interpolation on the computed properties. Strictly speaking, only the clamped quintic spline is able to reproduce all results obtained from the analytic IP. If the number of data points is on the order of 500, the quintic Hermite spline may also do a good job of reproducing the results.

The observed strong interpolation effects suggest that for tabulated IPs, the interpolation method should be definitely considered as part of the definition; otherwise, large errors may occur in atomistic simulations due to the use of such IPs.

2.1.2 Parameterization

Once we manage to devise an appropriate mathematical form of an IP, there still remain the parameters in the mathematical form to be determined. The parameterization process is typically formulated as a least-squares minimization problem, where we adjust the IP parameters so as to reproduce a training set of reference data obtained from experiments and/or first-principles calculations.

Least squares and maximum likelihood

Before discussing the parameterization of IPs, let's first briefly review the *least squares* method and the *maximum likelihood* method for parameter estimation and the relationship between them.

For a parametric model $\boldsymbol{h}(\boldsymbol{x};\boldsymbol{\theta})$, where $\boldsymbol{\theta}$ denotes the parameters in the model, given a set of observations of the input $\boldsymbol{x}_1,\ldots,\boldsymbol{x}_N$ and the corresponding output $\boldsymbol{y}_1,\ldots,\boldsymbol{y}_N$, we hope to find out what parameters $\boldsymbol{\theta}$ that are most likely to generate the outputs from the inputs. The least squares method aims to solve this problem by minimizing a quadratic loss function,

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\sum_{i=1}^{N}(\boldsymbol{y}_i - \boldsymbol{h}_i)^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\boldsymbol{y}_i - \boldsymbol{h}_i), \tag{2.10}$$

weighted by the inverse of an positive definite matrix $\boldsymbol{\Sigma}$, with respect to the parameters $\boldsymbol{\theta}$. The minimization problem can be solved using optimization algorithms such as the first-order gradient descent method or the second-order Newton's method.

From a probabilistic perspective, we can assume an observed output is given by the model prediction $\boldsymbol{h}(\boldsymbol{x};\boldsymbol{\theta})$ with an additive noise $\boldsymbol{\epsilon}$, i.e.

$$\boldsymbol{y} = \boldsymbol{h}(\boldsymbol{x};\boldsymbol{\theta}) + \boldsymbol{\epsilon}. \tag{2.11}$$

If the noise $\boldsymbol{\epsilon}$ follows a Gaussian distribution[1] with zero mean and a covariance matrix $\boldsymbol{\Sigma}$, i.e. $p(\boldsymbol{\epsilon}) = \mathcal{N}(\boldsymbol{\epsilon}|\boldsymbol{0},\boldsymbol{\Sigma})$, equation (2.11) can be written as a probabilistic model by

[1] Gaussian distribution is typically the choice if we have no extra information of the noise.

which an input $\boldsymbol{x}$ generates an output $\boldsymbol{y}$ given the parameters $\boldsymbol{\theta}$,

$$\begin{aligned} p(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{\theta}) &= \mathcal{N}(\boldsymbol{y}\,|\,\boldsymbol{h}(\boldsymbol{x};\boldsymbol{\theta}),\boldsymbol{\Sigma}) \\ &= \frac{1}{\sqrt{(2\pi)^M|\boldsymbol{\Sigma}|}}\exp\left[-\frac{1}{2}(\boldsymbol{y}-\boldsymbol{h})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\boldsymbol{y}-\boldsymbol{h})\right], \end{aligned} \tag{2.12}$$

where M is the dimension of $\boldsymbol{y}$ and $|\boldsymbol{\Sigma}|$ denotes the determinate of $\boldsymbol{\Sigma}$. This is the *likelihood* distribution for a single data point. For a data set of N observations, $\mathcal{D} = (\boldsymbol{X},\boldsymbol{Y})$ with inputs $\boldsymbol{X} = [\boldsymbol{x}_1^{\mathrm{T}};\ldots;\boldsymbol{x}_N^{\mathrm{T}}]$ and outputs $\boldsymbol{Y} = [\boldsymbol{y}_1^{\mathrm{T}};\ldots;\boldsymbol{y}_N^{\mathrm{T}}]$, the *likelihood* is

$$\begin{aligned} p(\boldsymbol{Y}\,|\,\boldsymbol{X},\boldsymbol{\theta}) &= \prod_{i=1}^{N} p(\boldsymbol{y}_i\,|\,\boldsymbol{x}_i,\boldsymbol{\theta}) \\ &= \left[(2\pi)^M|\boldsymbol{\Sigma}|\right]^{-\frac{N}{2}}\exp\left[-\frac{1}{2}\sum_{i=1}^{N}(\boldsymbol{y}_i-\boldsymbol{h}_i)^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\boldsymbol{y}_i-\boldsymbol{h}_i)\right], \end{aligned} \tag{2.13}$$

assuming the data points are independently and identically distributed (i.i.d.), i.e. the data points are drawn independently from the distribution in equation (2.12). An estimation of the parameters $\boldsymbol{\theta}$ can be then obtained by maximizing the likelihood distribution for the data set, and the parameters obtained in this manner are called the *maximum likelihood estimator*. The covariance $\boldsymbol{\Sigma}$ is not dependent on $\boldsymbol{\theta}$; therefore, maximizing equation (2.13) is equivalent to maximizing the argument of the exponential, which is further equivalent to minimizing equation (2.10).

We see that the parameters obtained from the least squares method and these from the maximum likelihood method shall be exactly the same. The least squares method implicitly assumes that the noise $\boldsymbol{\epsilon}$ is an independently and identically distributed (i.i.d.) variable with zero mean and covariance $\boldsymbol{\Sigma}$, leading to the minimization of the sum of the weighted squares. This is equivalent to maximizing the likelihood function for the data set under the Gaussian assumption, and therefore the least squares method is clearly a "disguised" maximum likelihood technique [96, 97].

The force-matching method

Traditionally, IPs are fitted to reproduce a set of material properties considered important for a given application such as the cohesive energy, equilibrium lattice constant,

and elastic constants of a given crystal phase. However, experience has shown that the *transferability* (i.e. the ability to accurately predict behaviors that they were not fitted to reproduce) of such IPs can be limited due to the small number of atomic configurations sampled in the training set (although recent work [98] has shown that this approach can be effective in some cases). Further, as the complexity of IPs increases (both in terms of the mathematical forms and the number of parameters), it can be difficult to obtain a sufficient number of material properties for the training set. This is particularly true for multispecies systems like intermetallic alloys.

To address these difficulties, Ercolessi and Adams [99] proposed the *force-matching* method, in which a training set containing material properties is augmented with the forces on atoms obtained by first-principles calculations for a set of atomic configurations. These can be configurations associated with important structures or simply random snapshots of the crystal as the atoms oscillate at finite temperature. By fitting to this information, the transferability of the potential is likely enhanced since it is exposed to a larger cross-section of configuration space. The issue of insufficient training data is also resolved since as many configurations as needed can be easily generated. This makes it possible to increase the number of parameters, and in fact in the original Ercolessi–Adams potential for aluminum [99] the mathematical forms were taken to be cubic splines with the spline knots serving as parameters. This gives maximum freedom to the fitting process.

The IPs presented in this thesis are all parameterized using the force-matching approach against a training set of first-principles total-energy calculations based on density functional theory (DFT). Besides forces, we usually include potential energy in the training set. For a training set of N configurations, equation (2.10) can be explicitly written as

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\sum_{i=1}^{N} w_i^{\mathrm{e}} \left[E(\boldsymbol{r}_i;\boldsymbol{\theta}) - \hat{E}_i\right]^2 + \frac{1}{2}\sum_{i=1}^{N} w_i^{\mathrm{f}} \|\boldsymbol{f}(\boldsymbol{r}_i;\boldsymbol{\theta}) - \hat{\boldsymbol{f}}_i\|^2, \tag{2.14}$$

where $E(\boldsymbol{r}_i;\boldsymbol{\theta})$ and $\boldsymbol{f}(\boldsymbol{r}_i;\boldsymbol{\theta}) = -\left(\partial\mathcal{V}/\partial\boldsymbol{r}\right)\big|_{\boldsymbol{r}_i}$ are the energy and forces in configuration i computed from an IP,[2] $\hat{E}_i$ and $\hat{\boldsymbol{f}}_i$ are the corresponding reference energy and forces

[2]Here, we abuse the notation for simplicity. Instead of denoting the coordinates of an atom, $\boldsymbol{r}_i$ indicates the concatenated coordinates of all the atoms in configuration i such that $\boldsymbol{r}_i \in \mathbb{R}^{3N_{\mathrm{a},i}}$, where

for configuration i in the training set, and w_i^{e} and w_i^{f} are the weights associated with the energy and forces of configuration i. The weights w_i^{e} and w_i^{f} are typically inversely proportional to $(N_{\mathrm{a},i})^2$ in which $N_{\mathrm{a},i}$ is the number of atoms in configuration i, so as to make each configuration contributes more or less equally to the loss function (i.e. avoid configurations with more atoms dominating the loss function). For energy in units of eV and forces in units of eV/Å, these weights have units of eV^{-2} and $(\mathrm{eV/Å})^{-2}$, respectively.

To obtain the optimal parameter set $\boldsymbol{\theta}$ of an IP that reproduces a training set as much as possible, global and/or local minimization algorithms are used to reduce the loss function in equation (2.14). For example, the *potfit* program uses simulated annealing for global minimization followed by a local polish using a conjugate gradient method [100, 101]. A difficulty associated with this procedure is that IPs are nonlinear functions that are often "sloppy" in the sense that their predictions are insensitive to certain combinations of the parameters [102]. These soft modes in parameter space can cause the minimization algorithms to fail to converge. Recently, an understanding of sloppy models based on ideas from differential geometry has led to efficient methods for fitting such models [103]. The basic idea is that the parameters of a model (like an IP) define a manifold in the space of data predictions. Fitting the model then corresponds to finding the point on the manifold closest to the training set. Using these ideas, Transtrum *et al.* [103, 104] augmented the Levenberg–Marquardt (LM) algorithm [105, 106] with a geodesic acceleration adjustment to improve convergence. The new geodesic LM algorithm [103, 104] is likely to be more efficient and more likely to converge for sloppy model systems than conventional approaches.

Geodesic LM algorithm has been used for a variety of applications in physics and biology, not to the IP fitting problem until we applied it to fit the environment-dependent interatomic potential (EDIP) for silicon [107]. We find that it is on average twice as likely to converge to the correct solution from different initial guesses than standard LM [105, 106] or Powell's [108] algorithms that tended to get trapped along sloppy directions [66]. However, the improved performance comes at the cost of about an order of magnitude increase in computational expense for geodesic LM. For the fitting of physics-based IPs, the computations are not prohibitive and geodesic LM should be the

$N_{\mathrm{a},i}$ is the number of atoms in configuration i. The same notation applies to the force $\boldsymbol{f}_i$.

preferred method. So we choose the geodesic LM method to carry out the minimization for the two IPs for two-dimensional (2D) materials that are discussed in sections 2.2 and 2.3.

2.2 A Stillinger–Weber potential for MoS_2

To date, several interatomic potentials (IPs) for MoS_2 have been proposed. The earliest published in 1975 is a valence force field (VFF) model by Wakabayashi *et al.* [109], in which the potential energy was decomposed into harmonic components. The interlayer interaction was assumed to be due to an axially symmetric force between sulfur atoms of neighboring layers, and the intralayer interaction was assumed to be associated with the stretching and bending of Mo–S bonds. The IP parameters were optimized to reproduce the phonon spectrum obtained from inelastic neutron scattering. Liang *et al.* [110] developed a reactive empirical bond order (REBO)-type IP for the Mo–S system using the master formula underlying the Abell [111] Tersoff [82,112,113] and REBO [114] potentials with an additional Lennard–Jones (LJ) [75,76] potential to describe the weak interlayer van der Waals (vdW) interactions. This IP was fit to a training set of the energy, bond length, and bond stiffness of Mo–Mo, Mo–S, S–S systems with the main objective to reproduce structural and elastic properties of MoS_2. Jiang *et al.* [115,116] developed two Stillinger–Weber (SW) [79] potentials for monolayer MoS_2. The first [115] considered all available two-body and three-body interactions in monolayer MoS_2 and was fit to the same phonon spectrum used in the VFF model [109]. In the second parameterization [116], Mo–Mo and S–S two-body interactions were neglected, and the potential was fit to bond lengths and bond angles from experiments and first-principles calculations, and to energies predicted by the VFF model [109]. A reactivate force field (ReaxFF) potential was developed by Ostadhossein *et al.* [117] to study energetics and reaction mechanisms in single- and multi-layer MoS_2. It was fit to a training set of energies, geometries, and charges derived from first-principles density functional theory (DFT) calculations for both clusters and periodic systems.

Recent developments in sensitivity analysis of stochastic systems based on relative entropy measures and Fisher information matrix (FIM) [118–120] have led to a deeper understanding of the force-matching methodology discussed in section 2.1.2. It is now

recognized that force matching is equivalent to relative entropy minimization provided that the training set of forces is obtained from a trajectory that samples the appropriate distribution function [121]. This improves the transferability of the potential since it can be shown that minimizing relative entropy also bounds the uncertainty in predictions of other observables [122]. The statistical mechanics approach to force matching was originally studied for equilibrium conditions [123, 124] and later extended to nonequilibrium steady states [122]. This generalization allows for treatment of driven systems subject to external conditions, such as thermal gradients and deformations.

In this section, we apply an information-theoretic based force-matching approach to retrain the SW potential of Jiang *et al.* [115, 116]. We find that this significantly improves the accuracy of the potential for a variety of properties. In addition, the information theory analysis yields (1) the uncertainty in the fitting parameters (i.e. the confidence with which the parameters are determined from a given training set); and (2) the sensitivity of the potential's predictions on its parameters (i.e. how variations in the parameters affect the results) [125, 126]. Here we only discuss the parameterization and testing of the SW potential for MoS_2, and a detailed sensitivity and uncertainty analysis of this model is discussed in section 4.2.

2.2.1 Definition of Stillinger–Weber model

The SW potential was originally introduced to model bulk silicon [79]. The innovation in this model was the inclusion of a three-body term to penalize configurations away from the tetrahedral ground state structure of Si. The model was later extended to other tetrahedral material systems including Ge [127], III–V compound semiconductors [128] and compounds of the major II–VI elements Zn, Cd, Hg, S, Se, and Te [129]. It has also been adapted for monolayer MoS_2 and monolayer black phosphorus that do not have a tetrahedral structure [115, 116].

The total SW potential energy $\mathcal{V}$ of a system consisting of N_a atoms is

$$\mathcal{V} = \sum_{i=1}^{N_a} \sum_{j>i}^{N_a} \phi_2(r_{ij}) + \sum_{i=1}^{N_a} \sum_{j\neq i}^{N_a} \sum_{\substack{k>j \\ k\neq i}}^{N_a} \phi_3(r_{ij}, r_{ik}, \beta_{jik}), \tag{2.15}$$

where the two-body interaction takes the form

$$\phi_2(r_{ij}) = \epsilon_{IJ}\hat{A}_{IJ}\left[B_{IJ}\left(\frac{r_{ij}}{\sigma_{IJ}}\right)^{-p_{IJ}} - \left(\frac{r_{ij}}{\sigma_{IJ}}\right)^{-q_{IJ}}\right] \times \exp\left(\frac{1}{r_{ij}/\sigma_{IJ} - a_{IJ}}\right), \quad (2.16)$$

and the three-body term is

$$\phi_3(r_{ij}, r_{ik}, \beta_{jik}) = \epsilon_{JIK}\hat{\lambda}_{JIK}\left[\cos\beta_{jik} - \cos\beta^0_{jik}\right]^2 \times \exp\left(\frac{\hat{\gamma}_{IJ}}{r_{ij}/\sigma_{IJ} - a_{IJ}} + \frac{\hat{\gamma}_{IK}}{r_{ik}/\sigma_{IK} - a_{IK}}\right), \quad (2.17)$$

in which $r_{ij} = \|\boldsymbol{r}_i - \boldsymbol{r}_j\|$ is the bond length between atoms i and j with $\boldsymbol{r}_i$ the coordinates of atom i, β_{jik} is the bond angle formed by bonds i–j and i–k with the vertex at atom i and β^0_{jik} is the corresponding predetermined reference angle. The IP parameters are $\epsilon, \hat{A}, B, p, q, \sigma, a, \hat{\lambda}, \hat{\gamma}$. Both the two-body and three-body terms are designed to be identically zero at the cutoff radius $r^{\text{cut}} = a\sigma$. The parameters depend on the species of the interacting atoms, which are indicated by uppercase subscripts. For example, ϵ_{IJ} is the parameter ϵ for the pairwise interaction between atom i of species I and atom j of species J.

Equations (2.16) and (2.17) can be recast in a form in which all parameters are independent and the dependence on the cutoff radius is made explicit. We define $A_{IJ} = \epsilon_{IJ}\hat{A}_{IJ}$, $\lambda_{JIK} = \epsilon_{JIK}\hat{\lambda}_{JIK}$, $\gamma_{IJ} = \sigma_{IJ}\hat{\gamma}_{IJ}$, and $r^{\text{cut}}_{IJ} = a_{IJ}\sigma_{IJ}$, then

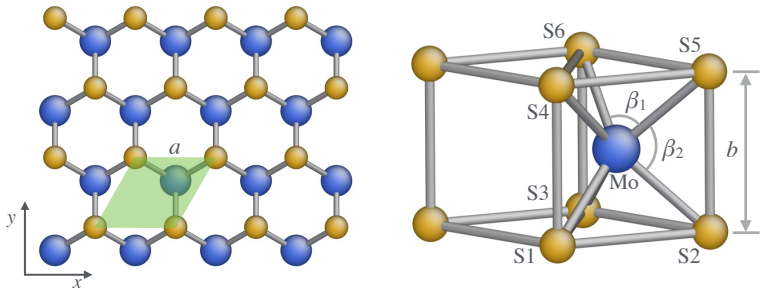
$$\phi_2(r_{ij}) = A_{IJ}\left[B_{IJ}\left(\frac{r_{ij}}{\sigma_{IJ}}\right)^{-p_{IJ}} - \left(\frac{r_{ij}}{\sigma_{IJ}}\right)^{-q_{IJ}}\right] \times \exp\left(\frac{\sigma_{IJ}}{r_{ij} - r^{\text{cut}}_{IJ}}\right), \quad (2.18)$$

and

$$\phi_3(r_{ij}, r_{ik}, \beta_{jik}) = \lambda_{JIK}\left[\cos\beta_{jik} - \cos\beta^0_{jik}\right]^2 \times \exp\left(\frac{\gamma_{IJ}}{r_{ij} - r^{\text{cut}}_{IJ}} + \frac{\gamma_{IK}}{r_{ik} - r^{\text{cut}}_{IK}}\right). \quad (2.19)$$

The new parameters are $A, B, p, q, \sigma, \lambda, \gamma$ along with the cutoff radii and equilibrium angles. Note that when $r > r^{\text{cut}}$, both ϕ_2 and ϕ_3 vanish. For MoS_2 we add an additional cutoff $r^{\text{cut}*}_{JK}$ for bond j–k in ϕ_3, i.e. ϕ_3 vanishes as well when $r_{jk} > r^{\text{cut}*}_{JK}$. This will be explained in section 2.2.1.

Based on the work of Jiang *et al.* [115, 116], two-body bond stretching (or compression) is considered for three types of interaction, i.e. $IJ \in \{$Mo–Mo, Mo–S, S–S$\}$ in



(a) (b)

Figure 2.1: Crystal structure of monolayer MoS_2. (a) Top view, where the green shaded region depicts a unit cell. (b) Side view of the shaded unit cell in (a). Each Mo atom is surrounded by six first-nearest-neighbor S atoms and each S atom is connected to three first-nearest-neighbor Mo atoms. Images rendered with AtomEye [130].

equation (2.18). For three-body bond bending, only interactions of type S-Mo-S (Mo is the species of the vertex atom) and Mo-S-Mo (S is the species of the vertex atom) are considered, i.e. in equation (2.19) $JIK \in \{\text{S–Mo–S, Mo–S–Mo}\}$. Consequently, there are only two λ parameters ($\lambda_{\text{S–Mo–S}}$ and $\lambda_{\text{Mo–S–Mo}}$) and a single γ parameter ($\gamma = \gamma_{\text{Mo–S}} = \gamma_{\text{S–Mo}}$). We will denote the set of all parameters as $\boldsymbol{\theta}$ in the following discussion.

Cutoffs and bond angles

The crystal structure of monolayer MoS_2 is shown in figure 2.1. It consists of a monatomic Mo plane sandwiched between two monatomic S planes. Mo and S atoms occupy alternating corners of a hexagon to form a honeycomb structure. A unit cell, the green shaded region in figure 2.1a, consists of one Mo atom and two S atoms. The in-plane zero-temperature equilibrium lattice constant of the relaxed structure obtained using the DFT code SIESTA [131] is $a_0 = 3.20$ Å, and the vertical separation between S layers is $b_0 = 3.19$ Å. Each Mo atom is surrounded by six first-nearest-neighbor S atoms and each S atom is connected to three first-nearest-neighbor Mo atoms.

The cutoff r_{IJ}^{cut} is set to the second-nearest neighbor distance of the corresponding IJ species. As an example, consider the calculation of $r_{\text{S–S}}^{\text{cut}}$. Referring to figure 2.1b, the nearest neighbors of atom S2 are S1, S3 and S5. In fact $d_{\text{S2–S1}} = d_{\text{S2–S3}} = a$ is slightly larger than $d_{\text{S2–S5}} = b$, however we ignore this small difference and treat all these atoms

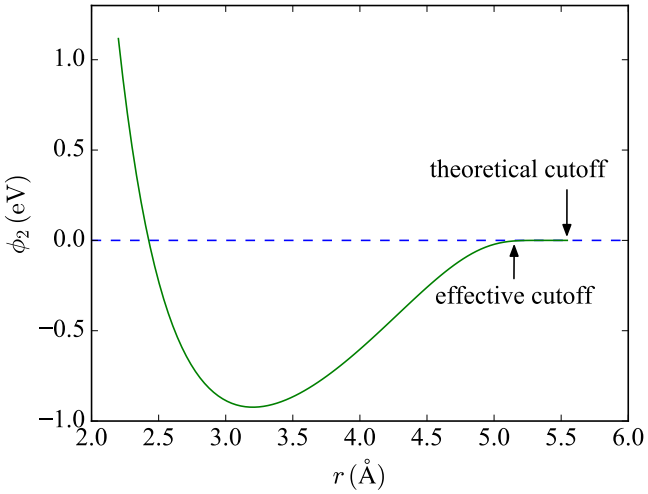


Figure 2.2: Pair function $\phi_2(r)$ of the SW potential, whose effective cutoff is much smaller than the theoretical one. The parameters used in the plot are $A = 4.15$ eV, $B = 0.44$, $p = 5$, $q = 0$, $\sigma = 2.85$ Å, and $r^{\text{cut}} = 5.55$ Å.

as the first neighbor ring for the purpose of determining the cutoff and the parameter $\sigma_{\text{S–S}}$ (as explained below). The second neighbors of S2 are atoms S4 and S6. Therefore $r^{\text{cut}}_{\text{S-S}} = d_{\text{S2–S4}} = \sqrt{a^2 + b^2} = 4.51956$ Å. The other two cutoffs are determined in a similar fashion: $r^{\text{cut}}_{\text{Mo–Mo}} = \sqrt{3}a = 5.54660$ Å and $r^{\text{cut}}_{\text{Mo–S}} = \sqrt{4a^2/3 + b^2/4} = 4.02692$ Å.

As pointed out by Zhou *et al.* [129], the SW two-body and three-body functions decay to close to zero at a distance smaller than the cutoff radius due to the presence of the exponential terms. This is demonstrated in figure 2.2 for the $\phi_2(r)$ function for the sample parameters listed in the caption. It is clear that while the theoretical cutoff is 5.55 Å, the potential energy becomes negligibly small beyond an *effective* cutoff of about $r = 5.10$ Å. If desired, this characteristic can be employed to expedite atomistic simulations by using the effective cutoff rather than the theoretical one to compute neighbor lists [129].

In the three-body term ϕ_3, an additional cutoff is employed to exclude certain interactions. Each Mo atom is surrounded by six first-nearest-neighbor S atoms, resulting in three different types of S–Mo–S angles after accounting for symmetry (β_1, β_2, and ∠S2–Mo–S4 in figure 2.1b). While β_1 and β_2 are almost of the same value, ∠S2–Mo–S4 is much larger. Because equation (2.19) only allows for one equilibrium angle, it is

desirable to exclude large-angle interactions of the third type so that the equilibrium structure of MoS_2 can be correctly described.[3] Following Refs. [115,116] and the GULP package [73,132], in addition to the two cutoffs r_{IJ}^{cut} and r_{IK}^{cut} included in equation (2.19), a new cutoff $r_{JK}^{\text{cut}^*}$ is applied to r_{jk} when J and K are S atoms. For $r_{jk} > r_{JK}^{\text{cut}^*}$, the three-body term involving atoms j, i and k is ignored. We take the additional cutoff to be $r_{\text{S-S}}^{\text{cut}^*} = 3.86095$ Å, corresponding to the average of the first- and second-nearest-neighbor distances of S–S bonds. This cutoff allows for bond angle interactions of types β_1 and β_2, but ∠S2–Mo–S4 type interactions will be excluded. We note that this introduces a discontinuity in the potential energy since the three-body term is abruptly removed at $r_{jk} = r_{JK}^{\text{cut}^*}$. The maximum discontinuity can be 2.67 eV, but this occurs when atom i is located in the middle between atoms j and k, which is far from the equilibrium structure. As long as the system is not subjected to extreme deformations far from the equilibrium ground state, the discontinuity will be mild if encountered and should not adversely affect molecular simulations.

Given $a_0 = 3.20$ Å and $b_0 = 3.19$ Å, it is straightforward to show that the angles in figure 2.1b are $\beta_1 = 81.92°$ and $\beta_2 = 81.61°$. Since the angles are quite close, we choose to use the same β^0 as the reference angle for both S–Mo–S and Mo–S–Mo three-body interactions. We set $\beta^0 = 81.79°$, which is the value of both β_1 and β_2 if a_0 and b_0 are equal.

Predetermined parameters and constraints

Aside from the cutoff radii specified in the previous section, the SW potential for MoS_2 has 18 parameters: three values each for A, B, p, q, σ, two for λ, and one for γ. It is non-trivial to fit so many parameters at once given that the IP is highly nonlinear. To facilitate the fitting process and make it more robust, some parameters are determined a priori and the values of some others are constrained.

In other parameterizations of the SW potential [79, 115, 116, 129], the exponents q and p were taken to be 0 and 4, respectively. Here we take $q = 0$, but allow p to be a fitting parameter that can only take on integer values.

In the original SW potential for Silicon [79], Stillinger and Weber determined σ by

[3] This is not a concern for Mo–S–Mo interactions, since there is only one type of Mo–S–Mo bond with a bond angle equal to β_1.

requiring $r_{\mathrm{m}} = 2^{1/6}\sigma$, where r_{m} is the distance at which $\phi_2(r)$ reaches its minimum.[4] In this work, σ is obtained in the same way. Given the lattice constants of the relaxed MoS_2 structure, $a_0 = 3.20$ Å and $b_0 = 3.19$ Å, the equilibrium bond lengths can be computed as $d_{\mathrm{Mo–Mo}} = 3.20$ Å, $d_{\mathrm{Mo–S}} = 2.44$ Å, and $d_{\mathrm{S–S}} = 3.19$ Å. Thus we have $\sigma_{\mathrm{Mo–Mo}} = 2.85295$ Å, $\sigma_{\mathrm{Mo–S}} = 2.17517$ Å and $\sigma_{\mathrm{S–S}} = 2.84133$ Å.

As in Ref. [116], we require that in the ground state structure all bonds are at their equilibrium lengths and all angles are at their equilibrium values, i.e. $(\partial\phi_2/\partial r)|_{r=d} = 0$ and $(\partial\phi_3/\partial\beta)|_{\beta=\beta^0} = 0$. The latter is satisfied automatically, and the former leads to a constraint relating B, p, q and σ,

$$\begin{aligned} B &= \frac{q(d/\sigma)^{-1-q}(d - r^{\mathrm{cut}})^2 + (d/\sigma)^{-q}\sigma^2}{p(d/\sigma)^{-1-p}(d - r^{\mathrm{cut}})^2 + (d/\sigma)^{-p}\sigma^2} \\ &= \frac{1}{pd^{-p-1}\sigma^{p-1}(d - r^{\mathrm{cut}})^2 + d^{-p}\sigma^p}, \end{aligned} \tag{2.20}$$

where d is the equilibrium bond length computed above and in the last equality $q = 0$ was used.

Accounting for the preset parameters and applying the constraint in equation (2.20), the parameters left to be determined are $\boldsymbol{\theta} = \{A, p, \lambda, \gamma\}$. This is a small subset of all the parameters, which greatly helps with the fitting process described next.

2.2.2 Parameterization

In the force-matching method [99], an IP is fit to first-principles forces for a training set of atomic configurations. If the configurations in the training set are obtained by sampling a thermodynamic ensemble (e.g. the canonical NVT ensemble or the isothermal-isobaric NPT ensemble), then the parameterization not only optimizes the forces but all observables that are defined as averages over the stationary distribution [122]. This significantly enhances the transferability of the IP. For the SW potential considered here, the training set is generated from a long thermostatted trajectory in the NPT ensemble from an *ab initio* molecular dynamics (AIMD) simulation.

The training set trajectory was obtained by AIMD using the DFT code SIESTA

[4] This equation comes from the LJ potential [75–77] $\phi(r) = 4\epsilon[(\sigma/r)^{12} - (\sigma/r)^6]$, whose minimum is at $r_{\mathrm{m}} = 2^{1/6}\sigma$. The two-body term $\phi_2(r)$ of the SW potential is based on the LJ potential with an additional exponential cutoff.

[131]. The interactions between ionic cores and valence electrons were modeled by a double zeta polarized basis set and norm-conserving pseudopotential [133] constructed within the Troullier-Martins formalism [134]. The exchange-correlation energy of the electrons is treated within the generalized gradient approximation (GGA) Perdew–Burke–Ernzerhof functional [135]. An energy cutoff of 90 Ry was used for the representation of charge density and potentials. Brillouin zone integration was carried out at a single k-point (Γ point).[5]

In the training set calculations, periodic boundary conditions were applied in all three directions with a vacuum of 30 Å perpendicular to the MoS_2 layer to minimize interactions between periodic images. The training set was constructed as follows. First, the equilibrium MoS_2 lattice structure was obtained by performing a full relaxation of a single unit cell allowing the cell to change its volume and shape until all stress components were less than 0.1 kBar and allowing the atoms to move until all forces were less than 0.03 eV/Å. The relaxed unit cell has an in-plane lattice constant of $a_0 = 3.20233$ Å and the separation between the two sulfur layers is $b_0 = 3.18928$ Å (see figure 2.1). Second, a rectangular block supercell was constructed with in-plane dimensions of 25.61 Å × 33.28 Å (corresponding to 8 × 12 relaxed unit cells) consisting of $N_\mathrm{a} = 288$ atoms: 96 Mo atoms and 196 S atoms. Third, AIMD simulations were performed using the supercell under NPT conditions with a pressure of $p = 0$ and temperature of $T = 750$ K.[6] The atoms were initially assigned random velocities drawn from the Maxwell-Boltzmann distribution with the temperature equal to twice the target temperature. The system was then integrated in time for 3000 steps with a time step of $\Delta t = 0.7$ fs. The first 1000 steps were discarded to allow the system to equilibrate. In the subsequent 2000 steps, the atom coordinates $\boldsymbol{r}$ and the forces on the atoms $\hat{\boldsymbol{f}}$ were recorded in the training set. Thus the training set is $\{(\boldsymbol{r}_i, \hat{\boldsymbol{f}}_m)\}_{i=1}^{N}$, where $N = 2000$, $\boldsymbol{r}_i \in \mathbb{R}^{3N_\mathrm{a}}$, and $\hat{\boldsymbol{f}}_i \in \mathbb{R}^{3N_\mathrm{a}}$.

The parameters $\boldsymbol{\theta} = \{A, p, \lambda, \gamma\}$ are optimized by minimizing the loss function

[5] Brillouin zone integration using a single k-point is generally inaccurate. However due to the relatively large supercell used here, the size of the reciprocal vectors are relatively small, and thus the sampling grid in reciprocal space is dense. The resulting accuracy is considered adequate, especially considering the high cost of the AIMD calculations that would greatly increase with a denser k-point grid.

[6] We also tried to fit the potential at other temperatures, but found only slight differences between the fitted potential parameters and the resulting predictions for material properties. We therefore only include the results for $T = 750$ K.

Table 2.1: Fitted SW–FM parameters in the two-body term ϕ_2.

	Interaction		
Parameter	Mo-Mo	Mo-S	S-S
A (eV)	3.9781804791	11.3797414404	1.1907355764
B	0.4446021306	0.5266688197	0.9015152673
p	5	5	5
q	0	0	0
σ (Å)	2.85295	2.17517	2.84133
r^{cut} (Å)	5.54660	4.02692	4.51956

Table 2.2: Fitted SW–FM parameters in the three-body term ϕ_3.

$\lambda_{\text{S-Mo-S}} = 7.4767529158$ eV	$\lambda_{\text{Mo-S-Mo}} = 8.1595181220$ eV
$\gamma = 1.3566322033$ Å	$\beta^0 = 81.7868°$
$r^{\text{cut}}_{\text{Mo-S}} = 4.02692$ Å	$r^{\text{cut}*}_{\text{S-S}} = 3.86095$ Å

defined in equation (2.14) using the geodesic Levenberg–Marquardt (LM) algorithm. There is no energy data in the training set, so the energy weight w_i^e in equation (2.14) is set to 0; the force weight w_i^f is set to 1[7]. The initial guesses of the parameters were taken from Ref. [115]. The fitted parameters are listed in tables 2.1 and 2.2. We denote the new SW potential as SW–FM (Stillinger–Weber Force Matching) for later use in comparison.

2.2.3 Results and predictions

To test the accuracy of the SW–FM potential, we computed the temperature dependence of the lattice constants and stiffness of MoS_2. These properties are important for the design of MoS_2 based electronic devices, since internal stress or strain due to thermal expansion can degrade performance or even cause damage [136]. The calculations were performed using the classical atomic simulation code large-scale atomic/molecular massively parallel simulator (LAMMPS) [71, 137]. For all simulations, periodic boundary conditions were applied in all directions, with a spacing of 40 Å in the direction perpendicular to the MoS_2 layer to isolate it from its periodic images. The simulation

[7]Since each configuration in the training set has the same number of atoms, there is no need to normalize the force weight by the number of atoms.

Table 2.3: Equilibrium lattice constants a_0 and b_0 (Å), cohesive energy E_c per unit cell (eV), and elastic constants C_{11} and C_{12} (N/m) for SW–FM, other IPs in the literature, and first-principles results.

Method	a_0	b_0	E_c	C_{11}	C_{12}
SW–FM	3.19702	3.19386	15.28	119.2	41.0
SW–Jiang 2013	3.09368	3.18216	12.76	140.8	52.7
SW–Jiang 2015	3.11072	3.12898	3.72	105.0	28.7
REBO	3.16752	3.24248	21.48	154.4	45.8
ReaxFF[a]	3.19	3.11	15.20	205.1	81.6
SIESTA (GGA: PBE)	3.20	3.19	15.90	-	-
VASP (GGA: PW91)[b]	3.20	3.13	15.55	-	-
VASP (LDA)[b]	3.11	3.11	19.05	-	-
VASP (GGA: PBE)[c]	3.19	3.13	15.21	-	-
VASP (LDA)[c]	3.13	3.12	18.75	-	-
VASP (LDA)[d]	-	-	-	140.0	40.0
VASP (GGA: PBE)[d]	-	-	-	130.0	40.0
VASP (GGA: PBE)[e]	-	-	-	132.7	33.0

[a]Ref. [117]
[b]Ref. [138]
[c]Ref. [139]
[d]Ref. [140]
[e]Ref. [141]

setup and results for the different properties are described below.

Lattice constants and cohesive energy

The zero-temperature equilibrium lattice constants and cohesive energy of MoS_2 were obtained by minimizing the energy of a single unit cell using conjugate gradients with energy and force tolerances of 10^{-10} eV and 10^{-10} eV/Å, respectively. The results for the SW–FM potential along with other IPs and DFT results are listed in table 2.3. As expected, both a_0 and b_0 agree with the SIESTA predictions since the σ parameters were preset to reproduce the equilibrium structure as explained in section 2.2.1. The cohesive energy per unit cell E_c predicted by SW–FM is in good agreement with SIESTA (and other DFT) results.

The cohesive energy versus lattice constant curves plotted in figure 2.3 for different IPs and SIESTA show the effect of stretching and compressing MoS_2, which can be

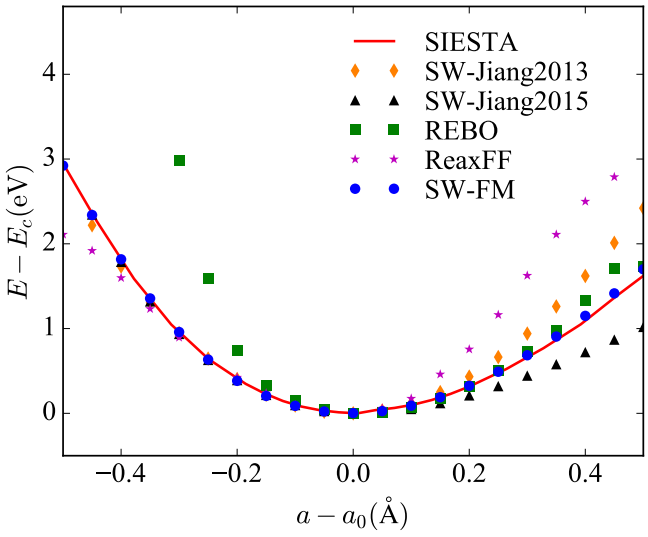


Figure 2.3: Energy of unit cell as a function of in-plane lattice constant a. The data points are shifted such that all minima coincide. E_c and a_0 for each potential are listed in table 2.3.

important for practical applications due to pre-straining or rippling. The points were computed by creating a unit cell with in-plane lattice constant a and relaxing the unit cell atoms in the out-of-plane direction. The results show that SW–FM agrees with SIESTA results across the entire range of stretching and compression (about ±15%), whereas other IPs agree either in tension or compression, but not both.

Elastic constant

The zero-temperature elastic constants were computed using LAMMPS by finite difference, $C_{11} = \Delta\sigma_1/\Delta\epsilon_1$ and $C_{12} = (\Delta\sigma_1/\Delta\epsilon_2 + \Delta\sigma_2/\Delta\epsilon_1)/2$ in Voigt notation, where $\Delta\sigma$ and $\Delta\epsilon$ are the stress and strain induced by infinitesimally displacing atoms from their equilibrium positions.

Due to symmetry C_{11} and C_{22} are the same, which means that orientation (at least the armchair and zigzag directions) does not affect the elastic behavior of MoS_2. The results for C_{11} and C_{12} are listed in table 2.3. The SW–FM predictions are in good overall agreement with DFT results, comparable to the other IPs (except for the ReaxFF potential, which appears to overestimate the elastic constants).

In-plane linear thermal expansion coefficient

The in-plane linear thermal expansion coefficient (LTEC) α_L of MoS_2 provides a measure of the temperature dependence of the lattice constants. There are two main methods for calculating LTEC from molecular dynamics (MD) simulations. First, in the direct method, the LTEC is computed from its definition by taking the first derivative of lattice constant with respect to temperature at constant pressure:

$$\alpha_L = \frac{1}{a} \left. \frac{\partial a}{\partial T} \right|_p . \tag{2.21}$$

Second, in the fluctuation method, the LTEC is computed as an ensemble average of the covariance of the Hamiltonian $\mathcal{H}$ and the volume V [142]:

$$\alpha_L = \frac{1}{2k_B T^2 \langle V \rangle} \left[\langle \mathcal{H} V \rangle - \langle \mathcal{H} \rangle \langle V \rangle \right], \tag{2.22}$$

where $\langle \cdot \rangle$ denotes a phase average, and k_B is the Boltzmann constant. A detailed derivation of equation (2.22) is given in Appendix A.

To generate the data for both methods, a series of isothermal-isobaric (NPT) MD simulations were performed with a configuration of 1200 atoms (400 Mo and 800 S) at different temperatures and zero pressure. The equations of motion were integrated using a velocity–Verlet algorithm with a time step of 1 fs. The system was initially maintained at constant temperature using a Langevin thermostat for 10^6 time steps. Then a Berendsen barostat was added and the system was evolved for another 10^6 time steps. This equilibration phase effectively dissipates lattice phonons generated by the initial conditions. Finally, the system was switched to an isothermal-isobaric (NPT) ensemble for 10^7 time steps using a Nose–Hoover thermostat and barostat to control the temperature and pressure with damping coefficients of 0.01 fs^{-1} and 0.001 fs^{-1}, respectively.

At a given temperature the equilibrium supercell size in the x direction, L_x, was computed by averaging the instant cell size values. The equilibrium lattice constant defined in equation (2.21) follows as $a = L_x/c$, where c is the number of unit cells along the x direction in the supercell. (For example, if we use the system depicted

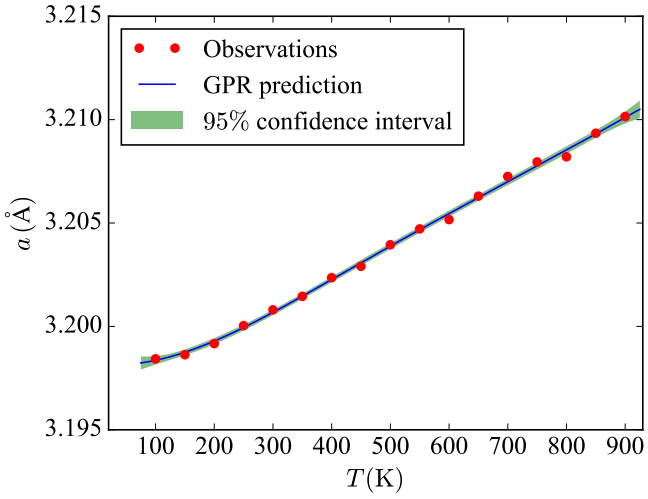
in figure 2.1a, c equals 4.)[8] The equilibrium lattice constant at different temperatures is plotted in figure 2.4a. To obtain the corresponding LTEC curve using the direct method in equation (2.21) it is necessary to obtain the slope of the lattice constant curve. To this end we use the Gaussian process regression (GPR) method implemented in scikit–learn [143, 144] to fit the lattice constant data (blue line in figure 2.4a).[9] The LTEC is then computed from equation (2.21) by using finite differences to compute the derivative of the GPR curve. To use GPR, it is necessary to provide lattice constant uncertainty. To compute this uncertainty at a given temperature, ten subsets, each with 5×10^5 simulation steps, were drawn randomly and independently from the simulation trajectory. The mean lattice constant was computed for each subset and from this set of values the standard deviation was obtained and used as the uncertainty. The result is plotted in figure 2.4b. The numerical values of the standard deviations are too small to be seen, so they are not depicted in the figures.

For the fluctuation method, α_L was computed using equation (2.22). The resulting LTEC α_L values are plotted in figure 2.5. Similar to the direct method, GPR was used to fit the data with uncertainties computed using the same procedure described above. In the figure, the uncertainties are shown as error bars, and the 95% confidence interval predicted by GPR is plotted as the shaded green region.

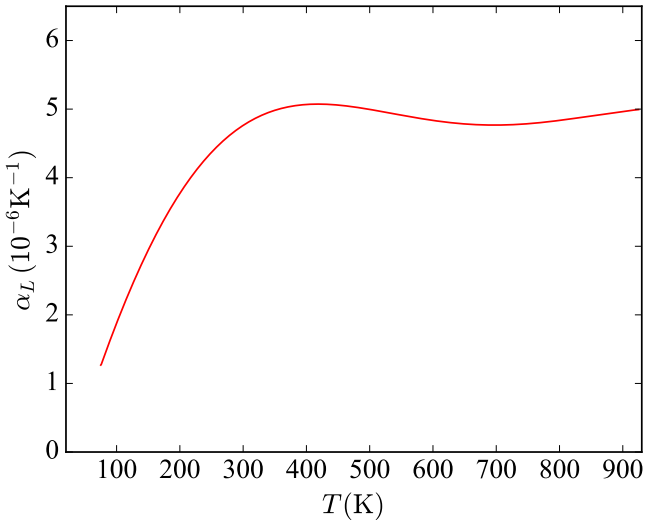
Both the direct and fluctuation methods using SW–FM show that the LTEC α_L increases quickly at low temperatures and saturates at about 400 K. These results are in agreement with quasiharmonic DFT predictions [136, 146], and classic MD prediction [145] using the REBO potential for MoS_2 [110]. We also computed the temperature dependence of the in-plane lattice constant using the SW potentials in Refs. [115, 116]. The former [115] predicts that the lattice constant decreases with increasing temperature, resulting in a negative LTEC α_L. The latter [116] predicts a positive increasing LTEC α_L in the temperature range 0 to 900 K, i.e. the LTEC α_L does not saturate at high temperature as observed by SW–FM and other sources as described above.

[8] We verified that upon heating the MoS_2 hexagonal lattice expands uniformly. Therefore it does not matter whether the lattice is oriented in the zigzag or armchair direction along the x-axis when computing the lattice constant a, since both will give the same result.

[9] Polynomials are often used to fit lattice constant data in order to compute the LTEC, for example in Ref. [145]. However, we found that in our case polynomials were far too sensitive. A small perturbation in one of the lattice constant data points can lead to a completely different LTEC curve.



(a)



(b)

Figure 2.4: (a) Equilibrium lattice constant a, and (b) the corresponding LTEC α_L computed using the direct method for the SW-FM potential.

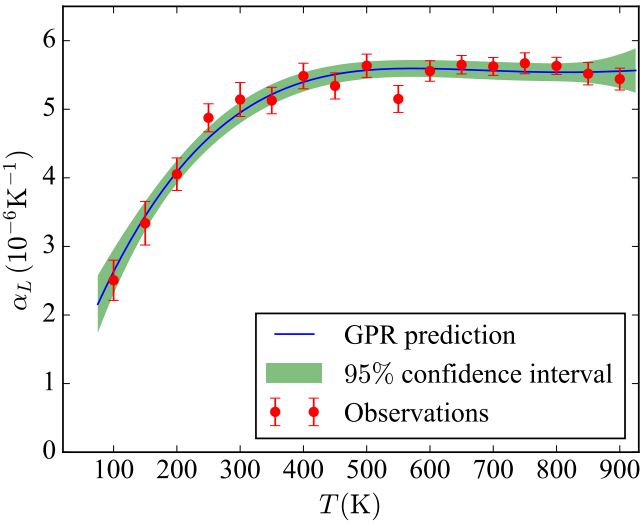


Figure 2.5: LTEC α_L computed using the fluctuation method for the SW-FM potential. The line is a GPR fit to the data.

2.2.4 Summary

We have parameterized a SW potential for MoS_2 using the force-matching method (SW–FM), where the potential parameters were optimized to match as closely as possible a training set of forces generated from a SIESTA AIMD trajectory at $T = 750$ K. The cutoffs and the reference bond angles were determined from the geometry of relaxed monolayer MoS_2 structure predicted by SIESTA. The equilibrium bond lengths and bond angles are prebuilt into the potential by applying appropriate constraints to the IP parameters. In this way, the relaxed structure of monolayer MoS_2 is guaranteed to have the correct geometry.

To test the accuracy of the fitted IP, it was used to compute the lattice constants, cohesive energy versus lattice constant curve, elastic constants, and in-plane linear thermal expansion coefficient. Our validation tests show that:

1. The SW–FM potential correctly predicts the equilibrium lattice constants, cohesive energy, and energy versus lattice constant curves.

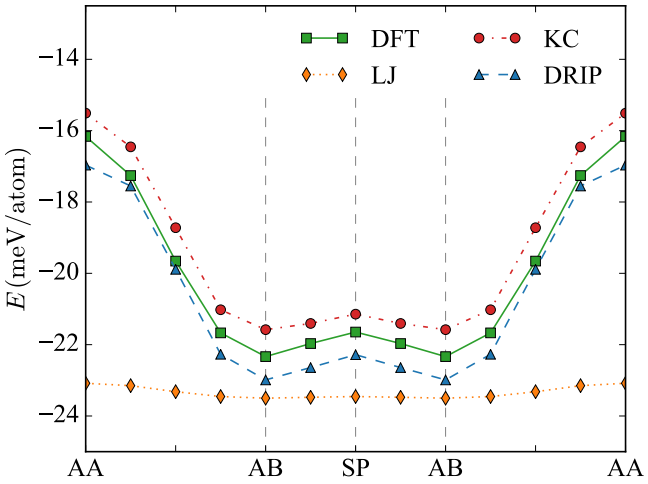
2. The elastic constant C_{11} is a bit underestimated compared with the first principles predictions, but the overall predictions for C_{11} and C_{12} are good.

3. The in-plane linear thermal expansion coefficient α_L, computed using both the direct method and the fluctuation method, increases rapidly at low temperature and saturates at high temperature in agreement with first-principles calculations and classical computations using the REBO potential for MoS_2.

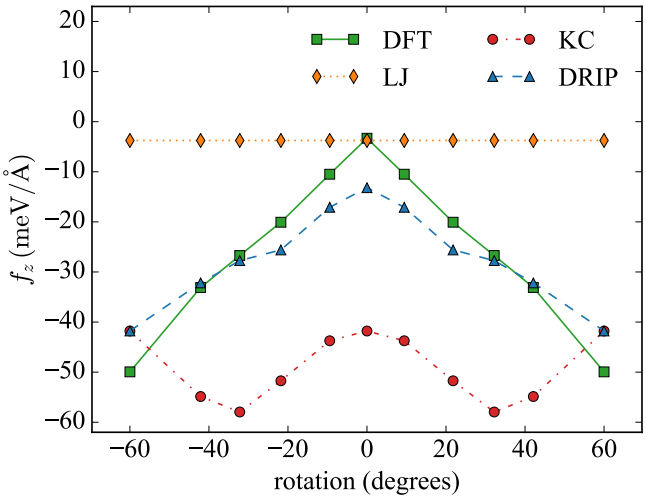
For the properties computed in this work, we find that the SW–FM potential outperforms the previous SW potentials for MoS_2 [115, 116] on which this work is based, and has comparable accuracy to the REBO [110] and ReaxFF [117] reactive potentials. Thanks to its simple mathematical form, MD simulations with SW–FM are significantly faster than with REBO or ReaxFF. We note that SW–FM is parameterized only for monolayer MoS_2 in the 2H phase, and thus should not be used for other phases of MoS_2 (e.g. bulk MoS_2 or the monolayer 1T phase).

2.3 An interlayer potential for graphene

A large number of interatomic potentials (IPs) have been developed to model the strong covalent bonds in carbon systems. Among these are bond-order potentials, such as the Tersoff [113, 147] and reactive empirical bond order (REBO) [83, 114] potentials, which allow for bond breaking and formation depending on the local atomic environments. Such models have been shown to be accurate for many problems and are widely used, but are not suitable for layered two-dimensional (2D) materials since they do not include long-range weak interactions. To address this, the adaptive intermolecular reactive empirical bond order (AIREBO) [148] potential (based on REBO) added a 6–12 form of the Lennard–Jones (LJ) potential [77] to model van der Waals (vdW) interactions. For graphitic structures, the LJ potential works well in describing the overall binding characteristics between graphene layers. For example, the LJ parameterization used in AIREBO predicts an equilibrium layer spacing of 3.357 Å and a c-axis elastic modulus of 37.78 GPa for graphite, in good agreement with first-principles and experimental results. The isotropic nature of LJ, that is, the fact that it depends only on distance between atoms and not orientation, makes it too smooth to distinguish energy variations for different relative alignments of layers [149]. Figure 2.6a shows the energy variation obtained by sliding one layer relative to the other along the armchair direction of a graphene bilayer (the stacking states are shown in figure 2.7). The energy remains



(a)



(b)

Figure 2.6: Energy and force variations when sliding and twisting a graphene bilayer. (a) Energy variation of sliding one layer relative to the other along the armchair direction. (b) Out-of-plane component of the force on the atom at the rotation center (blue circle labeled 1 in the bottom layer of the right-most plot in figure 2.7). Rotation by 0° corresponds to AB stacking, and rotation by ±60° corresponds to AA stacking. In both sliding and twisting, periodic boundary conditions are applied and the layer spacing is fixed at 3.4 Å. Details are provided in section 2.3.2.

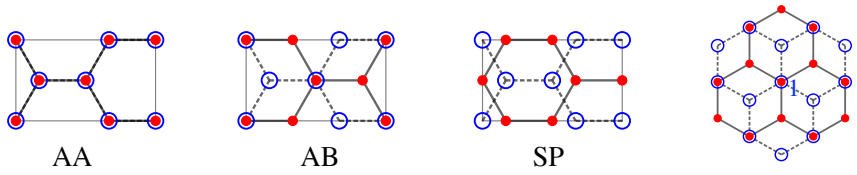


Figure 2.7: AA, AB, and SP stackings of bilayer graphene. Blue circles and red dots represent atoms in two separate layers.

nearly constant with a maximal difference of 0.41 meV/atom between the AA and AB stackings, a small fraction (6.6%) of the density functional theory (DFT) result (also shown in the figure).

The reason that the LJ potential fails to capture the energy variations due to interlayer sliding is that in addition to vdW, the interlayer interactions include short-range Pauli repulsion between overlapping π orbitals of adjacent layers. These repulsive interactions are not well described by a simple pair potential like LJ [150–152]. The Kolmogorov–Cresp (KC) interlayer potential was developed to account for this registry effect (relative alignment of layers) in graphitic structures [150]. In the KC potential, the dispersive (vdW) attraction between layers is described using the same theoretically-motivated r^{-6} term as in LJ, and π orbital overlap is modeled by a Morse [153] type exponential multiplied by a registry-dependent modifier that depends on the transverse distance between atom pairs. The KC potential has been modified and reparameterized to better fit the energy variations between different stacking states predicted by DFT-D (DFT with dispersion corrections) [154]. It has also been adapted for other 2D materials such as h-BN [151] and graphene/h-BN [152, 155] heterostructures.

The energy corrugation obtained by the KC potential is in agreement with DFT as shown in figure 2.6a. However, the forces obtained from the KC potential deviate significantly from the DFT results. This implies that equilibrium structures associated with energy minima will differ as well. To illustrate this point, consider a graphene bilayer where one layer is rigidly rotated relative to the other. Figure 2.6b shows the force in the z-direction (perpendicular to the layers) acting on the bottom atom on the rotation axis (atom 1 in the bottom layer of the right-most plot in figure 2.7) as a function of rotation angle. The force predicted by the KC potential decreases and then increases from AA ($\pm 60°$) to AB ($0°$), whereas DFT predicts a monotonic increase from

AA to AB. In particular, the KC potential yields the same z-force for the AA and AB stackings,[10] which indicates that the KC potential cannot distinguish the overlapping atoms at the rotation center in these states. This is intrinsic to the KC potential. The force on the central atom in the AA and AB states is identical, regardless of the choice of KC parameters. The modified KC potential [154] has the same problem. The LJ potential does even worse (figure 2.6b), predicting a constant force on the central atom that is independent of the rotation angle.

In this section, a new registry-dependent interlayer potential for graphitic structures is developed that addresses the limitations of the KC potential described above. A dihedral-angle-dependent term is introduced into the registry modifier of the repulsive part that makes it possible to distinguish forces in AA and AB states. We refer to this potential as the dihedral-angle-corrected registry-dependent interlayer potential (DRIP). DRIP is validated by showing that it correctly reproduces the DFT energy and forces for different sliding and rotated states as well as structural and elastic properties. It is then applied to study structural relaxation in twisted graphene bilayers and exfoliation of graphene from graphite; these representative example are large-scale applications that cannot be studied using DFT.

2.3.1 Definition of model

The DRIP mathematical form is

$$\mathcal{V} = \frac{1}{2}\sum_i \sum_{j\notin \text{layer}\, i} \phi_{ij}, \tag{2.23}$$

where $j \notin \text{layer}\, i$ means j runs over all atoms except for the ones that are in the same layer as atom i. The pairwise interaction is based on the KC form with dihedral modifications:

$$\phi_{ij} = f_c(x_r)\left[e^{-\lambda(r_{ij}-z_0)}\left[C + f(\rho_{ij}) + g(\rho_{ij},\{\alpha_{ij}^{(m)}\})\right] - A\left(\frac{z_0}{r_{ij}}\right)^6\right], \quad m = 1,2,3. \tag{2.24}$$

[10] Note that the x and y components of the force are zero at AA and AB due to symmetry.

The cutoff function $f_c(x)$ is same as that used in the reactivate force field (ReaxFF) potential [85] and the interlayer potential for h-BN [151, 152]:

$$f_c(x) = 20x^7 - 70x^6 + 84x^5 - 35x^4 + 1, \tag{2.25}$$

for $0 \leq x \leq 1$ and vanishes for $x > 1$, while it has zero first and second derivatives at $x = 1$; in the expressions where this function appears its argument is always non-negative. The variable x_r in equation (2.24) is the scaled pair distance $x_r = r_{ij}/r_{\text{cut}}$. The use of $f_c(x_r)$ ensures that DRIP is smooth at the cutoff r_{cut} (set to 12 Å), a feature that the KC model does not possess.

The term with r_{ij}^{-6} dependence in equation (2.24) models attractive vdW interactions (as in LJ), while the repulsive interactions due to orbital overlap are modeled by the exponential term multiplied by a registry-dependent modifier. The transverse distance function $f(\rho)$ has the same form as in KC:

$$f(\rho) = e^{-y^2}\left[C_0 + C_2 y^2 + C_4 y^4\right], \quad y = \frac{\rho}{\delta} \tag{2.26}$$

with its argument in equation (2.24) given by the expression

$$\rho_{ij}^2 = r_{ij}^2 - (\boldsymbol{n}_i \cdot \boldsymbol{r}_{ij})^2, \tag{2.27}$$

in which $\boldsymbol{r}_{ij}$ is the vector connecting atoms i and j, r_{ij} is the corresponding pair distance, and $\boldsymbol{n}_i$ is the layer normal at atom i. For example, as shown in figure 2.8, $\boldsymbol{n}_i$ can be defined as the normal to the plane determined by the three nearest-neighbors of atom i: k_1, k_2 and k_3:

$$\boldsymbol{n}_i = \frac{\boldsymbol{r}_{k_1k_2} \times \boldsymbol{r}_{k_1k_3}}{\|\boldsymbol{r}_{k_1k_2} \times \boldsymbol{r}_{k_1k_3}\|}. \tag{2.28}$$

Note that in general $\rho_{ij} \neq \rho_{ji}$ because the normals $\boldsymbol{n}_i$ and $\boldsymbol{n}_j$ depend on their local environments.

The dihedral angle function is given by

$$g(\rho, \{\alpha_{ij}^{(m)}\}) = B f_c(x_\rho) \sum_{m=1}^{3} e^{-\eta \alpha_{ij}^{(m)}}, \tag{2.29}$$

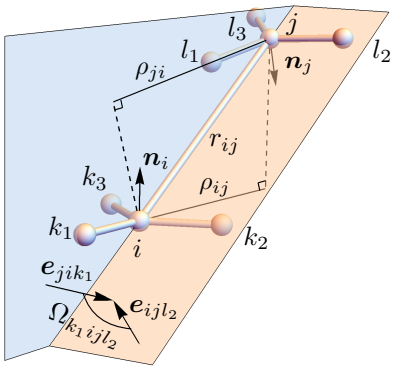


Figure 2.8: Schematic representation of an atomic geometry that defines the normal vectors $\boldsymbol{n}_i$ and $\boldsymbol{n}_j$ and the dihedral angle $\Omega_{k_1ijl_2}$.

where $\alpha_{ij}^{(m)}$ is the product of the three cosines of the dihedral angles formed by atom i (in layer 1), its mth nearest-neighbor k_m, atom j (in layer 2), and its three nearest-neighbors l_1, l_2 and l_3:

$$\alpha_{ij}^{(m)} = \cos\Omega_{k_mijl_1} \cos\Omega_{k_mijl_2} \cos\Omega_{k_mijl_3} \tag{2.30}$$

$$\cos\Omega_{kijl} = \boldsymbol{e}_{jik} \cdot \boldsymbol{e}_{ijl} \tag{2.31}$$

$$\boldsymbol{e}_{jik} = \frac{\boldsymbol{r}_{ik} \times \boldsymbol{r}_{ji}}{\|\boldsymbol{r}_{ik} \times \boldsymbol{r}_{ji}\|}, \qquad \boldsymbol{e}_{ijl} = \frac{\boldsymbol{r}_{jl} \times \boldsymbol{r}_{ij}}{\|\boldsymbol{r}_{jl} \times \boldsymbol{r}_{ij}\|}. \tag{2.32}$$

To understand the physical origin of the terms defined in equations (2.30) to (2.32), recall that a dihedral angle Ω is the angle between two planes defined by four points that intersect at a line defined by two of them as shown in figure 2.8. Here, the intersection line is defined by atoms i and j. The two planes are then defined by atoms (j, i, k_1) and (i, j, l_2). The normals to these planes are $\boldsymbol{e}_{jik_1}$ and $\boldsymbol{e}_{ijl_2}$, respectively, defined in equation (2.32), with the corresponding dihedral angle given by equation (2.31). The dihedral product $\alpha_{ij}^{(m)}$ monotonically decreases when twisting a graphene bilayer from AB to AA stacking, and consequently can be utilized to construct a potential function that distinguishes AB and AA stacking and the intermediate stacking states. The cutoff function $f_c(x_\rho)$ in equation (2.29) is the same as that in equation (2.25), and $x_\rho = \rho/\rho_{\text{cut}}$, where we set $\rho_{\text{cut}} = 1.562$ Å to include only a few of the computationally expensive 4-body dihedral angle interactions. The potential has a total of ten parameters, C_0, C_2,

C_4, C, δ, λ, B, η, A, and z_0, and two cutoffs r_{cut} and ρ_{cut}.

To determine the values of all the parameters that appear in the DRIP potential, we constructed a training set of energies and forces for graphene bilayers at different separation, sliding, and twisting states. The training set is generated from DFT calculations using the Vienna ab initio simulation package (VASP) [156, 157]. The exchange-correlation energy of the electrons is treated within the generalized gradient approximation (GGA) functional of Perdew, Burke and Ernzerhof (PBE) [135].

Standard density funtionals such as the local density approximation (LDA) and GGA accurately represent Pauli repulsion in interlayer interactions, but fail to capture vdW forces that result from dynamical correlations between fluctuating charge distributions.[11] To address this limitation, various approximate corrections have been proposed including the D2 method [158], the D3 method [159], the Tkatchenko and Scheffler (TS) method [160], the TS method with iterative Hirshfeld partitioning (TSIHP) method [161], the many-body dispersion (MBD) method [162], and the dDsC dispersion correction method [163]. To select a correction for the DRIP training set, we used these dispersion correction methods to compute a number of structural, energetic, and elastic properties. The results are shown in table 2.4 along with experimental values and more accurate adiabatic-connection fluctuation-dissipation theory based random-phase-approximation (ACFDT-RPA) computations that have been shown to provide a very accurate description of vdW interactions [164, 165]. The conclusion from these comparisons is that D2 and D3 provide inaccurate estimates for the layer spacing of AB graphene and graphite (d_{AB} and d_{graphite}), and TS, TSIHP, and dDsC significantly overestimate the graphite binding energy E_{graphite}. MBD provides the best overall accuracy for all considered properties and is therefore the vdW correction used in this work together with the PBE functional.

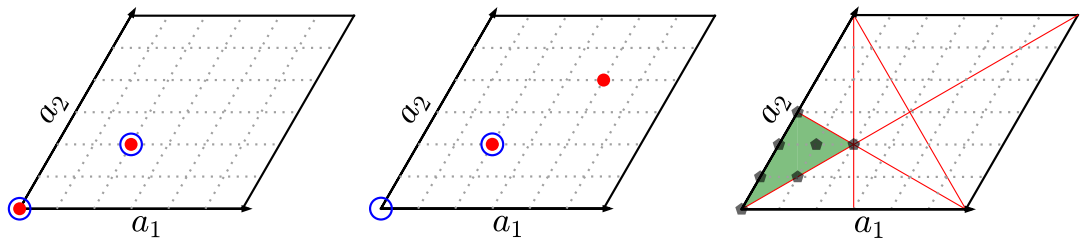
Each monolayer of the graphene bilayer is modeled as a slab with in-plane lattice constant $a = 2.46$ Å, and the supercell size in the direction perpendicular to the slab is set to 30 Å to minimize the interaction between periodic images. The sampling grid in reciprocal space is $20 \times 20 \times 1$, with an energy cutoff of 500 eV. A primitive unit cell of a graphene bilayer consists of four basis atoms. To generate a graphene bilayer with

[11] GGA predicts no binding at all at physically meaningful spacings for graphite. LDA gives the correct interlayer spacing for AB stacking, however, it underestimates the exfoliation energy by a factor of two and overestimates the compressibility [150].

Table 2.4: Properties obtained from various DFT vdW corrections compared with ACFDT-RPA and experimental results. Also included are results from various empirical potentials. The properties include: equilibrium layer spacings of bilayer graphene in AB stacking, d_{AB}, bilayer graphene in AA stacking, d_{AA}, and graphite, d_{graphite}; optimal interlayer binding energies for bilayer graphene (binding energy at the equilibrium spacing in AB stacking), E_{AB}, and graphite, E_{graphite}; energy differences between AA-stacked and AB-stacked bilayers, $\Delta E_{\mathrm{AA\text{-}AB}}$, and SP and AB stackings, $\Delta E_{\mathrm{SP\text{-}AB}}$, at a layer spacing of $d = 3.4$ Å; and the elastic modulus along the c-axis for graphite, C_{33}. All properties are computed using the in-plane lattice constant $a = 2.46$ Å.

	d_{AB} (Å)	d_{AA} (Å)	d_{graphite} (Å)	E_{AB} (meV/atom)	E_{graphite} (meV/atom)	$\Delta E_{\mathrm{AA\text{-}AB}}$ (meV/atom)	$\Delta E_{\mathrm{SP\text{-}AB}}$ (meV/atom)	C_{33} (GPa)
PBE+D2	3.248	3.527	3.218	24.84	55.20	10.35	1.16	39.12
PBE+D3	3.527	3.713	3.483	21.40	47.09	3.80	0.42	35.04
PBE+TS	3.357	3.511	3.329	36.36	82.33	7.97	1.01	68.31
PBE+TSIHP	3.379	3.529	3.350	35.73	80.42	7.48	1.22	64.73
PBE+MBD	3.423	3.638	3.398	22.63	48.96	6.17	0.69	31.64
PBE+dDsC	3.447	3.639	3.410	28.04	63.00	5.53	0.74	38.43
ACFDT-RPA	3.39[a]		3.34[b]		48[b]			36[b]
Experiment			3.34[c]		43 ± 5[d], 35 ± 10[e], 52 ± 5[f]	7.7[g]	0.86[g]	36.5[h], 38.7[i]
AIREBO	3.391	3.418	3.357	22.85	48.86	0.41	0.04	37.78
LCBOP	3.346	3.365	3.346	12.51	25.03	0.47	0.01	29.77
KC	3.374	3.602	3.337	21.60	47.44	6.07	0.44	34.45
DRIP	3.439	3.612	3.415	23.05	47.38	6.02	0.71	32.00

[a] Ref. [164].
[b] Ref. [165].
[c] Ref. [166].
[d] Ref. [167].
[e] Ref. [168].
[f] Ref. [169].
[g] Ref. [170]. Values inferred from experimental data on shear mode frequencies.
[h] Ref. [171].
[i] Ref. [172].

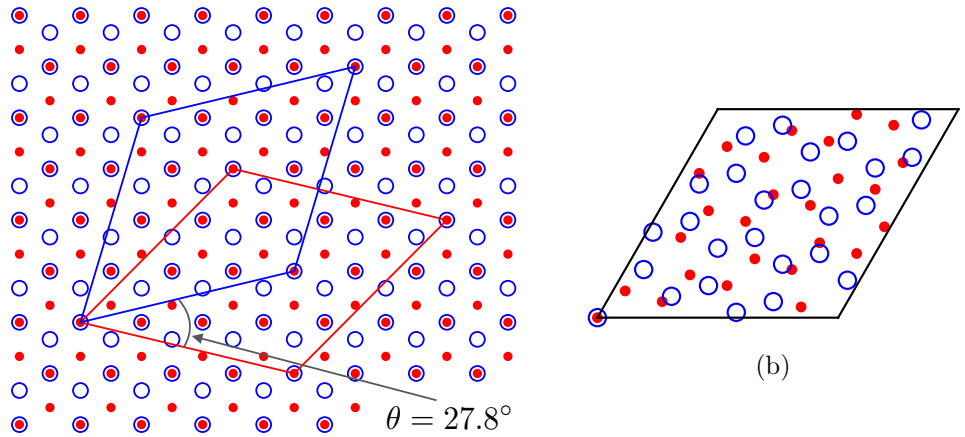


(a) (b) (c)

Figure 2.9: Primitive unit cell of a graphene bilayer: (a) AA stacking, (b) AB stacking, and (c) unique sampling region and sampling points.

different translational registry, the two atoms in the bottom layer are fixed at fractional positions $\boldsymbol{b}_1 = (0, 0, 0)$ and $\boldsymbol{b}_2 = (\frac{1}{3}, \frac{1}{3}, 0)$ relative to the graphene lattice vectors $\boldsymbol{a}_1, \boldsymbol{a}_2$, and $\boldsymbol{c}$, where $\boldsymbol{c}$ is perpendicular to the plane defined by $\boldsymbol{a}_1$ and $\boldsymbol{a}_2$ with length equal to the interlayer distance d. The other two atoms are located at $\boldsymbol{r}_1 = (p, q, 1)$ and $\boldsymbol{r}_2 = (p + \frac{1}{3}, q + \frac{1}{3}, 1)$. The two parameters $p \in [0, 1]$ and $q \in [0, 1]$ determine the translational registry. For example, the graphene bilayer is in AA stacking (figure 2.9a) when $p = 0$ and $q = 0$, and in AB stacking (figure 2.9b) when $p = \frac{1}{3}$ and $q = \frac{1}{3}$. Due to the symmetry of the honeycomb lattice, only 1/12 of the area defined by $\boldsymbol{a}_1$ and $\boldsymbol{a}_2$ needs to be sampled to fully explore all translational registry states (see the shaded region in figure 2.9c). The DRIP training set comprised the seven states indicated in the shaded region of figure 2.9c, specifically $(p, q) = (0, 0)$, $(0, \frac{1}{6})$, $(0, \frac{2}{6})$, $(0, \frac{3}{6})$, $(\frac{1}{6}, \frac{1}{6})$, $(\frac{1}{6}, \frac{2}{6})$, $(\frac{2}{6}, \frac{2}{6})$. These states include all the high-symmetry states of interest, including AA, AB, and the saddle point (SP) stacking ($p = 0, q = \frac{3}{6}$). The seven translational registry states are sampled at different layer spacings d, varying from 2.7 Å to 4.5 Å with a step size of 0.1 Å. For layer spacings larger than 4.5 Å but smaller than the cutoff $r_{\text{cut}} = 12$ Å, only bilayer graphene in AB stacking is included since the difference between the stacking states in this range is negligible (see discussion in section 2.3.2). Thus $7 \times 19 + 75 = 208$ translation configurations are included in the training set.

In addition to translation configurations, a set of twisted bilayer configurations



(a) (b)

Figure 2.10: Example of commensuration of a graphene bilayer. (a) The two layers are commensurate when rotated relative to each other by $\cos^{-1}(\frac{23}{26}) = 27.8°$, which corresponds to $m = 3, n = 7$ according to the condition in equation (2.33). (b) The resulting supercell after rotation, with 26 atoms in each layer.

are included in the training set. It is possible to construct a commensurate supercell arbitrarily close to any twisting angle according to the commensuration condition [149, 173, 174]

$$\theta = \cos^{-1}\left(\frac{3n^2 - m^2}{3n^2 + m^2}\right), \tag{2.33}$$

where m and n are any two integers satisfying $0 < m < n$. As an example, considering the AB-stacked bilayer in figure 2.10a, a commensurate bilayer can be obtained by rotating one of the layers by $\theta = 27.8°$ $(m = 3, n = 7)$ with the supercell shown in figure 2.10b. Four types of twisted bilayers with rotation angles 9.43°, 21.79°, 32.30° and 42.10° (corresponding to $(m, n) = (1, 7)$, $(1, 3)$, $(1, 2)$ and $(2, 3)$) are included in the training set. The twisted configurations were evaluated at layer spacings from 3.0 Å to 4.0 Å with a step size of 0.1 Å. Thus $4 \times 11 = 44$ twisted configurations are included in the training set. This does not include rotations for $\theta = 0°$ and $\theta = \pm 60°$ corresponding to the AB and AA stacking states, respectively, which are already included in the training set.

The parameters of the potential are optimized by minimizing the loss function in

equation (2.14), i.e.

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\sum_{i=1}^{N} w_i^{\mathrm{e}} \left[E(\boldsymbol{r}_i;\boldsymbol{\theta}) - \hat{E}_i\right]^2 + \frac{1}{2}\sum_{i=1}^{N} w_i^{\mathrm{f}} \|\boldsymbol{f}(\boldsymbol{r}_i;\boldsymbol{\theta}) - \hat{\boldsymbol{f}}_i\|^2. \tag{2.34}$$

The reference energy and forces from DFT, $\hat{E}_i$ and $\hat{\boldsymbol{f}}_i$, require explanation. Since DFT provides only the total energy and forces on atoms due to both intralayer and interlayer interactions it is necessary to separate out the interlayer contributions when constructing the training set. This is accomplished as follows. For configuration i, first the total energy and forces of the bilayer are obtained from DFT: $\hat{E}_i^{\text{bilayer}}$, $\hat{\boldsymbol{f}}_i^{\text{bilayer}}$. Then each monolayer is computed separately by removing all atoms from the other monolayer. Thus, there will be two energies, $\hat{E}_i^{\text{layer 1}}$ and $\hat{E}_i^{\text{layer 2}}$, and two forces, $\hat{\boldsymbol{f}}_i^{\text{layer 1}}$ and $\hat{\boldsymbol{f}}_i^{\text{layer 2}}$ (although each force vector will only contain nonzero components for the atoms belonging to its monolayer). The DFT interlayer energy and forces appearing in equation (2.34) are then defined as:

$$\hat{E}_i = \hat{E}_i^{\text{bilayer}} - \hat{E}_i^{\text{layer 1}} - \hat{E}_i^{\text{layer 2}}, \tag{2.35}$$

$$\hat{\boldsymbol{f}}_i = \hat{\boldsymbol{f}}_i^{\text{bilayer}} - \hat{\boldsymbol{f}}_i^{\text{layer 1}} - \hat{\boldsymbol{f}}_i^{\text{layer 2}}. \tag{2.36}$$

In the present case, the training set includes $N = 252$ configurations. Both the energy weight w_i^{e} and force weight w_i^{f} are set to 1. The optimization was carried out using the KIM-based learning-integrated fitting framework (KLIFF) [69] with a geodesic Levenberg–Marquardt (LM) minimization algorithm [66, 103, 104]. The objective is to find the set of parameters $\boldsymbol{\theta}$ that minimizes $\mathcal{L}(\boldsymbol{\theta})$. The optimal parameter set identified by this process and preset cutoffs are listed in table 2.5.

2.3.2 Testing of model

We performed an extensive set of calculations to test the ability of DRIP to reproduce its training set (described in section 2.3.1), and test its transferability to configurations outside the training set. The calculations using the potential were performed with large-scale atomic/molecular massively parallel simulator (LAMMPS) [71,137] and DFT calculations with VASP [156, 157]. Periodic boundary conditions are applied in both in-plane directions, and the in-plane lattice constant is fixed at $a = 2.46$ Å. The setup

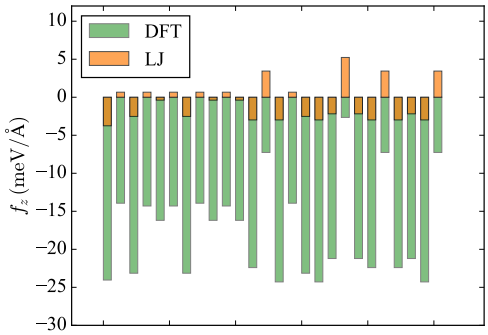
Table 2.5: DRIP parameters obtained by minimizing the loss function and preset cutoffs.

Parameter	Value	Parameter	Value
C_0 (meV)	11.598	B (meV)	7.6799
C_2 (meV)	12.981	η (1/Å)	1.1432
C_4 (meV)	32.515	A (meV)	22.216
C (meV)	7.8151	z_0 (Å)	3.3400
δ (Å)	0.83679	r_{cut} (Å)	12
λ (1/Å)	2.7158	ρ_{cut} (Å)	1.562

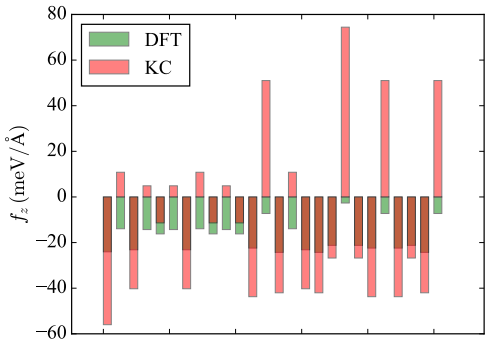
for the DFT computations is the same as that used for generating the training set in section 2.3.1.

Figure 2.11 shows the unrelaxed forces on the atoms in the bottom layer of the twisted bilayer shown in figure 2.10 with a layer spacing of $d = 3.4$ Å. There are 26 atoms in the bottom layer. For each, the out-of-plane force (z-component) is displayed as a bar. The plot compares the results of LJ, KC and DRIP with DFT. For the LJ potential, the parameterization in the AIREBO potential is used. The DRIP forces are in very good agreement with DFT, whereas the LJ potential yields almost zero forces, and the KC potential greatly overestimates the forces. (Note that the force ranges in the three panels are different). The force on the central atom when twisting a bilayer obtained from DRIP (denoted as 1 in figure 2.7) is displayed in figure 2.6b as a function of rotation. The results are in agreement with DFT, indicating that the dihedral modification in DRIP successfully addresses the deficiency of the KC potential.

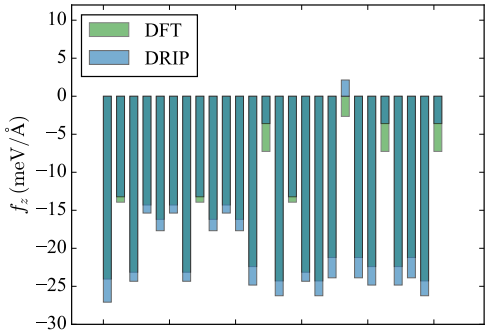
To investigate the accuracy of the IPs in a dynamical setting, trajectories are generated at a temperature of 300 K using *ab initio* molecular dynamics (AIMD) for bilayers in AA and AB stackings, and the twisted bilayer shown in figure 2.10. For each configuration along the trajectories, the DFT forces due to interlayer interactions are computed using the procedure defined in equation (2.36) and explained above. Next, LAMMPS is used to compute the LJ, KC and DRIP interlayer forces for the AIMD configurations. The error in the potential forces is shown in figure 2.12. Each dot in the plot represents one atom pulled from one of the configurations along the AIMD trajectories. The horizontal coordinate in the plot is the magnitude of the in-plane component (left panels) and out-of-plane component (right panels) of the DFT interlayer force acting on the atom. The force is separated in this way because the in-plane component is



(a)



(b)



(c)

Figure 2.11: Out-of-plane component of the forces on the 26 atoms in the bottom layer of the twisted bilayer shown in figure 2.10 (each represented as a bar) computed from DFT and the (a) LJ potential, (b) KC potential, and (c) DRIP model. The layer spacing is 3.4 Å.

significantly smaller than the out-of-plane component. (Note that this is only the force due to interlayer interactions. The force due to intralayer bonding is not included.) The vertical coordinate is the magnitude of the difference between the potential and DFT force vectors for that atom. We see that the in-plane force error for LJ aligns with the diagonal, i.e. the error equals the DFT force, which means that LJ predicts an in-plane force close to zero. This is because LJ provides a poor model for the anisotropic overlap of electronic orbitals between adjacent layers and thus has almost no barrier for relative sliding. The KC model performs better in the sense that it predicts resistance to sliding, however the overall accuracy in forces is poor. In contrast, DRIP provides consistently accurate in-plane forces across the range of DFT forces with errors less than 20 meV/Å. For the out-of-plane component both LJ and DRIP perform comparably providing good accuracy across the range of DFT forces, whereas the KC model again shows poor accuracy with very large errors in some cases.

Next, we consider energetics. The interlayer binding energy E_b of a graphene bilayer as a function of layer spacing d is shown in figure 2.13 for AB and AA stackings and the twisted configuration shown in figure 2.10. The LJ potential (figure 2.13a) cannot distinguish these states and gives nearly identical binding energy versus layer spacing curves for all three. Both KC (figure 2.13b) and DRIP (figure 2.13c) correctly capture the energy differences between the three stacking states. For all three IPs, the twisted bilayer curve lies between the other two, which is expected since the AB and AA stackings are minimum and maximum energy states. Also notable is that at large layer spacing, the curves for all three stacking states merge since registry effects due to π-orbital overlap become negligible and interactions are dominated by vdW attraction, which are the same for all three states and captured equally well by all three IPs.

A more complete view of the interlayer energetics is obtained by considering the generalized stacking fault energy (GSFE) surface obtained by sliding one layer relative to the other while keeping the layer spacing fixed. Figure 2.14 shows the results for a layer spacing of $d = 3.4$ Å calculated using DRIP and DFT. DRIP is in quantitative agreement with DFT results. The KC GSFE has a similar appearance and the LJ GSFE is nearly flat. The KC and LJ results are not included for brevity, but the energies of the three potentials along the dashed line in the left panel of figure 2.14 are displayed in figure 2.6a.

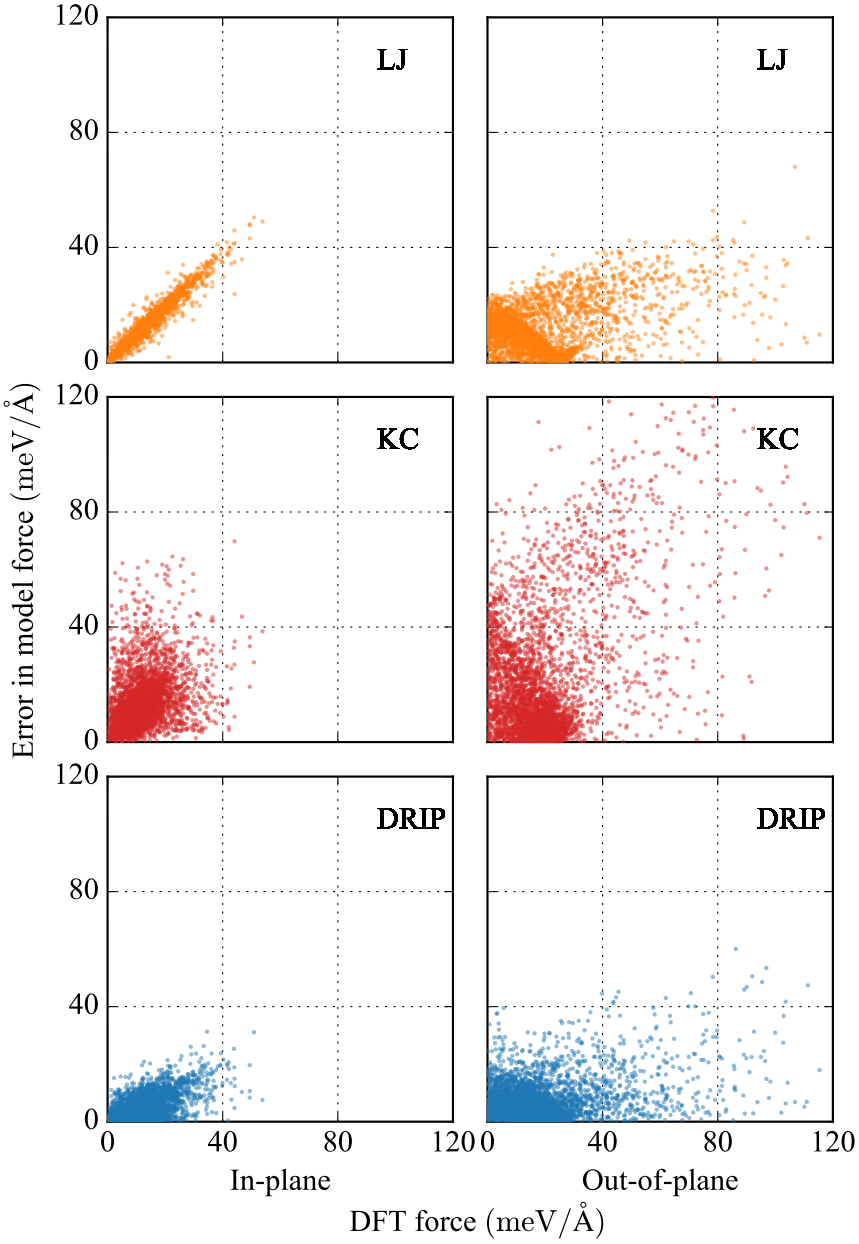
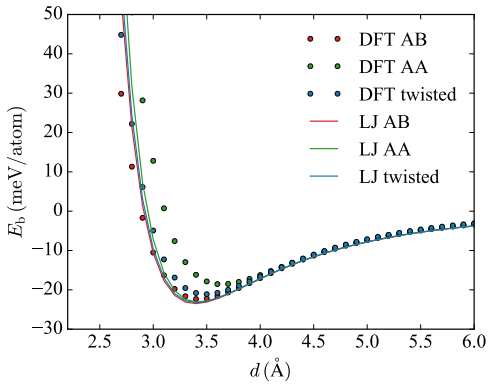
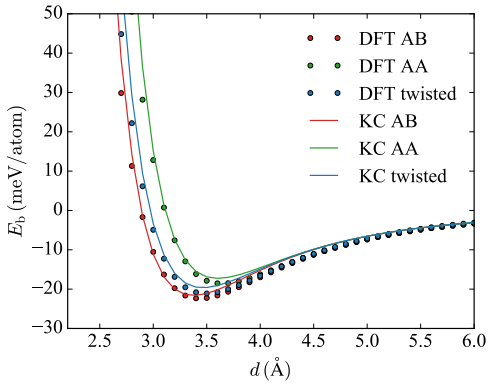


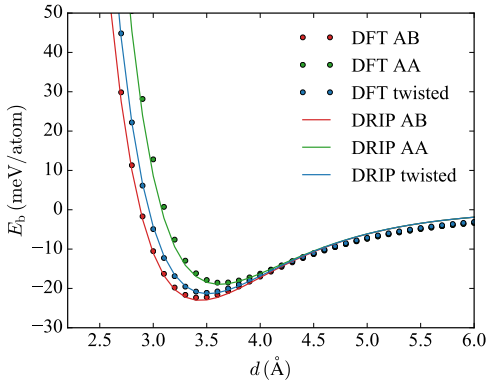
Figure 2.12: Deviation of potential forces from DFT results due to interlayer interactions. The configurations are taken from three AIMD trajectories at 300 K.



(a)



(b)



(c)

Figure 2.13: Interlayer binding energy E_b of a graphene bilayer versus layer spacing d for AA stacking, AB stacking, and a twisted bilayer with rotation angle $\theta = 27.8°$ (see figure 2.10) using (a) LJ potential, (b) KC potential, and (c) DRIP model, compared to DFT results.

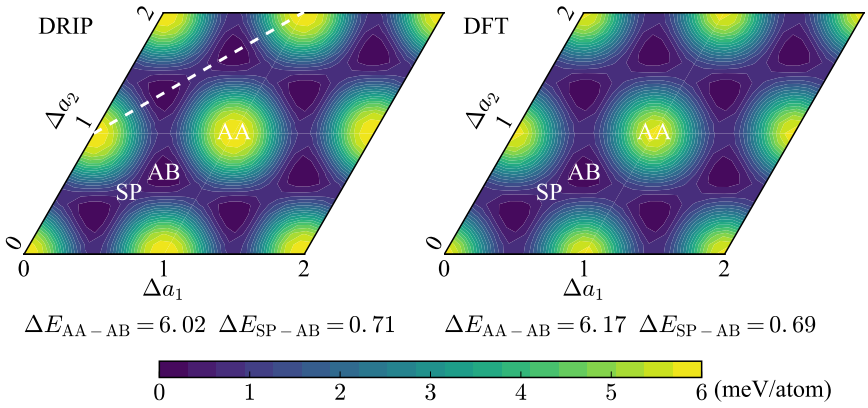


Figure 2.14: The GSFE obtained by sliding one layer relative to the other at a fixed layer spacing of $d = 3.4$ Å. The energy is relative to the AB state, which is -22.98 meV/atom for DRIP (on the left) and -22.33 meV/atom for DFT (on the right). The sliding parameters $\Delta\boldsymbol{a}_1$ and $\Delta\boldsymbol{a}_2$ are in units of in-plane lattice constant $a = 2.46$ Å.

As a final test, table 2.4 shows the predictions of DRIP for a number of structural, energetic, and elastic properties. The table also includes results for the long-range carbon bond order potential (LCBOP) [84] and AIREBO [148] potentials, as well as DFT and experimental results as described in section 2.3.1. The LCBOP potential uses two Morse [153] type terms to model long-range interactions, and LJ [77] is used in the AIREBO potential. The properties of the DRIP model are in good agreement with the PBE+MBD DFT computations with which the training set was generated.

2.3.3 Applications

To further compare the predictions of the KC potential and DRIP, we carried out two large-scale simulations, beyond the capability of DFT: (1) structural relaxation in a twisted graphene bilayer, and (2) exfoliation of a graphene layer off graphite. In these simulations, the interlayer interactions are modeled using either KC or DRIP, and the REBO potential is used to model the intralayer interactions.

Structural relaxation of a twisted graphene bilayer

The electronic properties of stacked 2D materials can be manipulated by controlling the relative rotation between the layers, which in turn leads to different structural

relaxation. A prototypical problem is the twisting of a graphene bilayer. The bilayer is created by rotating one layer relative to the other by $\theta = 0.82°$, setting $(m, n) = (1, 81)$ as discussed in section 2.3.1. The out-of-plane relaxation δ of an atom is obtained by subtracting the mean out-of-plane coordinates of all atoms in the top layer from the out-of-plane coordinate of that atom:

$$\delta_i = z_i - \frac{1}{N_\mathrm{a}} \sum_{j=1}^{N_\mathrm{a}} z_j \tag{2.37}$$

where z_i is the out-of-plane coordinate of atom i in the top layer and $N_\mathrm{a} = 9842$ is the number of atoms in the top layer.[12]

The out-of-plane relaxation of the twisted bilayer is plotted in figure 2.15. The results of DRIP and KC are qualitatively similar. The bright spots correspond to high-energy AA stacking, the long narrow ribbons correspond to SP stacking, and the triangular regions correspond to alternating AB and BA stackings. It has been shown that the formation of this structure is due to local rotation at AA domains [175]. Quantitatively, however, the two IPs give different out-of-plane relaxation, especially at the peaks as seen in figure 2.15b. The peak value predicted by DRIP is 0.076 Å, which is 26% smaller than the KC potential value of 0.103 Å. This difference at the peaks could lead to significant differences in electronic properties because twisted graphene bilayers develop highly-localized states around AA-stacked regions for small twist angles [176].

Exfoliation of graphene from graphite

Graphene can be prepared by exfoliating graphite. In this process, the vdW attraction between layers is overcome by peeling a single layer off a graphite crystal. A method as simple as sticking scotch tape to graphite and applying an upward force can be used [15]. To simulate this process, one edge of the top layer of a graphite crystal is pulled up under displacement control conditions as illustrated in figure 2.16a. The atoms at the left end of the top layer are displaced in the z-direction according to $d = d_0 + 0.2k$, where $d_0 = 3.35$ Å is the initial layer spacing, and $k = 0, 1, \ldots, 99$ is the step number. At each step k, once the displacement is applied to the left atoms, the remaining atoms

[12] Using the atoms in the bottom layer will yield the same results because the relaxed structure of the bottom layer and the top layer are identical.

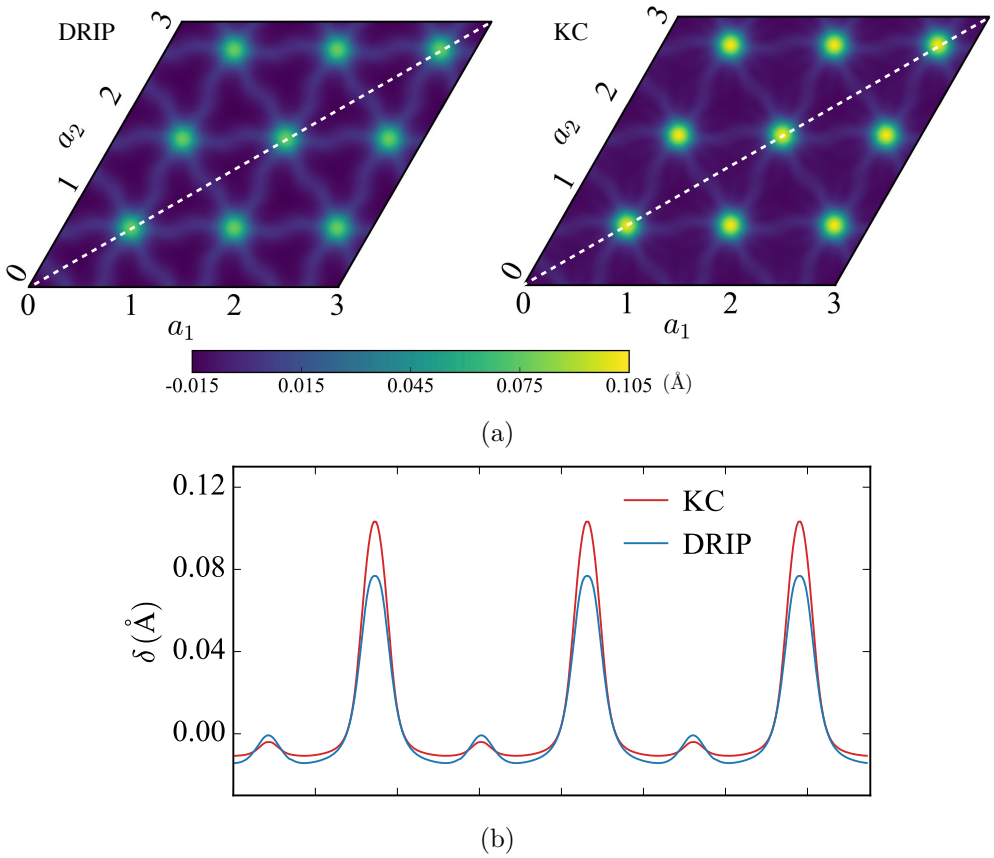
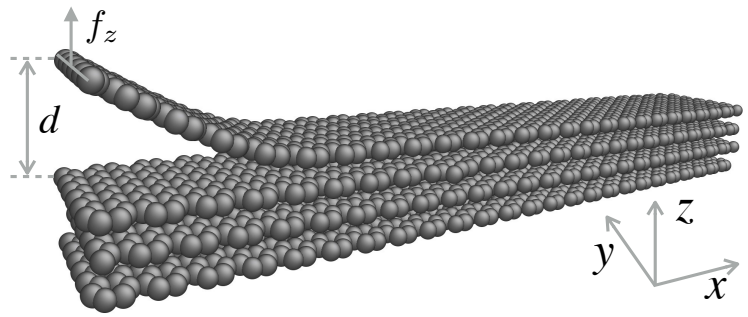
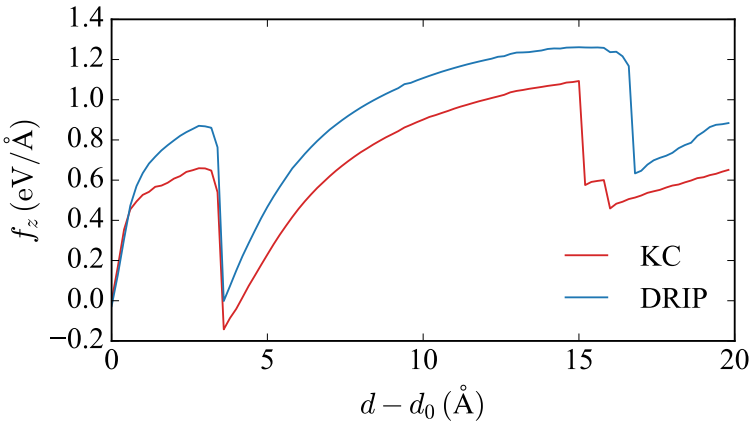


Figure 2.15: Out-of-plane relaxation in a twisted bilayer with a relative rotation of $\theta = 0.82°$. (a) Contour plot obtained from the DRIP model and the KC potential, and (b) relaxation along the diagonal indicated by the dashed line in panel (a). The bilayers shown in the figure corresponds to 3×3 supercells used in the computation, i.e. a_1 and a_2 are in units of in-plane lattice constant $a = 2.46$ Å.



(a)



(b)

Figure 2.16: (a) Schematic demonstrating the process of peeling a graphene layer off graphite, and (b) the normal force, f_z, needed to peel the top layer as a function of the displacement at the left end of the top layer, $d - d_0$. The armchair direction of graphite is aligned with the x-axis. The initial layer spacing is $d_0 = 3.35$ Å.

in the top layer are relaxed. The substrate (bottom three layers) is kept rigid during this process. The system contains 600 atoms in each layer of size 105.83 Å and 14.76 Å in the x and y directions, respectively. The system is periodic in the y direction, and non-periodic the other two directions.

The normal force, f_z, needed to pull the left end of the top layer is plotted in figure 2.16b. Both KC and DRIP give qualitatively similar results. The force first increases as the left end is pulled up and then exhibits a sudden drop at about 3 Å. The normal force has two contributions: (1) interlayer interactions with atoms in the substrate; and (2) covalent-bonded interactions with other atoms in the top layer. The former is almost unchanged before and after the load drop, therefore the drop is mainly due to the in-plane interactions in the top layer. Before the load drop, the right-end of the top layer is trapped in a local minimum created by the substrate (similar to the one denoted as AB in figure 2.14, although there we only consider a graphene bilayer), and consequently as the left end is pulled up, the top layer experiences an increasing axial strain. At about 3 Å, the right-end of the top layer snaps into an adjacent local minimum by moving in the negative x direction. As a result, the axial strain in the top layer is released and the load is reduced. The same explanation applies to the load drop at a displacement of about 15 Å, and it is expected to continue to occur periodically with continued pulling.

As for the results in bilayer relaxation, KC and DRIP are in qualitative agreement, but there are quantitative differences. The KC potential predicts an initial peeling load of about 0.65 eV/Å, which is about 75% of the 0.87 eV/Å value predicted by DRIP. The second snap-through occurs at a displacement of 16.6 Å for DRIP, and at 15.0 Å for KC.

2.3.4 Summary

The interlayer interactions in stacked 2D materials play an important role in determining the mathematicality of many nanodevices. For graphitic structures, the two-body pairwise LJ potential is too smooth to model the energy corrugation in different stacking states. The registry-dependent KC potential improves on this and correctly captures the energy variation, but fails to yield reasonable forces. In particular, the KC model

does not distinguish forces on atoms in the AA and AB stacking states that are different in DFT calculations. The KC model is also discontinuous at the cutoff, which can lead to difficulties in energy minimization and loss of energy conservation in dynamic applications.

To address these limitations, we developed a new potential for graphitic structures based on the KC model. The DRIP has a smooth cutoff and includes a dihedral-angle-dependent term to distinguish different stacking states and obtain accurate forces. The potential parameters were determined by training on a set of energies and forces for graphene bilayers at different layer spacing, sliding, and twisting, computed using GGA-DFT calculations, augmented with the MBD dispersion correction to account for the long-range vdW interactions.

To test the quality of DRIP, we employed it to compute energetics, forces, and structural, and elastic properties for a graphene bilayer in different states and graphite. The validation tests show that compared with first-principles results:

1. DRIP correctly predicts the equilibrium layer spacing, interlayer binding energy, and generalized stacking fault energy of a graphene bilayer, as well as the equilibrium layer spacing of graphite.

2. DRIP underestimates the c-axis elastic modulus C_{33} of graphite by about 10% relative to ACFDT-RPA and experiments, but this result is in good agreement with PBE+MBD to which DRIP was fit.

3. DRIP provides more accurate forces than the KC model across the entire range of bilayer rotations and in particular distinguishes the forces in the AA and AB states that the KC potential cannot.

In two large-scale applications, not amenable to DFT calculations, we showed that DRIP and KC agree qualitatively, but differ quantitatively by 26% in the out-of-plane relaxation of a twisted graphene bilayer, and by 23% in the normal force required to peel one graphene layer off graphite.

The added four-body dihedral-angle-dependent correction in DRIP is very short-ranged ($\rho_{\text{cut}} = 1.562$ Å) and therefore the computational overhead relative to KC is small. In fact, for the large-scale applications (bilayer relaxation and peeling) described

in section 2.3.3, DRIP was actually faster than KC in terms of the overall computation time due to improved convergence.

Although DRIP was parameterized against a training set consisting of graphene bilayers, it can be used to describe interlayer interactions for other systems such as graphite and multi-walled carbon nanotubes where the carbon atoms are arranged in layers. This IP only provides a description of the interlayer interactions, and therefore must be used together with a companion model that provides the intralayer interactions, such as the Tersoff [113, 147] or REBO [83, 114] potentials.

Chapter 3

Machine Learning Potentials

In chapter 2, we discussed the procedures to develop an interatomic potential (IP): first devise a mathematical form for the IP and then obtain the parameters in the mathematical form by solving an optimization problem. It is possible to place the IP development problem in a broader context of the theory of *machine learning*. Murphy defines machine learning as [177]:

> A set of methods that can automatically detect patterns in data, and then use the uncovered patterns to predict future data, or to perform other kinds of decision making under uncertainty (such as planning how to collect more data!).

Machine learning uses the tools of probability theory to solve such pattern recognition problems, typically divided into two types: *unsupervised learning* and *supervised learning*.

In unsupervised learning, we are only given a data set of inputs $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}$, without the corresponding outputs $\{\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_N\}$ as in supervised learning to be discussed below. It can be further subdivided into three categories based on the goal: *clustering*, *density estimation*, and *visualization* [96]. Clustering refers to the discovery of groups of similar examples within the data, density estimation refers to the determination of the distribution of data within the input space, and visualization refers to the projection of the data from a high-dimensional space down to a low-dimensional space (usually two or three dimensions). Unsupervised learning problems are not well-defined,

since there is no obvious error metric to test the predictions made by a model [177].

In supervised learning, besides the inputs $\{\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_N\}$, we also have their corresponding outputs $\{\boldsymbol{y}_1, \boldsymbol{y}_2, \ldots, \boldsymbol{y}_N\}$. If the outputs are a finite number of discrete categories and the aim is to assign each input to one of the categories, the task is called *classification*; if the desired outputs are continuous and the aim is to find a continuous function to map the inputs to the outputs, then the task is called *regression*.

The IP development problem belongs to regression in supervised learning. The inputs are the coordinates and species of atoms in atomic configurations, and the outputs are the total potential energies (and other properties that can be computed from atomistic simulations) of the configurations. Although the IPs discussed in chapter 2 fall within the definition of machine learning, we still call them physics-based IPs because the mathematical forms of such IPs are designed based on the underlying physics in the materials system, making it difficult to get a better model upon observation of more data. On the contrary, we call machine learning IPs the type of models that train general-purpose functions (e.g. Gaussian process (GP) and neural network (NN)) against a large amount of data and are systemically improvable upon observation of more data.

In the last decade, machine learning IPs [178–182] have been shown very successful for many materials systems ranging from organic molecules [180] to alloys [182]. In these methods, a training set of first-principles data is interpolated using suitable basis functions to obtain the energy and forces of intermediate configurations. In effect, they attempt to "learn" the quantum mechanical Schrödinger equation directly from the training set with minimal supervision. Given a description of the atomic environments (the local environments of an atom, discussed in details in section 3.1), a machine learning IP returns the atomic energy (energy of an atom), and the total energy is the sum of the atomic energies of all the atoms in the system. This separates the material physics, which is entirely contained in the training set, from the mathematical interpolation performed by the machine learning algorithm. Once properly tuned, machine learning IPs have the advantage that, in principle, they can describe arbitrary bonding states with a sufficiently dense training set.

In this chapter, we first review some methods to represent atomic environments, upon which machine learning regression methods are applied to model the potential

energy. Then we discuss several widely used regression methods for IPs. Finally, we present a neural network interatomic potential (NNIP) trained to model both the interlayer and intralayer interactions in multilayer graphene structures.

3.1 Representation of atomic environments

The appropriate representation of atomic environments is a crucial ingredient in modern computational chemistry, condensed matter physics, and materials science. It is widely used in applications such as structure search of molecules, molecular dynamics (MD) simulations of phase transitions, and, of course, interatomic potential (IP) constructions [183]. IPs are typically constructed as a sum of the energies of individual atoms[1] that are built on a carefully chosen representation of atomic environments. Therefore, selecting an appropriate representation of atomic environments is critical in designing IPs.

Atomic environments are typically represented as a vector of real values, called the *descriptor* (or *fingerprint*). Concatenating the position of all atoms in a Cartesian coordinate system provides a simple and unequivocal descriptor; however, it does not satisfy the invariant requirements by the laws of physics (item 1 discussed below). A good descriptor needs to retain the faithfulness of the Cartesian representation and should also be:

1. *invariant* with respect to *translation*, *rotation*, and *inversion* of the system (i.e. choice of the reference frame), and *permutation* of atoms of the same species. In fact, these are the invariant requirements that all IPs (physics-based and machine learning) must adhere to as discussed in section 2.1. In principle, one can let the machine learning regression algorithm to learn these requirements by providing an enormous amount of training data. This is obviously impossible to achieve in practice; therefore, these requirements are built into the representation and we ensure that the machine learning regression algorithm does not break these requirements.

[1] As discussed in section 2.1.1, although the total potential energy of some IPs is obtained as a sum of the potential energies of bond lengths, bond angles, etc., it can nevertheless be transformed as a sum over atomic energies.

2. *complete*. A system of descriptors is said to be complete if it uniquely determines the atomic environments, up to the above mentioned symmetries; it is over-complete if a proper subset of is complete. In other words, a representation is complete when there exists a one-to-one mapping (i.e. a bijection) between the atomic environments and the invariant descriptors. It is not complete if more than one atomic environments are mapped to the same invariant descriptors (i.e. surjection). It is over-complete if more than one invariant descriptors are assigned to a given atomic environments, but different atomic environments never have identical descriptors.

3. *continuous* and *differentiable*. Having discontinuities in the representation makes it extremely challenging, if not impossible, for a machine learning algorithm to recover the smoothness of a potential energy surface. IPs should be differentiable so as to compute certain quantities in atomistic simulations. For example, atomic forces require first derivatives of potential energy with respect to atomic coordinates and derivatives up to the 4th order are necessary for some physical properties, like elastic constants, computed using lattice dynamics [65].

4. *independent*. For crystalline solids and amorphous materials, the number of neighboring atoms that fall into the local environments of an atom (typically defined by a cutoff sphere with its center located at the atom) needs to be allowed to vary, because, in the course of an atomistic simulation, atoms can enter or leave the local environments. For the purpose of function fitting, it is preferable and more practical that the length of the descriptor vector is independent of the number of neighbors.

In this section, we survey some of the representative atomic environment descriptors developed in recent years, including the Coulomb matrix representation, the symmetry functions representation, the bispectrum representation, and the many-body tensor representation.

3.1.1 Coulomb matrix

The Coulomb matrix [180,184,185] representation is built on the same information that enters the Hamiltonian for an electronic structure calculation, namely, the coordinates

and atomic numbers of atoms. The environment of an atom A is represented as

$$M_{ij} = \begin{cases} 0.5 Z_i^{2.4} & \text{for } i = j \\ \frac{Z_i Z_j}{\|\boldsymbol{r}_i - \boldsymbol{r}_j\|} & \text{for } i \neq j \end{cases}, \tag{3.1}$$

where $\boldsymbol{r}_i$ and Z_i denote the Cartesian coordinates and atomic number of atom i, respectively, and the indices i and j run over atom A itself and its neighboring atoms within its cutoff sphere. The diagonal elements represent an exponential fit of the free atom potential energy to its nuclear charge (atomic number), and the off-diagonal elements encode the Coulomb repulsion between nuclear charges (atomic numbers) of atoms i and j.

The Coulomb matrix retains the translational, rotational, and inversional symmetries, but fails the permutational symmetry requirement as permutating two atoms with the same species change the order of rows and columns. To avoid the dependence on atom ordering, one can diagonalize the Coulomb matrix and use the ordered eigenvalues as the descriptor vector. For matrices with different size,[2] the eigenvalues of the smaller systems can be padded by zeros [180]. In such the Coulomb matrix representation satisfies requirements 1, 3, and 4; however, it is not complete since distinct atomic structures can have exactly the same descriptor [186].

3.1.2 Symmetry functions

The symmetry functions [178, 187] employ two sets of radial functions and angular functions to represent atomic environments. Radial functions are constructed as sums of two-body terms, and angular functions additionally contain sums of three-body terms. Three types of functions are proposed to describe the radial environments of an atom i:

$$G_i^1 = \sum_{j \neq i} f_\mathrm{c}(r_{ij}), \tag{3.2}$$

$$G_i^2 = \sum_{j \neq i} e^{-\eta (r_{ij} - R_\mathrm{s})^2} \cdot f_\mathrm{c}(r_{ij}), \tag{3.3}$$

[2] The Coulomb matrix for an atom with fewer neighbors has a smaller size than that with more neighbors.

and

$$G_i^3 = \sum_{j \neq i} \cos(\kappa r_{ij}) \cdot f_c(r_{ij}). \tag{3.4}$$

Two types of functions are proposed for the angular environments of an atom i:

$$G_i^4 = 2^{1-\zeta} \sum_{j \neq i} \sum_{\substack{k>j \\ k \neq i}} (1 + \lambda \cos\theta_{jik})^\zeta \cdot e^{-\eta(r_{ij}^2 + r_{ik}^2 + r_{jk}^2)} f_c(r_{ij}) \cdot f_c(r_{ik}) \cdot f_c(r_{jk}), \tag{3.5}$$

and

$$G_i^5 = 2^{1-\zeta} \sum_{j \neq i} \sum_{\substack{k>j \\ k \neq i}} (1 + \lambda \cos\theta_{jik})^\zeta \cdot e^{-\eta(r_{ij}^2 + r_{ik}^2 + r_{jk}^2)} f_c(r_{ij}) \cdot f_c(r_{ik}). \tag{3.6}$$

In the symmetry functions, r_{ij} denotes the Euclidean distance between atoms i and j, θ_{jik} is the angle between bonds j–i and k–i with the vertex at atom i, and η, R_s, κ, ζ and λ are hyperparameters. The cutoff function f_c is given by

$$f_c(r) = \begin{cases} \frac{1}{2}\left[\cos\left(\frac{\pi r}{r_{\text{cut}}}\right) + 1\right] & \text{for } r \leq r_{\text{cut}} \\ 0 & \text{for } r > r_{\text{cut}} \end{cases}, \tag{3.7}$$

where r_{cut} is the cutoff distance, atoms beyond which do not contribute to the local environments.

The symmetry functions take only pair distances between atoms as the arguments,[3] thus automatically satisfying the translational, rotational, and inversional symmetry requirement [17]. They are also permutationally symmetric since these functions are constructed by summing over all the bond lengths and bond angles within the cutoff sphere. They are obviously continuous and differentiable. Each choice of the hyperparameter η in G_i^1 constitutes to one component in the descriptor vector, and the same applies to G_i^2, G_i^3, G_i^4, and G_i^5. Consequently, the length of the descriptor vector is equal to the number of hyperparameter sets, independent of the number of neighboring atoms. The symmetry functions are not complete because interactions of orders higher than two-body and three-body are ignored. In sum, the symmetry functions satisfy requirements 1, 3, and 4, but not 2 discussed in section 3.1.

[3] The bond angles $\cos_{jik}$ in G_i^4 and G_i^5 can be easily rewritten as a function of pair distances using the law of cosines.

3.1.3 Bispectrum

We start by defining a spatial distribution of the neighbors in an atomic environment by a three-dimensional (3D) *atomic neighborhood density function*,

$$\rho(\boldsymbol{r}) = \sum_i w_{Z_i} \delta(\boldsymbol{r} - \boldsymbol{r}_i), \tag{3.8}$$

where the index i runs over all the atoms within some cutoff distance, w_{Z_i} is a weight factor assigned according to the atomic species of atom i, $\boldsymbol{r}_i$ is the distance vector connecting the center atom and its neighboring atom i,[4] and $\delta(\cdot)$ is the Dirac delta function.

Equation (3.8) can be expanded in terms of spherical harmonics,

$$\rho(\hat{\boldsymbol{r}}) = \sum_{l=0}^{\infty} \sum_{m=-l}^{l} c_{lm} Y_{lm}(\hat{\boldsymbol{r}}), \tag{3.9}$$

where $\hat{\boldsymbol{r}}$ is the point on the unit sphere corresponding to the direction of the vector $\boldsymbol{r}$, and $\rho(\hat{\boldsymbol{r}})$ is the projection of $\rho(\boldsymbol{r})$ onto the unit sphere. The coefficients c_{lm} can be obtained as

$$c_{lm} = \langle \rho | Y_{lm} \rangle = \sum_i Y_{lm}(\hat{\boldsymbol{r}}_i), \tag{3.10}$$

where $\langle \cdot | \cdot \rangle$ denotes function inner product. Applying a rigid-body rotation $R \in SO(3)$ to a spherical harmonic function Y_{lm} can be expressed as a linear combination of spherical harmonics,

$$RY_{lm} = \sum_{m'=-l}^{l} D^l_{mm'} Y_{lm'}, \tag{3.11}$$

where $\boldsymbol{D}^l$ is the Wigner matrix [188], whose elements are given by

$$D^l_{mm'} = \langle Y_{lm} | R | Y_{lm'} \rangle. \tag{3.12}$$

[4] Note, here we relax the notation to use $\boldsymbol{r}_i$ to indicate the distance vector, instead of the Cartesian coordinates of atom i.

The Wigner matrix is unitary, i.e.

$$(\boldsymbol{D}^l)^\dagger \boldsymbol{D}^l = \boldsymbol{I}, \tag{3.13}$$

where $(\cdot)^\dagger$ denotes complex conjugate transpose of a matrix, and $\boldsymbol{I}$ is the identity. The projected atomic neighborhood density function in equation (3.9) transforms under rotation R as,

$$\begin{aligned} R\rho &= R\sum_{l=0}^{\infty}\sum_{m=-l}^{l} c_{lm}Y_{lm} \\ &= \sum_{l=0}^{\infty}\sum_{m=-l}^{l} c_{lm}RY_{lm} \\ &= \sum_{l=0}^{\infty}\sum_{m=-l}^{l}\sum_{m'=-l}^{l} c_{lm}D^l_{mm'}Y_{lm'} \\ &= \sum_{l=0}^{\infty}\sum_{m'=-l}^{l}\left(\sum_{m=-l}^{l} c_{lm}D^l_{mm'}\right)Y_{lm'}, \end{aligned} \tag{3.14}$$

where in the third equality we use equation (3.11). Comparing equation (3.14) and equation (3.9), we see that the expansion coefficients transform under a rotation R as

$$c_{lm} \xrightarrow{R} \sum_{m'=-l}^{l} c_{lm'}D^l_{m'm} \iff \boldsymbol{c}_l \xrightarrow{R} \boldsymbol{D}^l\boldsymbol{c}_l. \tag{3.15}$$

As a result, the tensor product of the expansion coefficient transforms under a rotation R as

$$\boldsymbol{c}_{l_1} \otimes \boldsymbol{c}_{l_2} \xrightarrow{R} (\boldsymbol{D}^{l_1} \otimes \boldsymbol{D}^{l_2})(\boldsymbol{c}_{l_1} \otimes \boldsymbol{c}_{l_2}). \tag{3.16}$$

The tensor product of two Wigner matrices can be decomposed into a direct sum of Wigner matrices [183]:

$$\boldsymbol{D}^{l_1} \otimes \boldsymbol{D}^{l_2} = (\boldsymbol{C}^{l_1l_2})^\dagger \left[\bigoplus_{l=|l_1-l_2|}^{l_1+l_2} \boldsymbol{D}^l\right] \boldsymbol{C}^{l_1l_2}, \tag{3.17}$$

where $\left[\bigoplus_{l=|l_1-l_2|}^{l_1+l_2} \boldsymbol{D}^l\right]$ is a block diagonal matrix

$$\left[\bigoplus_{l=|l_1-l_2|}^{l_1+l_2} \boldsymbol{D}^l\right] = \begin{bmatrix} \boldsymbol{D}^{|l_1-l_2|} & & & \\ & \boldsymbol{D}^{|l_1-l_2|+1} & & \\ & & \ddots & \\ & & & \boldsymbol{D}^{|l_1+l_2|} \end{bmatrix}, \tag{3.18}$$

and $\boldsymbol{C}^{l_1 l_2}$ is the Clebsch-Gordan coefficients matrix [189]. The Clebsch-Gordan coefficients matrices are unitary (i.e. $(\boldsymbol{C}^{l_1 l_2})^\dagger \boldsymbol{C}^{l_1 l_2} = \boldsymbol{I}$). Combining equations (3.16) and (3.17), we see that $\boldsymbol{C}^{l_1 l_2}(\boldsymbol{c}_{l_1} \otimes \boldsymbol{c}_{l_2})$ transforms under a rotation R as

$$\begin{aligned} \boldsymbol{C}^{l_1 l_2}(\boldsymbol{c}_{l_1} \otimes \boldsymbol{c}_{l_2}) &\xrightarrow{R} \boldsymbol{C}^{l_1 l_2}(\boldsymbol{C}^{l_1 l_2})^\dagger \left[\bigoplus_{l=|l_1-l_2|}^{l_1+l_2} \boldsymbol{D}^l\right] \boldsymbol{C}^{l_1 l_2}(\boldsymbol{c}_{l_1} \otimes \boldsymbol{c}_{l_2}) \\ &= \left[\bigoplus_{l=|l_1-l_2|}^{l_1+l_2} \boldsymbol{D}^l\right] \boldsymbol{C}^{l_1 l_2}(\boldsymbol{c}_{l_1} \otimes \boldsymbol{c}_{l_2}). \end{aligned} \tag{3.19}$$

The form of equation (3.19) implies that $\boldsymbol{C}^{l_1 l_2}(\boldsymbol{c}_{l_1} \otimes \boldsymbol{c}_{l_2})$ itself possesses a diagonal matrix structure, and thus we can rewrite it as

$$\boldsymbol{C}^{l_1 l_2}(\boldsymbol{c}_{l_1} \otimes \boldsymbol{c}_{l_2}) = \bigoplus_{l=|l_1-l_2|}^{l_1+l_2} \boldsymbol{g}_{l l_1 l_2} \tag{3.20}$$

for some accordingly defined $\boldsymbol{g}_{l l_1 l_2}$, which transforms under rotation R as

$$\boldsymbol{g}_{l l_1 l_2} \xrightarrow{R} \boldsymbol{D}^l \boldsymbol{g}_{l l_1 l_2}. \tag{3.21}$$

The bispectrum of $\rho(\hat{\boldsymbol{r}})$ is defined as

$$b_{l l_1 l_2} = \boldsymbol{c}_l^\dagger \boldsymbol{g}_{l l_1 l_2}, \tag{3.22}$$

which is invariant to rotation (combining equations (3.15) and (3.21)):

$$b_{l l_1 l_2} \xrightarrow{R} (\boldsymbol{D}^l \boldsymbol{c}_l)^\dagger \boldsymbol{D}^l \boldsymbol{g}_{l l_1 l_2} = \boldsymbol{c}_l^\dagger \boldsymbol{g}_{l l_1 l_2}. \tag{3.23}$$

In indicial notation, the bispectrum defined in equation (3.22) can be written as

$$b_{ll_1l_2} = \sum_{m=-l}^{l} \sum_{m1=-l_1}^{l_1} \sum_{m2=-l_2}^{l_2} c^*_{lm} C^{ll_1l_2}_{mm_1m_2} c_{l_1m_1} c_{l_2m_2}, \tag{3.24}$$

where $(\cdot)^*$ denotes the complex conjugate of a scalar.

So far, we have focused only on the rotational symmetry of the representation, but neglected the radical distances of neighboring atoms to the center atom by projecting the atomic neighborhood density function onto the unit sphere. To introduce the radical dependence, one way is to complement the spherical harmonical function with a radical basis; however, as pointed out in [183], this can easily lead to a poor representation if the radial basis functions do not sufficiently overlap. By expanding equation (3.8) using a four-dimensional (4D) hyperspherical harmonics basis, the atomic neighborhood can still be represented in a 3D space, but without the need to explicitly introduce a radial basis set [183]. The derivation of bispectrum in the 4D space is analogous to that in the 3D space, and the final expression for the bispectrum is

$$B_{jj_1j_2} = \sum_{m'_1,m_1=-j_1}^{j_1} c^{j_1}_{m'_1,m_1} \sum_{m'_2,m_2=-j_2}^{j_2} c^{j_2}_{m'_2,m_2} \sum_{m',m=-j}^{j} C^{jj_1j_2}_{mm_1m_2} C^{jj_1j_2}_{m'm'_1m'_2} (c^{j}_{m'm})^{\dagger}. \tag{3.25}$$

The bispectrum of the atomic neighborhood density function is permutationally invariant as can be easily seen from equation (3.8) that changing the order of the atoms only affects the order of summation. It is rotationally invariant as shown above, and a proof of the translational symmetry can be found in Ref. [190]. It is continuous and differentiable, and the length of the descriptor vector depends on the number of expansion coefficients one hopes to use, independent of the number of atoms in the neighborhood. The bispectrum is not complete; however, it is still a remarkably rich invariant representation of atomic environments [190].

3.1.4 Many-body tensor

The Coulomb matrix discussed in section 3.1.1 is diagonalized and the eigenvalues can be used as a representation of the atomic environments. The closely related bag of bonds (BoB) representation [191] also uses the Coulomb matrix, but it arranges the

matrix components in a different manner to obtain the descriptor vector. The matrix components are assigned into a total number of N_s^2 bags according to the two species an component is associated with, where N_s is the total number of species in the system. The bags with fewer bonds are zero padded such that all the bags have the same number of bonds as the largest one (N_b). We can view the bag of bonds as a $N_s^2 \times N_b$ matrix. To proceed, the entries in each bag are sorted according to their magnitude, and then the bags are concatenated in a predefined order to form a vector representation of the atomic environments.

The many-body tensor representation (MBTR) [192] retains the same idea to stratify the bonds according to the species, but avoids any type of sorting by arranging the bond distances on a real-space axis x:

$$\hat{f}_2(x, Z_1, Z_2) = \sum_{i,j=1}^{N_\mathrm{a}} \delta(x - r_{ij}^{-1}) \delta_{Z_1,Z_i} \delta_{Z_2,Z_j}, \tag{3.26}$$

where N_a is the number of atoms, $\delta(\cdot)$ denotes the Dirac delta, $\delta_{\cdot,\cdot}$ denotes the Kronecker delta, and Z_i is the species of atom i. The above measure has a mixed continuous-discrete domain. To make it smooth, the Dirac delta $\delta(\cdot)$ can be replaced by another distribution (e.g. the Gaussian distribution) to achieve a broadening or smearing effect [192], and the Dronecker delta $\delta_{\cdot,\cdot}$ can be replaced by a species similarity matrix $C \in \mathbb{R}^{N_s \times N_s}$. We can also add a weighting function to scale the contributions of certain groups of atoms. Putting all these together, equation (3.26) becomes

$$f_2(x, Z_1, Z_2) = \sum_{i,j=1}^{N_\mathrm{a}} w_2(i,j) p(x, g_2(i,j)) C_{Z_1,Z_i} C_{Z_2,Z_j}, \tag{3.27}$$

where $g(i,j)$ describes a relation between atoms i and j. The above two-body representation can be readily generalized to a many-body representation:

$$f_k(x, \boldsymbol{Z}) = \sum_{i_1,\ldots,i_k=1}^{N_\mathrm{a}} w_k(i_1,\ldots,i_k) p(x, g_k(i_1,\ldots,i_k)) \prod_{j=1}^{k} C_{Z_j,Z_{i_j}}, \tag{3.28}$$

where $\boldsymbol{Z} \in \mathbb{N}^k$ are atom species, w_k assigns a weight for a group of k atoms, and g_k describes a relation among the k atoms in the group. Canonical choices of g_k for

$k = 1, 2, 3, 4$ are atoms count, bond length, bond angle, and dihedral angle, respectively.

Equation (3.28) is continuous along the x axis, not suitable for a vectorized representation of the atomic environments, but it can be readily discretized, resulting in a rank $k+1$ tensor of dimensions $N_s \times \cdots \times N_s \times N_x$ with $N_x = (x_{\max} - x_{\min})/\Delta x$. One can then linearize the element rank of the $k+1$ dimensional tensor to make it a vector representation of the atomic environments.

So far, we have only talked about how to use MBTR to represent molecules with a finite number of atoms. For periodic crystalline systems, there are infinitely many number of atoms (i.e. $N_{\mathrm{a}} = \infty$), making the sum in equation (3.28) diverges. As pointed out in [192], this can be overcome by requiring the indices $i_1, \ldots, i_k$ only run over the atoms in a primitive unit cell. Yet another method is to let the indices only run over the neighboring atoms within a cutoff sphere centered at the target atom.

The length of the MBTR descriptor is independent of the number of neighboring atoms by construction. The MBTR representation is invariant with respect to translation, rotation, and inversion of the space, as well as permutation of atoms with the same species. It is also continuous and differentiable if appropriate g_k is selected, for example, the above mentioned canonical choices. It is complete if a sufficiently large k is used (apparently $k = N_{\mathrm{a}}$ would work), although, in practice, we usually tend to choose a k no larger than 4, making the representation incomplete.

3.2 Regression methods for machine learning potentials

There are many machine learning regression methods suitable for constructing interatomic potentials (IPs). In this section, we review some of the most widely used ones, including the parametric linear regression and neural network (NN) models and the nonparametric kernel ridge regression (KRR) and Gaussian process (GP) models. IPs built using these methods have been shown successful in reproducing first-principles energies and forces and are widely used to study a variety of materials problems. Before delving into these regression methods, let me first introduce the notation that will be used throughout this section. We use a column vector $\boldsymbol{x}$ of length M to denote

an observation of the input data and a scalar y to denote the corresponding output.[5] Collectively, a $N \times M$ matrix $\boldsymbol{X} = [\boldsymbol{x}_1^{\mathrm{T}}; , \boldsymbol{x}_2^{\mathrm{T}}; \cdots ; \boldsymbol{x}_N^{\mathrm{T}}]$ represents a set of N inputs, where each row represents a specific observation, and correspondingly a column vector $\boldsymbol{y} = [y_1; y_2; \cdots ; y_N]$ of length N represents the set of all outputs. We use $\boldsymbol{x}^*$ to denote a new input not in the training set, for which we want to make prediction using a model, and use y^* to denote the prediction.

3.2.1 Linear regression

As the simplest regression method, *linear regression* models the output as a linear combination of the input:

$$y(\boldsymbol{x}; \boldsymbol{\theta}) = \theta_0 + \theta_1 x_1 + \cdots + \theta_M x_M = \sum_{i=0}^{M} \theta_i x_i = \boldsymbol{\theta}^{\mathrm{T}} \boldsymbol{x}, \tag{3.29}$$

where we define an additional dummy input $x_0 = 1$ for the convenience to write the equation in a vector form. This model is not only a linear function of the parameters $\boldsymbol{\theta}$, but also a linear function of the input $\boldsymbol{x}$. The linearity in input, however, significantly limits the generality of the model. Linear regression can be extended to model nonlinear relationship by considering linear combinations of nonlinear functions of the input, taking the form

$$y(\boldsymbol{x}; \boldsymbol{\theta}) = \sum_{i=0}^{D} \theta_i \phi_i(\boldsymbol{x}) = \boldsymbol{\theta}^{\mathrm{T}} \boldsymbol{\phi}(\boldsymbol{x}), \tag{3.30}$$

where $\phi_0(\boldsymbol{x}), \ldots, \phi_D(\boldsymbol{x})$ are a set of D nonlinear basis functions and we define a dummy basis function $\phi_0(\boldsymbol{x}) = 1$ as in equation (3.29). Although equation (3.30) models nonlinear relationship between the input and output, we still call functions of this form linear regression because it is linear in the parameters $\boldsymbol{\theta}$ [96, 177].

Given a data set $\mathcal{D} = (\boldsymbol{X}, \boldsymbol{y})$, the parameters $\boldsymbol{\theta}$ in equation (3.30) can be solved analytically using the least squares method discussed in section 2.1.2. For scalar output,

[5] For simplicity, we use a scalar output. The methods discussed in this section can be easily generalized to vector output.

equation (2.10) can be written as

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\frac{1}{\sigma^2}\sum_{i=1}^{N}\left[y_i - \boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{\phi}(\boldsymbol{x}_i)\right]^2, \tag{3.31}$$

where σ^2 is the variance. The gradient of $\mathcal{L}(\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$ is

$$\nabla_{\boldsymbol{\theta}}\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{\sigma^2}\sum_{i=1}^{N}\left[y_i - \boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{\phi}(\boldsymbol{x}_i)\right]\boldsymbol{\phi}(\boldsymbol{x}_i)^{\mathrm{T}}. \tag{3.32}$$

Setting it to zero, we have

$$\sum_{i=1}^{N} y_i\boldsymbol{\phi}(\boldsymbol{x}_i)^{\mathrm{T}} - \boldsymbol{\theta}^{\mathrm{T}}\left[\sum_{i=1}^{N}\boldsymbol{\phi}(\boldsymbol{x}_i)\boldsymbol{\phi}(\boldsymbol{x}_i)^{\mathrm{T}}\right] = \mathbf{0}. \tag{3.33}$$

Then the solution $\hat{\boldsymbol{\theta}}$ can be obtained as

$$\hat{\boldsymbol{\theta}} = (\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{y}, \tag{3.34}$$

known as the *normal equations* for least-squares problem, where $\boldsymbol{\Phi}$ is a $N \times (D+1)$ matrix, called the *design matrix*:

$$\boldsymbol{\Phi} = \begin{bmatrix} \phi_0(\boldsymbol{x}_1) & \phi_1(\boldsymbol{x}_1) & \dots & \phi_D(\boldsymbol{x}_1) \\ \phi_0(\boldsymbol{x}_2) & \phi_1(\boldsymbol{x}_2) & \dots & \phi_D(\boldsymbol{x}_2) \\ \vdots & \vdots & \ddots & \vdots \\ \phi_0(\boldsymbol{x}_N) & \phi_1(\boldsymbol{x}_N) & \dots & \phi_D(\boldsymbol{x}_N) \end{bmatrix}. \tag{3.35}$$

The quantity $(\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^{\mathrm{T}}$ is known as the *Moore-Penrose pseudo-inverse* of the matrix $\boldsymbol{\Phi}$, which can be regarded as a generalization of the notion of matrix inverse to non-square matrices [96]. In fact, fo invertible square matrix $\boldsymbol{\Phi}$, we have $(\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi})^{-1}\boldsymbol{\Phi}^{\mathrm{T}} = \boldsymbol{\Phi}^{-1}\boldsymbol{\Phi}^{-\mathrm{T}}\boldsymbol{\Phi}^{\mathrm{T}} = \boldsymbol{\Phi}^{-1}$.

The spectral neighbor analysis potential (SNAP) [193] adopts the linear regression approach, where the bispectrum discussed in section 3.1.3 is used as the basis functions.

3.2.2 Kernel ridge regression

KRR [177] is a combination of the ridge regression (linear least squares with an L_2 regularization) and the kernel trick. It learns a linear function in the kernel space, but for nonlinear kernels, this corresponds to a nonlinear function in the original input space.

For a linear regression model $y(\boldsymbol{x};\boldsymbol{\theta}) = \boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{\phi}(\boldsymbol{x})$, we add an L_2 regularization term to its original loss function (equation (3.31)) to obtain a new loss function[6]

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\sum_{i=1}^{N}\left[y_i - \boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{\phi}(\boldsymbol{x}_i)\right]^2, +\lambda\|\boldsymbol{\theta}\|^2, \tag{3.36}$$

where λ is a regularization constant. Solving this by setting the gradient of $\mathcal{L}(\boldsymbol{\theta})$ to zero as in section 3.2.1, we have

$$\hat{\boldsymbol{\theta}} = (\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\Phi} + \lambda\boldsymbol{I})^{-1}\boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{y}, \tag{3.37}$$

where $\boldsymbol{I}$ is an identity matrix. Using the matrix inversion identity,[7] equation (3.37) can be rewritten as

$$\hat{\boldsymbol{\theta}} = \boldsymbol{\Phi}^{\mathrm{T}}(\boldsymbol{\Phi}\boldsymbol{\Phi}^{\mathrm{T}} + \lambda\boldsymbol{I})^{-1}\boldsymbol{y}. \tag{3.38}$$

Defining a new variable $\boldsymbol{\alpha} \coloneqq (\boldsymbol{\Phi}\boldsymbol{\Phi}^{\mathrm{T}} + \lambda\boldsymbol{I})^{-1}\boldsymbol{y}$, we have

$$\hat{\boldsymbol{\theta}} = \boldsymbol{\Phi}^{\mathrm{T}}\boldsymbol{\alpha} = \sum_{i=1}^{N}\alpha_i\boldsymbol{\phi}(\boldsymbol{x}_i). \tag{3.39}$$

The fact that $\boldsymbol{\phi}(\boldsymbol{x}_i)$ corresponds to the ith observation (i.e. the ith row of the design matrix $\boldsymbol{\Phi}$) suggests that the optimal solution is a linear combination of the N observations in the training set.

At test time, the prediction for a new data point $\boldsymbol{x}^*$ can be computed as

$$y(\boldsymbol{x}^*;\hat{\boldsymbol{\theta}}) = \hat{\boldsymbol{\theta}}^{\mathrm{T}}\boldsymbol{\phi}(\boldsymbol{x}^*) = \sum_{i=1}^{N}\alpha_i\boldsymbol{\phi}(\boldsymbol{x}_i)^{\mathrm{T}}\boldsymbol{\phi}(\boldsymbol{x}^*). \tag{3.40}$$

[6] The variance σ^2 in equation (3.31) is absorbed into the regularization constant λ.

[7] For any matrix $\boldsymbol{P}$ and $\boldsymbol{Q}$, we have $\boldsymbol{P} + \boldsymbol{PQP} = \boldsymbol{P}(\boldsymbol{I} + \boldsymbol{QP}) = (\boldsymbol{I} + \boldsymbol{PQ})\boldsymbol{P}$. So if both $\boldsymbol{I} + \boldsymbol{PQ}$ and $\boldsymbol{I} + \boldsymbol{QP}$ are invertible, then $(\boldsymbol{I} + \boldsymbol{PQ})^{-1}\boldsymbol{P} = \boldsymbol{P}(\boldsymbol{I} + \boldsymbol{QP})^{-1}$.

Equation (3.40) is in the form of an inner product, and thus we can apply the kernel trick:

$$y(\boldsymbol{x}^*; \hat{\boldsymbol{\theta}}) = \sum_{i=1}^{N} \alpha_i \kappa(\boldsymbol{\phi}(\boldsymbol{x}_i), \boldsymbol{\phi}(\boldsymbol{x}^*)), \tag{3.41}$$

where $\kappa(\boldsymbol{\phi}(\boldsymbol{x}_i), \boldsymbol{\phi}(\boldsymbol{x}^*))$ is a *kernel function* that measures the similarity between $\boldsymbol{\phi}(\boldsymbol{x}_i)$ and $\boldsymbol{\phi}(\boldsymbol{x}^*)$.

Rupp *et al.* [180] have employed KRR to model molecular atomization energies. They use a Gaussian kernel to measure the similarity between molecules,

$$\kappa = \exp\left[-\frac{1}{2\sigma^2} d(\boldsymbol{\phi}(\boldsymbol{x}), \boldsymbol{\phi}(\boldsymbol{x}'))\right], \tag{3.42}$$

where the distance between molecules is defined as the L_2 norm of the eigenvalues of the Coulomb matrix discussed in section 3.1.1:

$$d(\boldsymbol{\phi}(\boldsymbol{x}), \boldsymbol{\phi}(\boldsymbol{x}')) = \sqrt{\sum_i |\epsilon_i - \epsilon_i'|^2}, \tag{3.43}$$

in which ϵ_i and ϵ_i' are the ith eigenvalues of the Coulomb matrices of the two molecules, respectively.

3.2.3 Gaussian process

KRR is a deterministic model that cannot yield well-calibrated probabilistic outputs. The Bayesian framework provides a probabilistic approach to calibrate models. In the Bayesian approach, we assume there is some function to map an input to an output, $y = f(\boldsymbol{x}; \boldsymbol{\theta})$, and a prior distribution over the parameters $p(\boldsymbol{\theta})$. Given a training set, and a likelihood of the data, $p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta})$, we can then obtain the posterior distribution over $\boldsymbol{\theta}$ as $p(\boldsymbol{\theta}|\boldsymbol{X}, \boldsymbol{y}) \propto p(\boldsymbol{\theta})p(\boldsymbol{y}|\boldsymbol{X}, \boldsymbol{\theta})$. With the posterior distribution, we can evaluate the predictive distribution for a new data point

$$p(y^*|\boldsymbol{x}^*, \boldsymbol{X}, \boldsymbol{y}) = \int p(y^*|\boldsymbol{x}^*, \boldsymbol{\theta}) p(\boldsymbol{\theta}|\boldsymbol{X}, \boldsymbol{y}) \, \mathrm{d}\boldsymbol{\theta}. \tag{3.44}$$

The above approach focuses on parametric representation of the function, $y = f(\boldsymbol{x}; \boldsymbol{\theta})$; however, instead of inferring for the parameters $\boldsymbol{\theta}$, we can define a prior distribution

over functions and perform Bayesian inference over the functions directly.

GP is defined as a probability distribution over functions $f(\boldsymbol{x})$ such that the set of values of $f(\boldsymbol{x}_1), \ldots, f(\boldsymbol{x}_N)$ evaluated at an arbitrary set of points $\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N$ is a joint Gaussian distribution [96,177]. Similar to a Gaussian distribution for random variables, a GP can be denoted as

$$f(\boldsymbol{x}) \sim \mathcal{GP}(m(\boldsymbol{x}), \kappa(\boldsymbol{x}, \boldsymbol{x}')), \tag{3.45}$$

where

$$m(\boldsymbol{x}) = \mathbb{E}[f(\boldsymbol{x})] \tag{3.46}$$

is the mean function, and

$$\kappa(\boldsymbol{x}, \boldsymbol{x}') = \mathbb{E}[(f(\boldsymbol{x}) - m(\boldsymbol{x}))(f(\boldsymbol{x}' - m(\boldsymbol{x}'))] \tag{3.47}$$

is the kernel or covariance function, which needs to be positive definite. For a finite data set $\boldsymbol{X}$, this process defines a joint Gaussian distribution

$$p(\boldsymbol{f}) = \mathcal{N}(\boldsymbol{\mu}, \boldsymbol{K}), \tag{3.48}$$

where $\boldsymbol{\mu} = [m(\boldsymbol{x}_1), \ldots, m(\boldsymbol{x}_N)]$ is the mean vector, and $K_{ij} = \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j)$ is the *Gram matrix*. In most applications, the mean function $\boldsymbol{\mu}$ is simply taken as the constant function $\mathbf{0}$, as no prior knowledge of the value of the latent function underlying the data is available. As a result, we have

$$p(\boldsymbol{f}) = \mathcal{N}(\mathbf{0}, \boldsymbol{K}). \tag{3.49}$$

To apply GP to regression problems with noise in the observations, we assume the noise ϵ is additive, i.e. $y = f(\boldsymbol{x}) + \epsilon$, and it is independent and identically distributed (i.i.d.), following a Gaussian distribution such that

$$p(y|f) = \mathcal{N}(f, \sigma^2), \tag{3.50}$$

where σ^2 is the variance of the noise. The joint distribution of the outputs $\boldsymbol{y} =$

$[y_1, \ldots, y_N]^{\mathrm{T}}$ conditioned on $\boldsymbol{f} = [f_1, \ldots, f_N]^{\mathrm{T}}$ is a Gaussian of the form

$$p(\boldsymbol{y}|\boldsymbol{f}) = \mathcal{N}(\boldsymbol{f}, \sigma^2 \boldsymbol{I}), \tag{3.51}$$

where $\boldsymbol{I}$ is an identity matrix. With the GP prior in equation (3.49) and the conditional distribution in equation (3.51), the marginal distribution conditioned on the data set $\mathcal{D} = (\boldsymbol{X}, \boldsymbol{y})$ can be obtained by integrating over $\boldsymbol{f}$

$$p(\boldsymbol{y}) = \int p(\boldsymbol{y}|\boldsymbol{f}) p(\boldsymbol{f}) \, \mathrm{d}\boldsymbol{f} = \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Sigma}), \tag{3.52}$$

where the covariance matrix has components

$$\Sigma_{ij} = \kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) + \sigma^2 \delta_{ij}. \tag{3.53}$$

So far, we have obtained the marginal distribution of a GP upon observations of the training set. To make prediction for a new data point $(\boldsymbol{x}^*, y^*)$, we need to find the joint distribution of the training set and the new data point, which, according to equation (3.52), is given by

$$\begin{pmatrix} \boldsymbol{y} \\ y^* \end{pmatrix} = \mathcal{N}\left(\boldsymbol{0}, \begin{pmatrix} \boldsymbol{\Sigma} & \boldsymbol{k} \\ \boldsymbol{k}^{\mathrm{T}} & c \end{pmatrix}\right), \tag{3.54}$$

where $\boldsymbol{\Sigma}$ is the $N \times N$ covariance matrix given in equation (3.53), the vector $\boldsymbol{k}$ has elements $k_i = \kappa(\boldsymbol{x}_i, \boldsymbol{x}^*)$ for $i = 1, \ldots, N$, and $c = \kappa(\boldsymbol{x}^*, \boldsymbol{x}^*) + \sigma^2$. Therefore, the predictive distribution is [96]

$$p(y^*|\boldsymbol{y}) = \mathcal{N}(m^*, (\sigma^2)^*), \tag{3.55}$$

where the mean is $m^* = \boldsymbol{k}^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} \boldsymbol{y}$ and the covariance is $(\sigma^2)^* = c - \boldsymbol{k}^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} \boldsymbol{k}$. The predictive mean can also be written as

$$m^* = \boldsymbol{k}^{\mathrm{T}} \boldsymbol{\Sigma}^{-1} \boldsymbol{y} = \sum_{i=1}^{N} \alpha_i \kappa(\boldsymbol{x}_i, \boldsymbol{x}^*), \tag{3.56}$$

where α_i is the ith component of $\boldsymbol{\Sigma}^{-1} \boldsymbol{y}$.

We note that the predictive mean of GP regression (equation (3.56)) has the same form as the prediction made by KRR (equation (3.41)). However, GP regression has the advantage that it is a full probabilistic model, from which confidence interval of a prediction can be obtained by manipulating the predictive covariance.

Csányi *et al.* have employed the smooth overlap of atomic positions (SOAP) kernel [183] to create GP potentials for a wide range of materials such as carbon, silicon, and germanium [179, 194, 195].

3.2.4 Neural network

The linear regression model discussed in section 3.2.1 takes the form

$$y(\boldsymbol{x};\boldsymbol{\theta}) = f(\boldsymbol{\theta}^{\mathrm{T}}\boldsymbol{\phi}(\boldsymbol{x})), \tag{3.57}$$

where $f(\cdot)$ is merely an identity function in this case. In order to apply such models to large-scale problems, it is necessary to adapt the basis functions $\boldsymbol{\phi}(\cdot)$ to the data [96]. An approach is to make the basis functions parametric and allow these parameters to be adjusted, along with the original parameters $\boldsymbol{\theta}$, during training. A feed-forward NN is a series of models in the form of equation (3.57) composed on top of each other, with the outmost function $f(\cdot)$ remaining an identity function, but the others replaced by some nonlinear activation functions.[8] In such, we achieve the goal to transform each basis function to a nonlinear function of linear combination of the inputs.

An NN can be represented graphically in the form of a network diagram as shown in figure 3.1. This example NN consists of an input layer, two hidden layers, and an output layer. Each node in the hidden layers is connected to all the nodes in the previous layer and the next layer, and the node value is[9]

$$y_m^n = \sigma\left(\sum_{n'} y_{m-1}^{n'} w_m^{n',n} + b_m^n\right), \quad m = 1, 2, 3, \tag{3.58}$$

[8] If the inner activation functions are linear, then the network can be replaced by an equivalent model in the form of equation (3.57). This follows from the fact that the composition of successive linear transformations is itself a linear transformation.

[9] Following the NN literature, a bias parameter b is separated from the set of weight parameters. The bias parameter b is associated with an input variable whose value is clamped at 1.

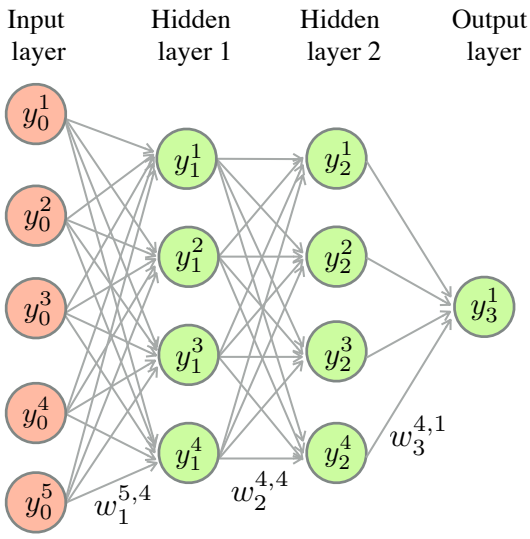


Figure 3.1: Schematic representation of an NN comprised of an input layer, two hidden layers and an output layer. Each arrow connecting two nodes between adjacent NN layers represents a weight. Biases and activation function are not shown in this plot. See text for explanation of the variables.

where y_m^n is the value of node n in layer m, $w_m^{n',n}$ is the weight connecting node n' in layer $m-1$ and node n in layer m, b_m^n is the bias applied to node n of layer m, and σ is an activation function (e.g. hyperbolic tangent) that introduces nonlinearity into the NN. In a more compact way, equation (3.58) can be written as $\boldsymbol{y}_m = \sigma(\boldsymbol{y}_{m-1}\boldsymbol{W}_m + \boldsymbol{b}_m)$,[10] where $\boldsymbol{y}_m$ is a row vector of the node values in layer m, $\boldsymbol{W}_m$ is a weight matrix, and $\boldsymbol{b}_m$ is a row vector of the biases. For example, $\boldsymbol{y}_1$ and $\boldsymbol{b}_1$ are row vectors each with 4 elements and $\boldsymbol{W}_1$ is a 5×4 matrix for the NN shown in figure 3.1. Consequently, the output can be expressed as[11]

$$\boldsymbol{y}_3 = \sigma[\sigma[\boldsymbol{y}_0\boldsymbol{W}_1 + \boldsymbol{b}_1]\boldsymbol{W}_2 + \boldsymbol{b}_2]\boldsymbol{W}_3 + \boldsymbol{b}_3. \tag{3.59}$$

In essence, the NN model is nothing but a nonlinear function $y = f(\boldsymbol{x};\boldsymbol{\theta})$ that maps a set of input to a set of output controlled by adjustable parameters $\boldsymbol{\theta} = \{\boldsymbol{W}, \boldsymbol{b}\}$. Therefore, training an NN is not different from training any other nonlinear parametric

[10] The activation function is applied element-wisely.
[11] Usually the activation function is not applied to the output layer.

model. Given a training set $\mathcal{D} = (\boldsymbol{X}, \boldsymbol{y})$, we minimize the loss function

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2} \sum_{i=1}^{N} \| f(\boldsymbol{x}_i; \boldsymbol{\theta}) - y_i \|^2, \tag{3.60}$$

with respect to the parameters $\boldsymbol{\theta}$. A large number of, if not all, minimization algorithms require the gradient of the loss function with respect to the parameters. Thanks to the structure of the NN model, there is an efficient technique to evaluate the gradient of the loss function in equation (3.60). This can be achieved by using a local message passing scheme in which information is sent alternately forwards and backwards through the NN, known as the *error backpropagation* [196]. The error backpropagation technique only requires an overall computational cost of $\mathcal{O}(\boldsymbol{W})$, proportional to the number of weight parameters in the NN [96].

In principle, we can use any minimization algorithm to optimize the parameters, such as the Levenberg–Marquardt (LM) method discussed in section 2.1.2 and the Broyden–Fletcher–Goldfarb–Shanno minimization algorithm (BFGS) method that we shall use in section 3.3. The loss function in equation (3.60) decomposes as a sum over the training set, so does the gradient:

$$\nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2} \sum_{i=1}^{N} \nabla_{\boldsymbol{\theta}} \| f(\boldsymbol{x}_i; \boldsymbol{\theta}) - y_i \|^2. \tag{3.61}$$

Therefore, the computational cost of one minimization step is $\mathcal{O}(N)$, proportional to the number of data points N in the training set. A recurring problem in machine learning is that large training sets are necessary for good generalization. So, batch optimization methods that require the whole training set (e.g. LM and BFGS) are computationally expensive for machine learning problems, although they typically have good convergence behaviors and lead to a small final loss.

In practice, nearly all machine learning is powered by the stochastic gradient descent (SGD) algorithm [197], an extension of the standard gradient descent algorithm. The insight of SGD is to treat the gradient as an expectation and estimate this expectation using a (small) subset of the training data. Specifically, at each minimization step, instead of using the whole training set to compute the gradient, we sample a *minibatch*

of examples $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_{N'}\}$ from the training set. The minibatch size N' is typically chosen to be a relatively small number, ranging from 1 to a few hundred, and it is usually held fixed as the training set size N grows. In such, we can estimate the gradient at each minimization step with $\mathcal{O}(1)$ time:

$$\nabla_{\boldsymbol{\theta}} \mathcal{L}(\boldsymbol{\theta}) \approx \frac{N}{2N'} \sum_{i=1}^{N'} \nabla_{\boldsymbol{\theta}} \| f(\boldsymbol{x}_i; \boldsymbol{\theta}) - y_i \|^2. \tag{3.62}$$

Another crucial feature of SGD is that it allows the loss to increase during minimization, which is inevitable in training dropout NNs that will be discussed in section 4.3, because the NN structure changes from step to step when dropout is applied.

3.3 A neural network potential for multilayer graphene

As a prototype of stacked two-dimensional (2D) materials, multilayer graphene exhibits strong sp^2 covalent bonds within a layer and weak dispersion and orbital repulsion interactions between layers. The cohesive energy of monolayer graphene, characterizing intralayer bonding, is 8.06 eV/atom, whereas the interlayer binding energy of bilayer graphene is only 0.02263 eV/atom. Although weak, it is the interlayer interactions that define the function of many nanodevices such as nanobearings, nanomotors, and nanoresonators [150], and also drive incommensurate to commensurate structural transitions [175, 198], which lead to novel transport properties [45, 199].

As discussed in section 2.3, there have been several efforts to develop an interatomic potential (IP) for carbon systems. Early efforts include the bond-order Tersoff [113, 147] and reactive empirical bond order (REBO) [83, 114] potentials, which modulate the strength of bonds based on their atomic environments. These potentials provide a reasonable description for strong covalent bonds, but do not account for dispersion interactions and thus are inherently short-ranged in nature. To address this limitation, the adaptive intermolecular reactive empirical bond order (AIREBO) [148] potential adds a 6–12 Lennard–Jones (LJ) [77] term to model dispersion, and the long-range carbon bond order potential (LCBOP) [84] and AIREBO–M [200] potentials add Morse [153] terms for this purpose. The more complex reactivate force field (ReaxFF) [201] potential constructs the bond order differently than the above potentials and includes

explicit terms to account for van der Waals (vdW), Coulombic, and under- and over-coordination energies.

These IPs have been shown to work well for a variety of applications, but in many cases their quantitative predictions are inaccurate when compared with first-principles and experimental results. For example, the phonon dispersion curves of monolayer graphene at 0 K computed using these IPs deviate largely from density functional theory (DFT) results, especially for the optical modes (discussed later in section 3.3.2). As for interlayer interactions, the Tersoff and REBO potentials cannot be used because they do not account for long-range dispersion interactions. The AIREBO, AIREBO–M, LCBOP, and ReaxFF potentials do predict overall binding characteristics between graphene layers, such as the equilibrium layer spacing and the c-axis elastic modulus, but are unable to accurately distinguish energy variations for different relative alignments of layers [68]. The reason is that in addition to dispersion, the interlayer interactions include short-range Pauli repulsion between overlapping π orbitals of adjacent layers. The repulsive interaction is not correctly modeled in these IPs. The registry-dependent Kolmogorov–Cresp (KC) potential [150] and the dihedral-angle-corrected registry-dependent interlayer potential (DRIP) [68] discussed on section 2.3 address this by employing a term that depends on the transverse distance between atom pairs to capture the repulsion due to orbital overlapping. However, a major limitation of the KC potential and DRIP is that they are not reactive, i.e. they require an *a priori* fixed assignment of atoms into layers. This prevents the study of many problems of interest, such as vacancy migration between layers [202].

For carbon systems, Csányi *et al.* have developed two Gaussian approximation potentials (GAPs)[12]: one for liquid and amorphous carbon [203] and the other for monolayer graphene [194]. Khaliullin *et al.* [11, 204] have developed neural network (NN) potentials to model phase transition from graphite to diamond. Generally speaking, the *transferability* (i.e. the ability of an IP to make accurate predictions outside its training set) of machine learning potentials is low. Therefore, given their training sets, the GAP for liquid and amorphous carbon and the NN potentials for phase transition are not suitable for multilayer graphene structures. The GAP for graphene is an accurate model that correctly reproduces many properties of monolayer graphene obtained from

[12] GAP uses Gaussian process as the repression method.

DFT [194]; however, similar to the Tersoff and REBO potentials, it lacks a description of the interlayer interactions and therefore cannot be used for multilayer graphene structures.

In this section, we present a new hybrid NN and physics-based potential for multilayer graphene systems that is reactive and provides an accurate description of both the intralayer and interlayer interactions. The potential is referred to as "hNN-Gr$_x$" (where the subscript x indicates that it can be used for multiple graphene layers). The long-range dispersion attraction is modeled using a theoretically-motivated r^{-6} term (as in the LJ potential), and the short-range interactions are described using a general-purpose NN. The latter include both the covalent bonds within a layer and the repulsion due to overlapping orbitals of adjacent layers. The parameters in the new hNN-Gr$_x$ potential are trained against a large dataset of monolayer graphene, bilayer graphene, and graphite configurations obtained from DFT calculations with an accurate dispersion correction.

3.3.1 Definition of model

The total potential energy of a configuration consisting of N atoms is decomposed into the contributions of individual atoms

$$E = \sum_{i=1}^{N_a} E_i, \tag{3.63}$$

where E_i is the atomic energy of atom i, composed of a long-range interaction part and a short-range interaction part, i.e. $E_i = E_i^{\text{long}} + E_i^{\text{short}}$. The long-range dispersion attraction is modeled by a theoretically-motivated r^{-6} term as in the LJ potential,

$$E_i^{\text{long}} = -A \sum_{j \neq i}^{N_a} r_{ij}^{-6} \cdot S_{\text{up}}(x) \cdot S_{\text{down}}(x), \tag{3.64}$$

where A is a fitting parameter, r_{ij} is the distance between atoms i and j, and $S_{\text{up}}(x)$ and $S_{\text{down}}(x)$ are switching functions that turn interactions on and off in certain distance

Table 3.1: Summary of parameters in the hNN-Gr$_x$ potential and hyperparameters that define the neural network structure in the short-range part of the potential.

A	8.3427 eV
$r_{\text{up}}^{\text{min}}$	2 Å
$r_{\text{up}}^{\text{max}}$	4 Å
$r_{\text{down}}^{\text{min}}$	9 Å
$r_{\text{down}}^{\text{max}}$	10 Å
number of hidden layers	3
number of nodes in hidden layers	30
activation function σ	tanh
$r_{\text{cutoff}}^{\text{short}}$	5 Å
descriptors	see section 3.1.2
weights	see appendix B
biases	see appendix B

ranges. The down switching function is defined as

$$S_{\text{down}}(x) = \begin{cases} 1, & x < 0 \\ -6x^5 + 15x^4 - 10x^3 + 1, & 0 \leq x \leq 1 \\ 0, & x > 1 \end{cases}. \tag{3.65}$$

This function monotonically decreases from one to zero over the range $x \in [0, 1]$, and has zero first and second derivatives at both $x = 0$ and $x = 1$. The up switching function is the complementary expression, $S_{\text{up}}(x) = 1 - S_{\text{down}}(x)$. The switches are applied within a desired distance interval $[r^{\text{min}}, r^{\text{max}}]$ using the dimensionless argument,

$$x = x(r_{ij}) = \frac{r_{ij} - r^{\text{min}}}{r^{\text{max}} - r^{\text{min}}}. \tag{3.66}$$

The values of r^{min} and r^{max} for the up and down switching functions are given in table 3.1. With these values, the down switching function $S_{\text{down}}(x)$ causes the potential to smoothly vanish at the cutoff $r_{\text{down}}^{\text{max}}$, and the up switching function $S_{\text{up}}(x)$ turns off the long-range interactions when the pair distance r_{ij} is smaller than $r_{\text{up}}^{\text{min}}$.

The short-range interactions (including both the covalent bonds within a layer and the repulsion between overlapping orbitals of adjacent layers) are represented by an

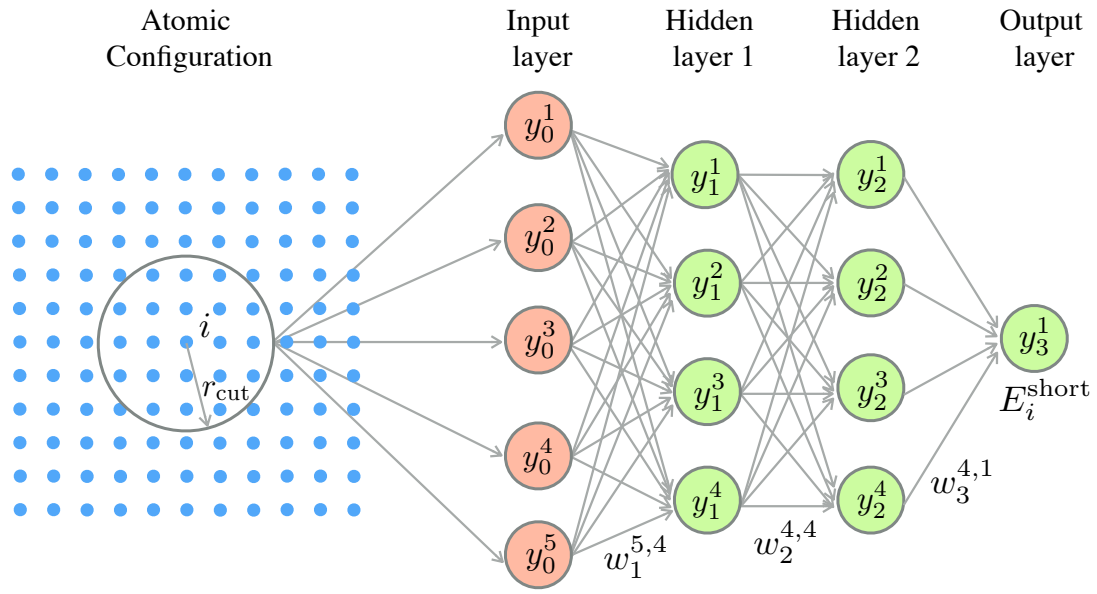


Figure 3.2: Schematic representation of an NN potential for the short-range energy E_i^{short} of atom i. This NN consists of an input layer, two hidden layers, and an output layer. The configuration of neighbors of atom i within a cutoff r_{cut} is transformed to a descriptor vector y_0^j $(j = 1, 2, \ldots, 5)$, which serves as the input to the NN. The arrows connecting nodes in adjacent layers represent weights. Biases and activation functions are not shown in this figure. See text for explanation of the variables.

NN as shown schematically in figure 3.2. For this example NN, the short-range atomic energy E_i^{short} can be expressed as (see section 3.2.4 for more information)

$$E_i^{\text{short}} = \sigma[\sigma[\boldsymbol{y}_0 \boldsymbol{W}_1 + \boldsymbol{b}_1]\boldsymbol{W}_2 + \boldsymbol{b}_2]\boldsymbol{W}_3 + \boldsymbol{b}_3. \tag{3.67}$$

IPs must be invariant with respect to translation, rotation, and inversion of space, and permutation of chemically equivalent atoms as discussed in section 3.1. To ensure that the NN satisfies these requirements, the environment of atom i, which is the input to the NN, must be transformed to a new representation called a *descriptor* that automatically satisfies these invariances. Thus the input layer $\boldsymbol{y}_0$ is a descriptor vector which is a function of the set of positions $\boldsymbol{r}_i^{\text{neigh}}$ of all atoms within the neighborhood

of atom i defined by the cutoff distance r_{cut} (including atom i itself), i.e.[13]

$$y_0^j = g^j(\boldsymbol{r}_i^{\text{neigh}}), \tag{3.68}$$

where j ranges over the components of the descriptor vector. As discussed in section 3.1, there are various types of descriptors to represent the atomic environments. In this section, we use symmetry functions [187] discussed in section 3.1.2 and the hyperparameters are given in appendix B.

A challenging aspect of training an NN, which is also a source of the power and flexibility of the method, is that it is up to developer to select the number of descriptor terms to retain, the number of hidden layers, the number of nodes within each hidden layer (which need not be the same), and the activation function. It is also possible to create different connectivity scenarios between layers. Here we have opted for simplicity and adopted a fully-connected network with the same number of nodes in each hidden layer to reduce the number of hyperparameters that need to be determined in the training process. We chose the commonly used hyperbolic tangent function, $\tanh(x) = (e^x - e^{-x})/(e^x + e^{-x})$, as the nonlinear activation function σ.

The hNN-Gr$_x$ potential parameters were determined from a dataset of energies and forces for pristine and defected monolayer graphene, bilayer graphene, and graphite at various states. This includes configurations with compressed and stretched cells, random perturbations of atoms, and configurations drawn from *ab initio* molecular dynamics (AIMD) trajectories at different temperatures. The dataset consists of a total number of 14,250 configurations that are randomly divided into a training set of 13,500 configurations (95%) and a test set of 750 configurations (5%). The dataset along with a detailed description of the configurations are provided in appendix D. The data set is generated from DFT calculations using Vienna ab initio simulation package (VASP) in the same way as described in section 2.3.1.

The hNN-Gr$_x$ potential is fit in two stages: first the parameters in the long-range part in equation (3.64) are determined, then the parameters in the short-range NN part in equation (3.67).

For the long-range part, the interval bounds in the switching functions ($r_{\text{up}}^{\text{min}}$, $r_{\text{up}}^{\text{max}}$,

[13] The descriptor values are normalized by subtracting from each component y_0^j the mean value for this component across all atomic environments in the training set and dividing by the standard deviation.

$r_{\text{down}}^{\text{min}}$, $r_{\text{down}}^{\text{max}}$) are listed in table 3.1. The $r_{\text{up}}^{\text{min}}$ and $r_{\text{up}}^{\text{max}}$ values are selected based on the graphene equilibrium lattice spacing of about 3.4 Å, $r_{\text{down}}^{\text{max}}$ sets the cutoff of the long-range interactions and is based on prior experience with DRIP [68] (see section 2.3), and $r_{\text{down}}^{\text{min}}$ is set a bit lower to create a smooth transition. After fixing these, a single parameter $\boldsymbol{\theta} = \{A\}$ remains to be determined. It is optimized by minimizing a loss function $\mathcal{L}(\boldsymbol{\theta})$ that quantifies the difference between the predictions of equation (3.64) and DFT results for a subset of the training set. The subset consists of 52 configurations of AB-stacked bilayer graphene at various layer spacings ranging from $r_{\text{up}}^{\text{max}}$ to $r_{\text{down}}^{\text{min}}$. The parameters are optimized by minimizing the loss function in equation (2.14), i.e.

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\sum_{i=1}^{N} w_i^{\text{e}} \left[E(\boldsymbol{r}_i;\boldsymbol{\theta}) - \hat{E}_i\right]^2 + \frac{1}{2}\sum_{i=1}^{N} w_i^{\text{f}} \|\boldsymbol{f}(\boldsymbol{r}_i;\boldsymbol{\theta}) - \hat{\boldsymbol{f}}_i\|^2, \tag{3.69}$$

where $E(\boldsymbol{r}_i;\boldsymbol{\theta})$ and $\boldsymbol{f}(\boldsymbol{r}_i;\boldsymbol{\theta}) = -\left(\partial E/\partial \boldsymbol{r}\right)|_{\boldsymbol{r}_i}$ are the potential energy and forces of configuration i, in which $E(\boldsymbol{r}_i;\boldsymbol{\theta}) = E^{\text{long}} = \sum_{n=1}^{N_{\text{a},i}} E_n^{\text{long}}$, where $N_{\text{a},i}$ is the number of atoms in configuration i. The energy weight w_i^{e} and force weight w_i^{f} of configuration i have units of eV^{-2} and $(\text{eV/Å})^{-2}$, respectively, given energy in units of eV and forces in units of eV/Å. We set w_i^{e} to $1/(N_{\text{a},i})^2$, and w_i^{f} to $1/(10(N_{\text{a},i})^2)$.[14] The target DFT energy and forces for the long-range part $\hat{E}_i$ and $\hat{\boldsymbol{f}}_i$ consider only interlayer interactions, obtained in the same way as described in details in section 2.3.1. The resulting parameter A is given in table 3.1.

With the long-range interactions determined, the next step is to determine the short-range part of the potential. The same loss function in equation (3.69) is used with three differences compared with the long-range fitting: (i) the parameters $\boldsymbol{\theta}$ are the weights $\boldsymbol{W}$ and biases $\boldsymbol{b}$ in the NN; (ii) the entire training set is used; and (iii) the target energies $\hat{E}_i$ and forces $\hat{\boldsymbol{f}}_i$ are the differences between the total DFT values and the predictions from the long-range contribution in equation (3.64). The third item ensures that the potential produces correct total energy and forces when the long-range and short-range parts are used together.

The optimization was carried out using the KIM-based learning-integrated fitting

[14] The weights are inversely proportional to $(N_{\text{a},i})^2$ such that each configuration contributes more or less equally to the loss $L(\boldsymbol{\theta})$. This prevents configurations with more atoms from dominating the optimization.

framework (KLIFF) [69] with a Broyden–Fletcher–Goldfarb–Shanno minimization algorithm (BFGS) minimizer [205]. A grid search was performed to determine the optimal number of hidden layers and nodes by fitting the potential to the training set in each case and finding which provided the minimum loss for the test set.[15] Using this process, it was found that 3 hidden layers with 30 nodes per layer was the optimal choice. The resulting energy root-mean-square error (RMSE) and forces RMSE for the test set are 4.66 meV/atom and 41.41 meV/(Å atom), respectively, and 4.56 meV/atom and 41.13 meV/(Å atom) for the training set. See table 3.1 for details of the NN parameters.

3.3.2 Testing of the hNN-Gr$_x$ potential

An extensive set of calculations were performed to test the ability of the new hNN-Gr$_x$ potential to reproduce structural, energetic, and elastic properties of monolayer graphene, bilayer graphene, and graphite obtained from DFT. A portion of the results is presented in table 3.2 together with results from widely used IPs, *ab initio* ACFDT–RPA, and experiments.

The in-plane lattice parameter of monolayer graphene, a, is obtained by fitting the Birch–Murnaghan equation of state (EOS) [208] (to conform to the approach used in DFT computations). The results presented in table 3.2 show that AIREBO and AIREBO–M underestimate the value of a, Tersoff overestimates it, and the other potentials give values close to the experimental and DFT results. Table 3.2 also shows the values of the equilibrium layer spacing for bilayer graphene in AB stacking d_{AB}, bilayer graphene in AA stacking d_{AA}, and graphite d_{graphite}. These values are also obtained from the Birch–Murnaghan EOS, keeping the in-plane lattice parameter fixed to its equilibrium monolayer value. The hNN-Gr$_x$ potential and DRIP are in good agreement with DFT(PBE+MBD) results to which they were fit. The KC model is in better agreement with more accurate ACFDT–RPA. The remaining IPs all underestimate the AA separation, and have inconsistent results for AB and graphite: AIREBO and LCBOP are accurate for both, and AIREBO–M and ReaxFF underestimate both. Given this it is not surprising that except for hNN-Gr$_x$, KC, and DRIP, all of the above IPs provide inaccurate values for $d_{\mathrm{AA}} - d_{\mathrm{AB}}$. The DFT value is 0.215 Å, and the potentials predict:

[15] The loss of the test set is used to make the determination, rather than the training set, to prevent overfitting.

Table 3.2: Summary of structural, energetic, and elastic properties computed from the hNN-Gr$_x$ potential and other widely used potentials. The properties include in-plane lattice parameter of monolayer graphene, a; equilibrium layer spacing of bilayer graphene in AB stacking, d_{AB}, bilayer graphene in AA stacking, d_{AA}, and graphite, d_{graphite}; optimal interlayer binding energy of bilayer graphene, E_{AB}; cohesive energy of monolayer graphene, E_{coh}; single-vacancy formation energy in monolayer graphene, E_{v}; and elastic moduli of graphite (outside parentheses) and monolayer graphene (in parentheses). Also included are some first-principles and experimental results, as well as the computational expense relative to Tersoff. Notes: (1) Since the Tersoff, REBO, and GAP–Gr (GAP for graphene) potentials lack the ability to model interlayer interactions, their predictions for properties related to interlayer interactions are not included. (2) The KC and DRIP potentials only model interlayer interactions and therefore cannot be used to compute the in-plane lattice parameter. Results from these potentials used an in-plane lattice parameter of $a = 2.46$ Å. For elastic properties, only the modulus related to stretching perpendicular to the layers is computed in this case.

Method	a (Å)	d_{AB} (Å)	d_{AA} (Å)	d_{graphite} (Å)	E_{AB} (eV/atom)	E_{coh} (eV/atom)	E_{v} (eV)	C_{11} (GPa)	C_{12} (GPa)	C_{13} (GPa)	C_{33} (GPa)	C_{44} (GPa)	Time (relative)
hNN-Gr$_x$ (present)	2.467	3.457	3.618	3.402	21.63	8.07	8.08	978.31 (1061.83)	176.54 (208.77)	−66.74	40.35	1.79	279.4
AIREBO [148]	2.419	3.392	3.416	3.358	23.61	7.43	7.94	1153.50 (1162.46)	144.87 (147.64)	0.08	40.40	0.28	4.5
AIREBO–M [200]	2.420	3.299	3.324	3.294	16.18	7.42	7.93	1174.25 (1157.43)	147.66 (146.23)	-0.02	35.72	0.28	4.9
LCBOP [84]	2.459	3.346	3.365	3.346	12.52	7.35	8.13	1049.91 (1054.32)	157.29 (159.03)	0.04	29.80	0.23	1.6
ReaxFF [201]	2.462	3.285	3.294	3.260	34.59	7.52	7.52	1147.67 (1119.84)	831.84 (811.43)	−0.77	34.41	0.15	26.1
Tersoff [147]	2.530					7.39	7.12	(1274.00)	(−240.11)				1
REBO [83]	2.460					7.39	7.82	(1059.25)	(148.33)				1.6
GAP–Gr [194]	2.467					7.96	6.55	(1108.81)	(212.19)				3814.7
KC [150]		3.374	3.602	3.337	21.60						34.45		36.6
DRIP [68]		3.439	3.612	3.415	23.05						32.00		35.5
DFT(PBE+MBD)	2.466	3.426	3.641	3.400	22.63	8.06	7.93	1080.12 (1084.41)	162.25 (161.25)	−4.63	33.18	3.32	$\sim 10^7$
ACFDT–RPA		3.39[a]		3.34[b]							36[b]		
Experiment	2.46[c]			3.34[d]				1060[e] (1018[f])	180[e]	15[e]	36.5[e]	0.27[e]	
	2.46[g]			3.356[h]				1109[h]	139[h]	0[h]	38.7[h]	4.95[h]	

[a] [164].
[b] [165].
[c] [206].
[d] [166].
[e] [171].
[f] [38].
[g] [207].
[h] [172].

0.024 Å (AIREBO), 0.025 Å (AIREBO–M), 0.019 Å (LCBOP), and 0.009 Å (ReaxFF). The reason for the poor accuracy is that these potentials cannot distinguish the AA and AB stacking states. This is discussed further below.

Next, we consider energetics. The interlayer binding energy of a graphene bilayer E_b as a function of layer spacing d is shown in figure 3.3 for AB and AA stacking. The curves are shifted such that $\Delta E = E_\mathrm{b} - E_\mathrm{AB}$ and $\Delta d = d - d_\mathrm{AB}$, where E_AB (listed in table 3.2) is the optimal interlayer binding energy of AB-stacked bilayer graphene at the equilibrium layer spacing d_AB (i.e. E_AB is the depth of the energy well relative to a reference state at infinite separation). We see that the AIREBO, AIREBO–M, LCBOP, and ReaxFF potentials give nearly identical results for energy versus separation in the AB and AA stacking states in contrast to DFT where a clear difference exists. In addition, the AIREBO–M and LCBOP potentials underestimate the depth of the energy wells, whereas the ReaxFF potential overestimates it. (This can be seen by considering the values predicted by these potentials relative to DFT at the largest separation of $\Delta d = 2.5$ Å, which is approaching the reference state). The hNN-Gr$_x$ potential correctly captures the energy difference between the AB and AA stacking states as well as the depth of the energy wells. KC and DRIP can also capture the energy difference (see section 2.3.2). Also notable is that at large separation, the curves for the two stacking states merge since registry effects due to π-orbital overlap become negligible and interlayer interactions are dominated by dispersion attraction. This effect is captured correctly by the hNN-Gr$_x$ potential.

A more complete view of the interlayer energetics is obtained by considering the generalized stacking fault energy (GSFE) surface obtained by sliding one layer relative to the other while keeping the layer spacing fixed. Figure 3.4 shows the results for a layer spacing of $d = 3.4$ Å; the hNN-Gr$_x$ potential is in quantitative agreement with DFT results. The KC and DRIP GSFEs have a similar appearance (see section 2.3.2), whereas the AIREBO, AIREBO–M, LCBOP, and ReaxFF GSFEs are nearly flat (not shown).

Also listed in table 3.2 are the cohesive energy E_coh and relaxed single-vacancy formation energy E_v for monolayer graphene. The latter is computed as $E_\mathrm{v} = E_2 - E_1 - \mu$, where E_1 and E_2 are the relaxed energy of monolayer graphene before and after the single vacancy is created (by removing an atom from the simulation cell),

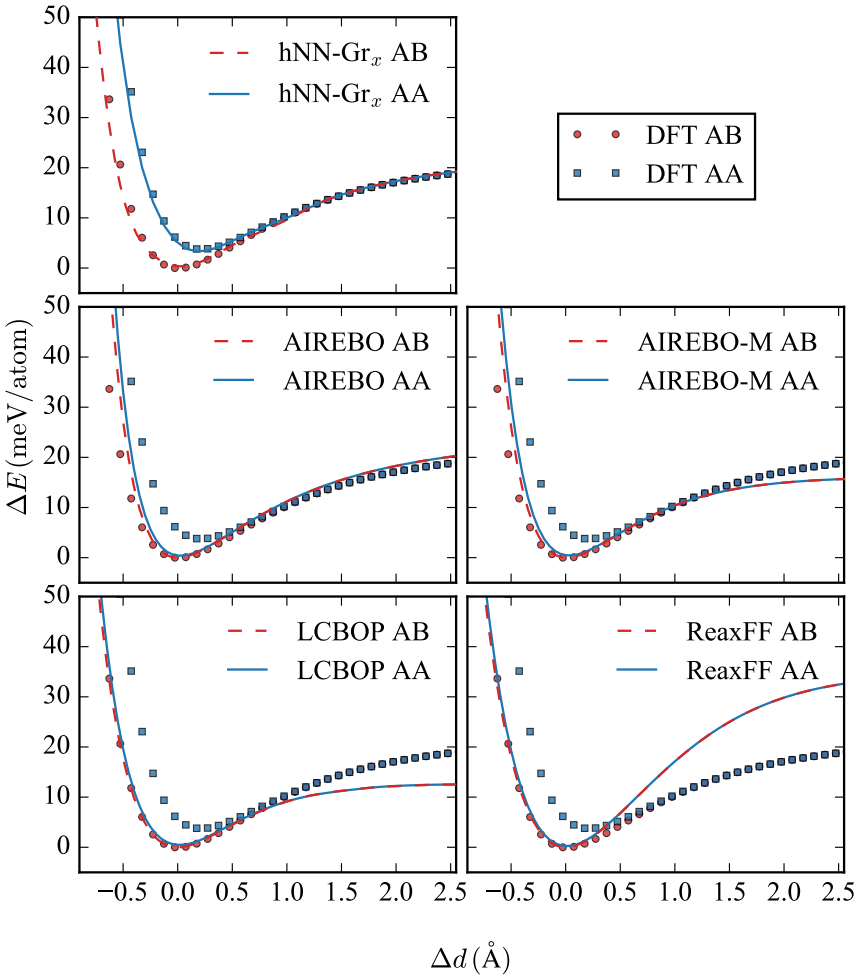


Figure 3.3: Interlayer binding energy of a graphene bilayer versus layer spacing for AB and AA stackings obtained from various potentials compared with DFT results. The curves are shifted such that the minimum energy in AB stacking is located at (0, 0).

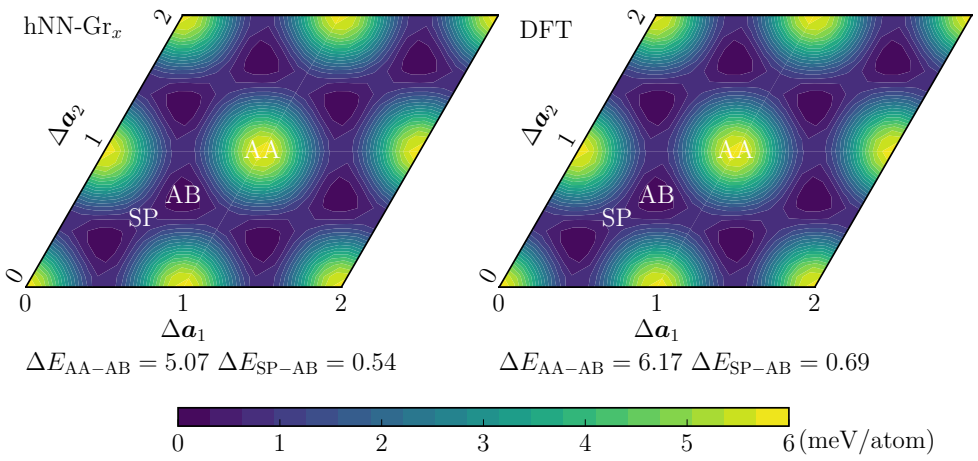


Figure 3.4: The GSFE of bilayer graphene obtained by sliding one layer relative to the other at a fixed layer spacing of $d = 3.4$ Å. The energy is relative to the AB state, which is -21.53 meV/atom for the new hNN-Gr$_x$ potential (on the left) and -22.33 meV/atom for DFT (on the right). $\Delta E_{\mathrm{AA\text{-}AB}}$ denotes the energy difference between the AA and AB states, and similarly $\Delta E_{\mathrm{SP\text{-}AB}}$ denotes the energy difference between saddle point (SP) and AB states. The sliding parameters $\Delta \boldsymbol{a}_1$ and $\Delta \boldsymbol{a}_2$ are in units of lattice parameter $a = 2.466$ Å.

and μ is the chemical potential of carbon, taken to be the cohesive energy E_{coh} here. All IPs perform reasonably well for these two properties except that the single-vacancy formation energy predicted by GAP–Gr is significantly smaller compared with the other IPs and DFT. This is likely because GAP–Gr was only trained against configurations drawn from molecular dynamics (MD) trajectories of ideal graphene.

Finally, we consider elasticity properties. The elastic moduli of hexagonal graphite was computed using finite differences. The five independent components are listed in table 3.2. For each IP, the graphitic structure is constructed using its corresponding in-plane lattice parameter, a, and equilibrium layer spacing d_{graphite}. In addition, the in-plane elastic moduli C_{11} and C_{12} of monolayer graphene were computed (values listed in parentheses). Similar to graphite, the graphene structure is constructed using the corresponding in-plane lattice parameter of each IP, whereas the "thickness" of graphene (required to obtain bulk units) is assumed to be 3.34 Å in all cases. The results show that for graphite the hNN-Gr$_x$ potential is in good agreement with DFT for C_{11} (9.5%) and C_{12} (8.8%), reasonable agreement for C_{33} (21.6%) and C_{44} (46.1%), and incorrect

for C_{13} (1340%) (although we note that the DFT results disagree with experiments in this case). For graphene, the hNN-Gr$_x$ potential is in excellent agreement for C_{11} (2.1%), but overestimates C_{12} (29.5%). For the other potentials, notable disagreements are: (i) ReaxFF predicts significantly larger values of C_{12} of both graphite and graphene; (ii) All of the potentials greatly underestimate C_{44} for graphite; (iii) Tersoff overestimates C_{11} and predicts negative C_{12} for graphene; and (iv) GAP–Gr overestimates C_{12} for graphene.

While the elastic moduli provide insight into the elastic behavior of the IPs, a more complete view is gained from the phonon dispersion curves. A number of thermodynamic properties, such as the thermal expansion coefficient and heat capacity, can be obtained directly from dispersion relations via calculation of the free energy. Figure 3.5 shows the phonon dispersion curves of monolayer graphene calculated using finite differences as implemented in the phonopy package [209]. The predictions of the hNN-Gr$_x$ potential and GAP–Gr are in excellent agreement with DFT. The other potentials provide good agreement for some phonon branches, but not all. REBO quantitatively predicts the shape and dispersion character of most of the phonon branches, but fails for the high-frequency transverse optical (TO) and longitudinal optical (LO) branches. LCBOP, AIREBO, AIREBO–M, and ReaxFF are comparable, qualitatively predicting the overall shapes of most curves, but are in poor quantitative agreement with DFT. Tersoff has the worst performance with poor qualitative agreement for most branches. We note that a drawback common to all of the physics-based potentials is that they fail to capture the dispersive behavior of the high-frequency LO and TO branches, which hNN-Gr$_x$ and GAP-Gr predict with negligible error. The phonon dispersions of bilayer graphene and graphite (not shown here) are identical to monolayer graphene, except that the ZA branch splits into two doubly degenerate branches near the Γ point [210, 211].

For the properties computed above and the IPs tested, the results indicate that overall, machine learning potentials (both hNN-Gr$_x$ and GAP–Gr) have higher accuracy than the physics-based potentials. However, the accuracy comes at the price of increased computational cost. Table 3.2 shows the time (relative to Tersoff) that it takes each potential to complete an MD trajectory of the same duration under the canonical ensemble. The simulations were carried out using large-scale atomic/molecular massively parallel simulator (LAMMPS) [71, 137] with our hNN-Gr$_x$ implemented in

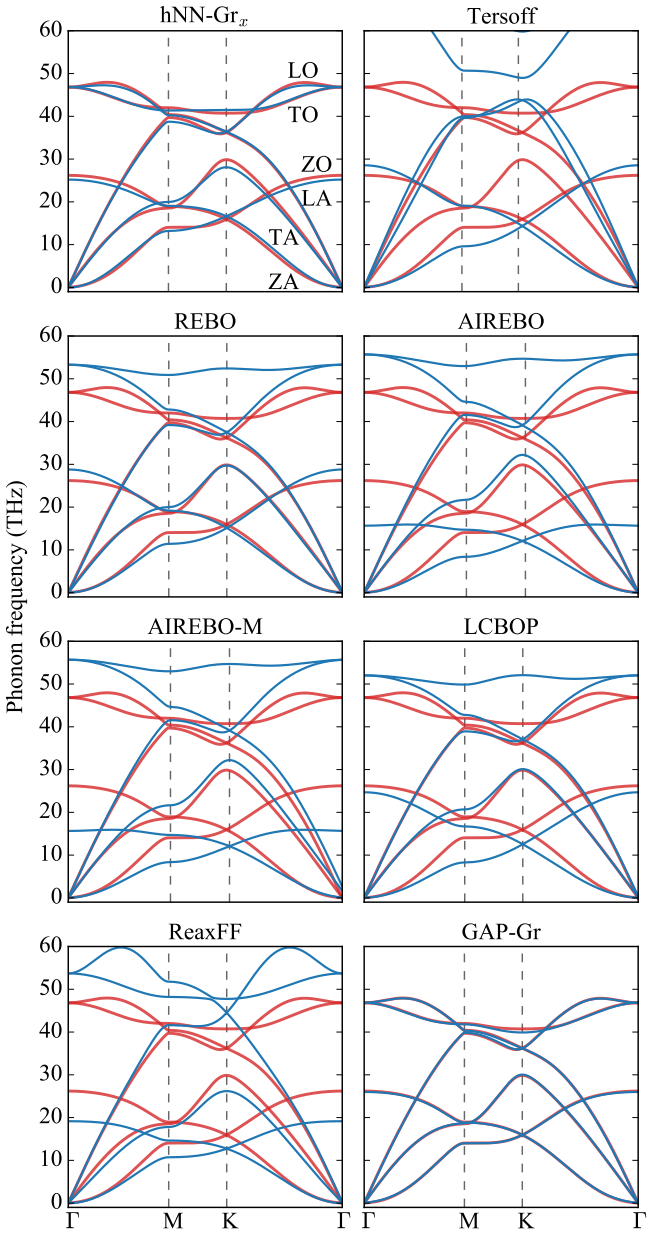


Figure 3.5: Phonon dispersion curves of monolayer graphene along high-symmetry points in the first Brillouin zone. The red curve is the DFT prediction, and the blue curves are results from the potentials. Branch labels are shown in the upper left panel, where "L" stands for longitudinal, "T" for transverse, "Z" for flexural, "O" for optical, and "A" for acoustic. Note that parts of the highest two branches by the Tersoff potential are not shown.

KIM [70, 212], GAP–Gr implemented in QUIP [213], and the other potentials natively built into LAMMPS.[16] While GAP–Gr is nearly 4000 times slower than Tersoff, the hNN-Gr$_x$ potential is much faster, only about 280 times[17] slower than Tersoff. This is a benefit of parametric methods: the evaluation time does not depend on the size of the training set. Both hNN-Gr$_x$ and GAP–Gr are still significantly faster than a first-principles method like DFT, although they are significantly slower than the tested physics-based potentials. KC and DRIP are relatively more expensive than the other physics-based potentials because to model long-range dispersion attraction, they need to use a much larger cutoff distance. For example, DRIP uses a cutoff of 12 Å, whereas the other physics-based potentials considered here typically have cutoffs smaller than 5 Å.

3.3.3 Applications

The hNN-Gr$_x$ potential is applied to two problems of interest that are beyond the capabilities of DFT: (i) thermal conductivity of monolayer graphene; and (ii) interlayer friction in bilayer graphene. In both cases the effect of vacancies on the results are explored.

Thermal conductivity

Graphene has been reported to have extremely high thermal conductivity with experimentally measured values between 1500 and 2500 W/mK [31–35] in suspended samples at room temperature. (For comparison, copper has a thermal conductivity of about 400 W/mK.) Despite these efforts, accurate determination of the thermal conductivity of graphene remains challenging because thermal transport in this material is very sensitive to defects and experimental conditions [214, 215]. Atomistic simulations using IPs provide an alternative approach to study the thermal conductivity in graphene and investigate the effect of defects. In graphene at room temperature, the vast majority

[16] The configuration used in the simulations is monolayer graphene (bilayer graphene for KC and DRIP) consisting of 192 atoms. Both KIM and QUIP have interfaces to LAMMPS, so that their potentials can be used directly. The simulations were performed in serial mode with one core.

[17] For the hNN-Gr$_x$ potential, the relative computational cost of the long-range LJ part to the short-range NN part is 1:93. Within the NN part, the ratio of the time to evaluate the descriptors and the time associated with other computations (e.g. calculating energy and forces) is 75:18. Thus it is clear that the bottleneck is the evaluation of the descriptors.

of the thermal transport is due to lattice vibrations,[18] and thus we focus on the lattice contribution in this work. An accurate prediction of the lattice contribution depends on the ability of the potential to describe the phonon dispersion curves, and in particular the ZA mode associated with out-of-plane vibrations that provides the dominant contribution to the lattice thermal conductivity in suspended graphene [218, 219]. As seen in figure 3.5, the hNN-Gr$_x$ potential is highly accurate in predicting all phonon dispersion branches including ZA.

The thermal conductivity is computed using the Green–Kubo method, an equilibrium MD approach. The Green–Kubo expression, based on linear-response theory, is [220, 221]

$$\kappa_{ij} = \frac{1}{\Omega k_{\mathrm{B}} T^2} \int_0^\infty \langle J_i(t) J_j(0) \rangle \, \mathrm{d}t, \tag{3.70}$$

where $i, j \in \{x, y, z\}$ are Cartesian components, k_{B} is Boltzmann's constant, T is the temperature, $\langle J_i(t) J_j(0) \rangle$ is the heat current auto-correlation (HCA) function expressed as a phase average, and Ω is the volume of the system defined as the area of graphene multiplied by the van der Waals thickness (3.457 Å in the present case; see table 3.2). The upper limit of the integral in equation (3.70) can be approximated by t_P, the correlation time required for the HCA to decay to zero. In the case of an MD simulation, the phase average in the HCA is approximated by a time average computed at discrete MD time steps. Consequently, equation (3.70) is in fact a summation and we actually compute [221]

$$\kappa_{ij}(t_P) = \frac{\Delta t}{\Omega k_{\mathrm{B}} T^2} \sum_{p=1}^{P} (Q - p)^{-1} \sum_{q=1}^{Q-p} J_i(p + q) J_j(q), \tag{3.71}$$

where Δt is the MD time step, Q is the total number of steps, $P = t_P / \Delta t$ is the number of steps for integration (should be smaller than Q), and $J_i(p + q)$ is the ith component of the heat current at step $p + q$.

A key component of the Green–Kubo method is the definition of the heat current. We note that the heat current implemented in the LAMMPS MD code [71, 137] is

[18] The electronic contribution is estimated to be 1% according to the Wiedemann–Franz law [216]. A later DFT study shows that the Wiedemann–Franz law is broadly satisfied at low and high temperatures but deviates largely at room temperatures [217]. Even in the latter case, the lattice contributions still accounts for about 90% of the total thermal conductivity.

intended for pair potentials only. For many-body potentials, such as the hNN-Gr$_x$ potential, using the LAMMPS expression can lead to incorrect results.[19] In this work, we use the definition in [222], which applies to arbitrary many-body potentials.

We study the thermal conductivity in pristine graphene and investigate the impact of defects. In practice, graphene can contain a variety of defects including single vacancies, double vacancies, Stone–Wales defects, adatoms, dislocations, and grain boundaries [223, 224]. Here, we focus on single vacancies, which have been experimentally shown to be a common type of defect in graphene [225]. The base graphene system consists of a periodic rectangular supercell of size 51.25 Å by 49.32 Å in the x (armchair) and y (zigzag) directions comprised of 960 atoms. Separate calculations showed that this system is sufficiently large to obtain converged thermal conductivity for ideal graphene in agreement with previously published results in [219]. Single vacancies are generated by randomly removing atoms from the supercell. The equations of motion are integrated using a velocity-Verlet algorithm with a time step of $\Delta t = 1$ fs. The system is initially thermalized for 0.5 ns at a constant temperature of $T = 300$ K under NVT conditions (canonical ensemble) using a Langevin thermostat. The thermostat is then switched off and data for the Green–Kubo expression is collected under NVE conditions (microcanonical ensemble). A time scale on the order of nanoseconds is necessary to sufficiently converge the HCA function [221]. We ran the NVE simulation for 10 ns based on previous studies of thermal conductivity in graphene [44, 226].

The thermal conductivity in the x (armchair) direction, κ_{xx}, as a function of t_P for pristine graphene, graphene with a 0.1% vacancy density (one vacancy per supercell), graphene with a 0.2% vacancy density (two vacancies per supercell) is plotted in figure 3.6. In each case, the thermal conductivity is computed by averaging over eight uncorrelated trajectories with different initial conditions. We see that the majority of the samples are well converged after $t_P = 0.5$ ns, with the mean showing an even better convergence. The thermal conductivity of pristine graphene measured at $t_P = 0.5$ ns is 2531 W/mK, in good agreement with the experimental values of 1500–2500 W/mK for suspended graphene [31–35]. The thermal conductivity for the graphene with a 0.1% vacancy density is 415 W/mK, an 84% reduction, and for graphene with a 0.2% vacancy

[19] See [86] for a comparison of the thermal conductivity obtained using different definitions of the heat current for the Tersoff potential [113, 147].

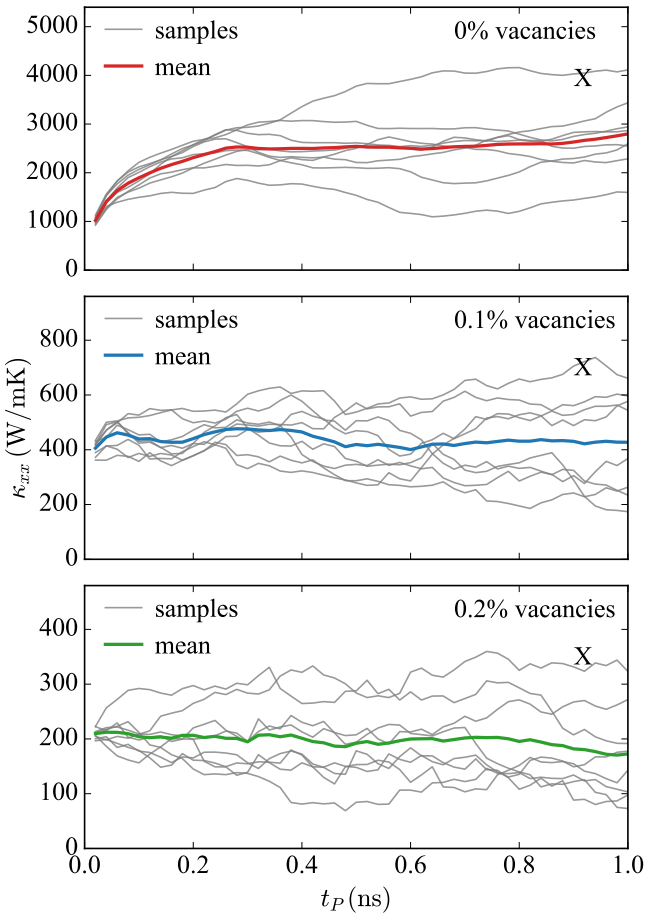


Figure 3.6: Thermal conductivity in the x direction, κ_{xx}, as a function of t_P for pristine graphene, graphene with 0.1% vacancy density, and graphene with 0.2% vacancy density. In each panel, the thin gray lines are the HCA cumulative averages obtained from eight independent trajectories, and the thick lines (red, blue, or green) are the means of these HCA curves. The "X" denotes the sample with the largest κ_{xx} at $t_P = 1$ ns among the eight samples whose normalized HCA is shown in figure 3.7.

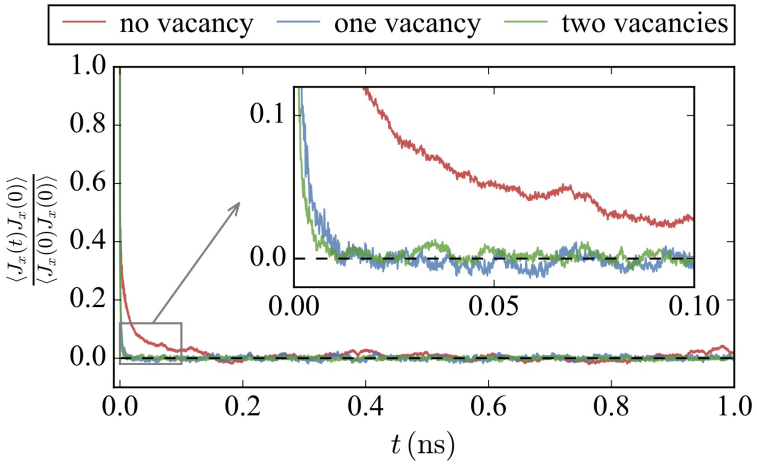


Figure 3.7: Normalized HCA, $\langle J_x(t)J_x(0)\rangle \,/\, \langle J_x(0)J_x(0)\rangle$, as a function of time t for pristine graphene, graphene with 0.1% vacancy density, and graphene with 0.2% vacancy density. The red, blue, and green curves are for the samples marked with an "X" for graphene with 0, 0.1% and 0.2% vacancy density in figure 3.6.

density it is 195 W/mK, a 92% reduction. Similar values were obtained in the y (zigzag) direction, i.e. $\kappa_{yy} \approx \kappa_{xx}$ as expected due to isotropy in the graphene plane.

In figure 3.7, we plot the normalized HCA, $\langle J_x(t)J_x(0)\rangle \,/\, \langle J_x(0)J_x(0)\rangle$, for the samples marked with an "X" in figure 3.6. It is clear that the normalized HCA decays to zero much earlier than $t = 0.5$ ns for all three types of graphene, indicating that $t_P = 0.5$ ns is sufficient for calculating the thermal conductivity. Further, the decay of the normalized HCAs for graphene containing vacancies is much faster than that of pristine graphene, which is related to the fact that the thermal conductivity in defective graphene is much smaller than in pristine graphene. (Note that $\langle J_x(0)J_x(0)\rangle$ is almost the same for all three cases and thermal conductivity is the integral of the HCA). The underlying mechanism for the reduced thermal conductivity of graphene with vacancies is that vacancy defects are a strong scattering source for phonons, which govern heat transport in this system. Creation of a single vacancy leaves three carbon atoms with two-fold coordination, effectively breaking the sp^2 characteristics of the local lattice. These two-fold coordinated atoms are less likely to follow the normal pattern of vibrations in pristine graphene and cause a significant degree of scattering [226].

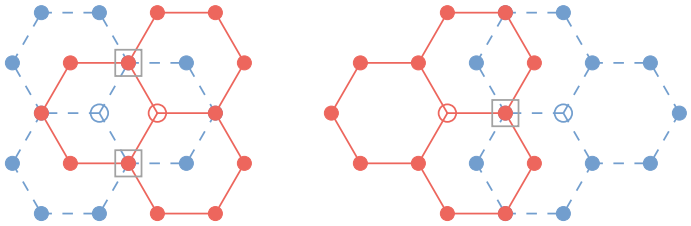


Figure 3.8: Close-proximity divacancies in adjacent layers of AB-stacked bilayer graphene that favor the formation of covalent bonds between layers. Hollow circles denote vacancies, and gray squares are locations where covalent bonds can form between atoms in adjacent layers. (There are two atoms in each gray square; the blue atom in the bottom layer is hidden by the red atom in the top layer.) Following the notation in [227], the subscript 2 in $V_2^1(\beta\beta)$ and $V_2^2(\beta\beta)$ indicates that two single vacancies form a divacancy, the superscripts 1 and 2 denote first- and second-nearest interlayer neighbors, and β means that a vacancy is located at the hexagonal ring center of the other layer.

Interlayer friction

Although the bonding between layers in multilayer graphene is weak, the material still exhibits significant resistance to sliding due to orbital overlap between layers. The friction becomes even larger when covalent bonds are formed between adjacent layers. Such bonds have been proposed to occur when vacancies exist in close proximity to each other in the top and bottom layers and react to form covalent bonds in their vicinity [227]. A plausible mechanism for this to happen is the creation of vacancies through high-energy ion or electron bombardment of multilayer graphene [228]. Here, we study the effect of vacancies and interlayer covalent bonding on friction in bilayer graphene.

A number of possible interlayer divacancies can form via the coalescence of single vacancies in adjacent layers leading to the formation of covalent bonds [227, 229]. We focus on the two structures shown in figure 3.8, where the two vacancies are first- and second-nearest interlayer neighbors referred to as $V_2^1(\beta\beta)$ and $V_2^2(\beta\beta)$ (see the figure caption for an explanation of the notation).

Graphene bilayers containing the two types of divacancies $V_2^1(\beta\beta)$ and $V_2^2(\beta\beta)$ are fully relaxed using DFT and the hNN-Gr$_x$ potential. An important point is that in

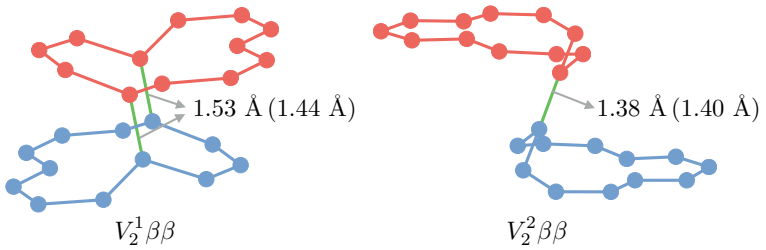


Figure 3.9: Core structures of the $V_2^1(\beta\beta)$ and $V_2^2(\beta\beta)$ divacancies after relaxation. The interlayer covalent bond(s) formed near the divacancy are colored green. The bond length predicted by DFT (hNN-Gr$_x$) is shown.

order for covalent bonds to form between layers it is necessary to compress the bilayer in the direction perpendicular to the layers, so that the layers are brought to within a spacing of about 2.4 Å prior to relaxation. Both DFT and the hNN-Gr$_x$ potential predict the same core structure after relaxation as shown in figure 3.9. Two interlayer covalent bonds of equal length (colored green) are formed in the first-nearest-neighbor divacancy ($V_2^1(\beta\beta)$). The bond length is predicted by the hNN-Gr$_x$ potential to be 1.44 Å, which is good agreement with the DFT value of 1.53 Å. The formation of the covalently-bonded divacancy leaves a two-fold coordinated atom in each layer, which is electronically unsaturated and could be chemically active. For the second-nearest-neighbor divacancy ($V_2^2(\beta\beta)$) only one bridging bond is formed with a length of 1.40 Å according to the hNN-Gr$_x$ potential. Again there is good agreement with DFT, which predicts a bond length of 1.38 Å. As expected the single bond is stronger than the pair of bonds for the first-nearest-neighbor divacancy as demonstrated by the shorter bond length in this case. The $V_2^2(\beta\beta)$ divacancy leaves two two-fold coordinated atoms in each layer, which reconstruct to form a bond (not shown) with a bond length predicted to be 1.84 Å by hNN-Gr$_x$ and 2.15 Å by DFT. (The two atoms are 2.466 Å away from each other in pristine graphene.)

Next, we measure the interlayer friction force in bilayer graphene with and without the two types of divacancies. The setup for this simulation is shown in figure 3.10 for the armchair direction. A graphene layer (red) is placed on top of a larger layer (blue) and pulled to the right under displacement control conditions. The bottom layer has a width of 76.88 Å (in the x direction) and height 22.19 Å (in the y direction) and contains

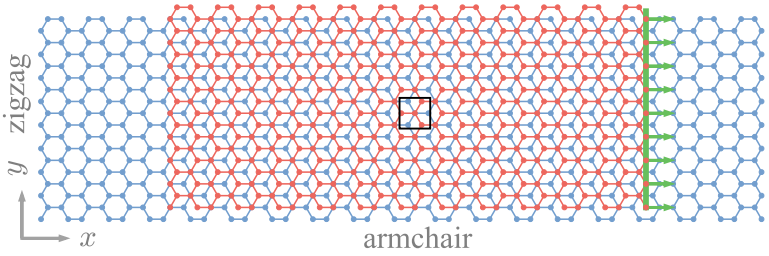
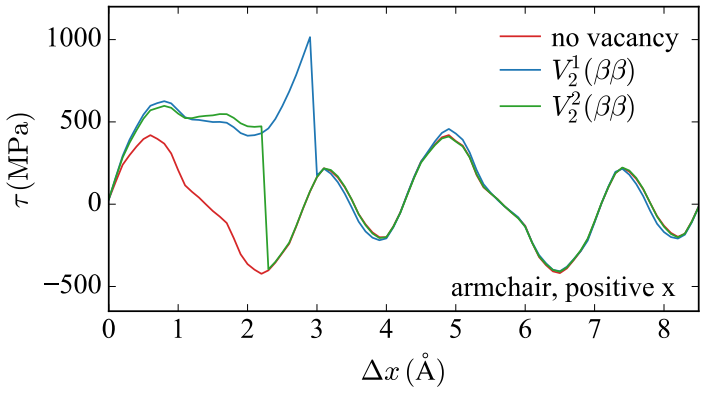


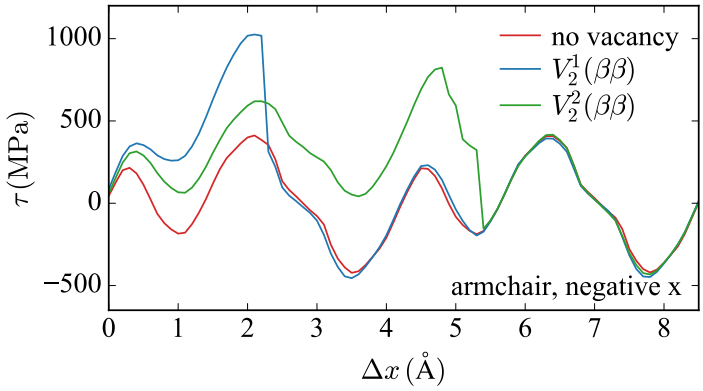
Figure 3.10: Representation of the simulation supercell used to compute the friction force in bilayer graphene with and without covalently-bonded divacancies in adjacent layers. The force required to pull the top layer to the right along the armchair direction is measured. The black rectangle indicates the location of divancncies when included.

648 atoms. The top layer has a width of 49.83 Å and 432 atoms. When divacancies are included, they are introduced into the center of the bilayer at the location indicated by the black rectangle in figure 3.10. Periodic boundary conditions are applied in the x and y directions, and the direction perpendicular to the plane is free. Thus the system corresponds to an infinite graphene nanoribbon with finite width in the x-direction (top layer) sliding on an infinite graphene layer (bottom). The atoms at the right end of the top layer (green shaded region) are displaced in the x direction with a step size of 0.1 Å. At each step, after applying the displacement to these atoms, the total energy of the system is minimized subject to the following constraints: (1) The atoms at the right end of the bottom layer are fixed in all three directions; and (2) the x coordinates of the atoms at the right end of the top layer are fixed to their displaced positions. Following relaxation, the force F required to hold the top layer in its displaced position is computed as the total force acting on the constrained atoms in the top layer. From this the shear stress is computed as $\tau = F/A$, where A is the area of the top layer. The shear stress is a more useful property than the force since it can be more readily compared across systems.

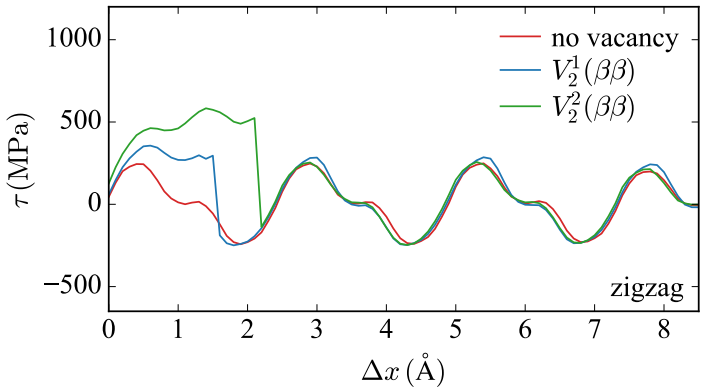
Figure 3.11a shows τ as a function of the pulling distance Δx along the positive armchair direction. For a pristine bilayer without vacancies, the maximum shear stress is 423 MPa at $\Delta x = 0.6$ Å with a periodicity of $\sqrt{3}a = 4.27$ Å reflecting the underlying periodic nature of the bilayer structure. Note that the shear stress is negative once the top layer passes the unstable equilibrium state where it is balanced between forces pulling it forward and backwards. The maximum shear stress for $V_2^1(\beta\beta)$ is 1014 MPa



(a)



(b)



(c)

Figure 3.11: Shear stress τ of friction versus pulling distance Δx for bilayer graphene with and without divacancies. Three different pulling directions are shown (see figure 3.10): (a) and (b) armchair edge in the positive and negative x directions, and (c) zigzag edge in the y direction (positive and negative are the same).

at $\Delta x = 2.9$ Å. The interlayer bond breaks immediately once the shear stress reaches this maximum, leading to an abrupt drop in the shear stress. In contrast for the $V_2^2(\beta\beta)$ divacancy, the interlayer bond does not break at the maximum shear stress of 597 MPa at $\Delta x = 0.8$ Å, but instead breaks later at a somewhat lower shear stress at $\Delta x = 2.2$ Å. Once the interlayer bonds are broken, the $V_2^1(\beta\beta)$ and $V_2^2(\beta\beta)$ curves follow the pristine bilayer curve almost identically. This suggests that the presence of single vacancies in the layers (in the absence of interlayer covalent bonding) has a negligible effect on friction.

We expect the shear stress for pristine graphene to depend on the pulling direction due to the changing crystallographic orientation. The effect of the divacancies will also depend on orientation. For example, referring to figure 3.8, we see that when pulling the top layer to the right, the single vacancies in $V_2^1(\beta\beta)$ move apart, whereas when pulling to the left they initially move closer together. We explore friction anisotropy by considering two more directions in figure 3.10: (1) pulling to the left along the armchair direction, and (2) pulling upwards along the zigzag direction (downwards is the same due to symmetry). In the first case, the simulation setup is the same as in figure 3.10, except that the atoms on the left end of the top layer are pulled in the negative x direction. In the second case, a bilayer is constructed with similar geometry to figure 3.10, but with the zigzag edge aligned with the x direction and the armchair edge aligned with the y direction. This system contains 370 atoms in the top layer and 560 atoms in the bottom layer.

The shear stress versus pulling distance for these two cases are shown in figures 3.11b and 3.11c. The results in the negative armchair direction (figure 3.11b) are similar to those in the positive armchair direction (figure 3.11a), but with some differences. The maximum shear stress for pristine graphene is the same as in figure 3.11a due to symmetry, but for $V_2^1(\beta\beta)$ it is 1018 MPa at $\Delta x = 2.2$ Å, which is still larger than that for $V_2^2(\beta\beta)$, 824 MPa at $\Delta x = 4.8$ Å. However, in this orientation $V_2^1(\beta\beta)$ breaks earlier (and immediately as before), whereas $V_2^2(\beta\beta)$ exhibits a large amount of slip prior to bond failure. For the zigzag direction in figure 3.11c, the shear stress for pristine bilayer has a periodicity of 2.466 Å (smaller than that in figures 3.11a and 3.11b). The maximum shear stress for the pristine bilayer, $V_2^1(\beta\beta)$, and $V_2^2(\beta\beta)$ are 248 MPa, 352 MPa, and 583 MPa, respectively, all smaller than their counterparts in figure 3.11a and figure 3.11b. This direction has the lowest friction resistance.

3.3.4 Summary

We have developed a hybrid NN potential for multilayer graphene structures called "hNN-Gr$_x$." This potential employs an NN to capture the short-range intralayer covalent bonds and interlayer orbital overlap interactions, and a theoretically-motivated r^{-6} term to model the long-range interlayer dispersion. The potential parameters are determined by training against a large dataset of energies and forces for monolayer graphene, bilayer graphene, and graphite in various states. The training set is computed from DFT using the PBE functional augmented with the MBD dispersion correction to account for long-range vdW interactions.

The potential was tested against a variety of structural, energetic, and elastic properties to which it was not directly fit. The validation tests show that:

1. The hNN-Gr$_x$ potential correctly predicts the in-plane lattice parameter, equilibrium layer spacings, interlayer binding energies, and generalized stack fault energies for multilayer graphene structures. An important feature is that it can distinguish the energies of bilayer graphene in the AA and AB stacking states.

2. The hNN-Gr$_x$ potential has good agreement with DFT for the C_{11} and C_{12} elastic moduli for both graphene and graphite. For the other elastic moduli of graphite the agreement is reasonable for C_{33} and C_{44}, but poor for C_{13}. (We note however that DFT results are inconsistent with experiments in the latter case.)

3. The phonon dispersion curves calculated from the hNN-Gr$_x$ potential are in excellent agreement with DFT result, significantly better than any other empirical potential, except for GAP-Gr (which is also a machine learning potential). We note that GAP-Gr is limited to single-layer graphene.

The hNN-Gr$_x$ potential was applied to several large-scale applications, not amenable to DFT calculations. The thermal conductivity of monolayer graphene with different vacancy densities is computed using a Green–Kubo approach. The thermal conductivity of pristine graphene is found to be 2531 W/mK, consistent with experimental measurements (1500–2500 W/mK). The thermal conductivity is dramatically reduced with the addition of vacancies due to phonon scattering: 415 W/mK for a vacancy density of 0.1%, and 195 W/mK for 0.2%.

In a second application, the effect of covalent bonds between layers in bilayer graphene on friction is explored. Such bonds are predicted to occur when vacancies in separate layers exist in close proximity and the bilayer is compressed. The hNN-Gr$_x$ potential predicts the formation of interlayer covalent bonds and a corresponding divacancy structure in agreement with DFT. It is found that the presence of these bonds increases the friction between layers by up to a factor of four depending on the sliding direction.

We have shown that the new hNN-Gr$_x$ potential provides a complete and accurate description of both the intralayer and interlayer interactions in multilayer graphene structures. It can be used to study mechanical and thermal properties of these materials, and investigate the effects of defects. Unlike interlayer potentials like KC [150] and DRIP [68] this potential does not assign atoms membership to layers or assume a layered structure to characterized the registry geometry. Thus, for example, hNN-Gr$_x$ could be used to model passage of atoms between layers.

Chapter 4

Uncertainty Quantification in Potentials

Historically, atomistic simulation with interatomic potentials (IPs) is viewed as a tool limited to provide only qualitative insight. A key reason is that in such simulations there are many sources of uncertainty that are difficult to quantify, thus failing to give confidence interval on the results obtained from simulations [14]. The uncertainty in atomistic simulations with IPs can be categorized into three types: (i) *numerical uncertainty*, (ii) *structural uncertainty*, and (iii) *parametric uncertainty*, although some researchers use slightly different terms [9,10,14]. Numerical uncertainty originates from the particular computational setup, including finite time of sampling, size of integration time step, size of simulation box, etc. We focus on the structural uncertainty and parametric uncertainty that originate from IPs in this chapter. Structural uncertainty refers to several approximations in an IP, reflected in the mathematical form of the IP. Parametric uncertainty refers to the lack of knowledge in the precise values of the parameters in the mathematical form of the IP.

To make atomistic simulations more trustable and to obtain quantitative simulation results, it is imperative to quantify the uncertainty in IPs and propagate it to simulation results. Although atomistic simulations date back to the 1950s [230], only recently efforts have been put to carry out uncertainty quantification and propagation (UQ+P) for atomistic simulations with IPs. In this chapter, we first review some of the latest

work in this filed, and then present two new methods that we developed to conduct UQ+P in atomistic simulations. The first is based on Fisher information theory, and we apply it to the Stillinger–Weber (SW) potential for MoS_2 discussed in section 2.2. The second is tied to the dropout technique used in neural network (NN), and we apply the dropout technique to the neural network interatomic potential (NNIP) for multilayer graphene structures discussed in section 3.3.

4.1 Approaches for uncertainty quantification

4.1.1 Fisher information theory

Let $p(\boldsymbol{z}\,|\,\boldsymbol{\theta})$ be the probability distribution of a random variable $\boldsymbol{z}$ conditioned on the values of $\boldsymbol{\theta}$, the *score* is defined as the partial derivative of the natural logarithm of this distribution with respect to $\boldsymbol{\theta}$ [231],

$$\boldsymbol{s} = \frac{\partial}{\partial \boldsymbol{\theta}} \ln p(\boldsymbol{z}\,|\,\boldsymbol{\theta}). \tag{4.1}$$

The expectation of the score is $\mathbf{0}$, i.e. $\mathbb{E}_{\boldsymbol{z}}[\boldsymbol{s}] = \mathbf{0}$, and the *Fisher information* is defined as the variance of the score:

$$\mathcal{I}(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{z}}\left[\left(\frac{\partial}{\partial \boldsymbol{\theta}} \ln p(\boldsymbol{z}\,|\,\boldsymbol{\theta})\right)^2\right] = -\mathbb{E}_{\boldsymbol{z}}\left[\frac{\partial^2}{\partial \boldsymbol{\theta}^2} \ln p(\boldsymbol{z}\,|\,\boldsymbol{\theta})\right], \tag{4.2}$$

where the last equality assumes $\ln p(\boldsymbol{z}\,|\,\boldsymbol{\theta})$ is twice differentiable. The Fisher information is a measure of the amount of information that the observable random variable $\boldsymbol{z}$ carries about the unknown parameter $\boldsymbol{\theta}$.

The Fisher information is closely related to the Kullback–Leibler (KL) divergence (also called *relative entropy*). The KL divergence between two distribution $p(\boldsymbol{z})$ and $q(\boldsymbol{z})$ is [231–233]

$$D_{\mathrm{KL}}(p(\boldsymbol{z}) \| q(\boldsymbol{z})) = \int_{-\infty}^{\infty} p(\boldsymbol{z}) \frac{p(\boldsymbol{z})}{q(\boldsymbol{z})} \mathrm{d}\,\boldsymbol{z}. \tag{4.3}$$

Let's consider the KL divergence between a parametric distribution with perturbed parameters $p(\boldsymbol{z}\,|\,\boldsymbol{\theta}+\Delta\boldsymbol{\theta})$ and the parametric distribution itself $p(\boldsymbol{z}\,|\,\boldsymbol{\theta})$, i.e. $g = D_{\mathrm{KL}}(p(\boldsymbol{z}\,|\,\boldsymbol{\theta}+$

$\Delta\boldsymbol{\theta})\|p(\boldsymbol{z}\,|\,\boldsymbol{\theta}))$. Expanding g at $\Delta\boldsymbol{\theta}=\mathbf{0}$, we have

$$g = g|_{\Delta\boldsymbol{\theta}=\mathbf{0}} + \Delta\boldsymbol{\theta}^{\mathrm{T}}\frac{\partial g}{\partial\boldsymbol{\theta}}\bigg|_{\Delta\boldsymbol{\theta}=\mathbf{0}} + \frac{1}{2}\Delta\boldsymbol{\theta}^{\mathrm{T}}\,\frac{\partial^2 g}{\partial\boldsymbol{\theta}^2}\bigg|_{\Delta\boldsymbol{\theta}=\mathbf{0}}\Delta\boldsymbol{\theta} + \mathcal{O}(\|\Delta\boldsymbol{\theta}\|^3). \tag{4.4}$$

The first and second terms are both $\mathbf{0}$ since the KL divergence g achieves its minimum $\mathbf{0}$ at $\Delta\boldsymbol{\theta}=\mathbf{0}$, and it can be shown that

$$\frac{\partial^2 g}{\partial\boldsymbol{\theta}^2}\bigg|_{\Delta\boldsymbol{\theta}=\mathbf{0}} = \mathcal{I}(\boldsymbol{\theta}). \tag{4.5}$$

Therefore,

$$D_{\mathrm{KL}}(p(\boldsymbol{z}\,|\,\boldsymbol{\theta}+\Delta\boldsymbol{\theta})\|p(\boldsymbol{z}\,|\,\boldsymbol{\theta})) = \frac{1}{2}\Delta\boldsymbol{\theta}^{\mathrm{T}}\mathcal{I}(\boldsymbol{\theta})\Delta\boldsymbol{\theta} + \mathcal{O}(\|\Delta\boldsymbol{\theta}\|^3), \tag{4.6}$$

indicating that the Fisher information represents the curvature of the KL divergence.

For the least-squares minimization problem discussed in section 2.1.2, the likelihood function takes the form (i.e. equation (2.12))

$$p(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{\theta}) = \frac{1}{\sqrt{(2\pi)^M|\boldsymbol{\Sigma}|}}\exp\left[-\frac{1}{2}(\boldsymbol{y}-\boldsymbol{h})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\boldsymbol{y}-\boldsymbol{h})\right], \tag{4.7}$$

where $\boldsymbol{h}=\boldsymbol{h}(\boldsymbol{x};\boldsymbol{\theta})$ is a parametric model. The Fisher information (equation (4.2)) for this distribution is (see appendix C for a derivation)

$$\mathcal{I}(\boldsymbol{\theta}) = \mathbb{E}_{\boldsymbol{y}}\left[\left(\frac{\partial\boldsymbol{h}}{\partial\boldsymbol{\theta}}\right)^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\left(\frac{\partial\boldsymbol{h}}{\partial\boldsymbol{\theta}}\right)\right]. \tag{4.8}$$

Using the Monte Carlo (MC) sampling technique, we can estimate the above expectation as [97, 234]

$$\mathcal{I}(\boldsymbol{\theta}) \approx \frac{1}{N}\sum_{i=1}^{N}\left(\frac{\partial\boldsymbol{h}_i}{\partial\boldsymbol{\theta}}\right)^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\left(\frac{\partial\boldsymbol{h}_i}{\partial\boldsymbol{\theta}}\right), \tag{4.9}$$

where the sum runs over the whole data set comprised of N observations.

Both the outputs of the model $\boldsymbol{h}$ and the parameters $\boldsymbol{\theta}$ are typically finite; consequently, the Fisher information can be represented as a $N_p\times N_p$ matrix $\boldsymbol{F}(\boldsymbol{\theta})$, where N_p is the length of the parameter vector $\boldsymbol{\theta}$. The diagonal elements of the inverse Fisher

information matrix (FIM) provide lower bounds on the variance of any unbiased estimator $\hat{\boldsymbol{\theta}}$ for the parameters (e.g. the least-squares estimator discussed in section 2.1.2), known as the Cramér–Rao bound [125, 126],

$$\mathrm{Var}_{\boldsymbol{\theta}}[\hat{\theta}_i] \geq [\boldsymbol{F}^{-1}]_{ii}. \tag{4.10}$$

The KL divergence can provide an upper bound for a large family of observables via the Csiszár–Kullback–Pinsker inequality [235],

$$|\mathbb{E}_{\boldsymbol{\theta}+\Delta\boldsymbol{\theta}}[P] - \mathbb{E}_{\boldsymbol{\theta}}[P]| \leq \|P\|_\infty \sqrt{2D_{\mathrm{KL}}(p(\boldsymbol{z}\,|\,\boldsymbol{\theta})\|p(\boldsymbol{z}\,|\,\boldsymbol{\theta}+\boldsymbol{\Delta}\theta))}, \tag{4.11}$$

where $\Delta\boldsymbol{\theta}$ is a perturbation of the parameter vector $\boldsymbol{\theta}$, and $\|P\|_\infty$ denotes the supremum of an observable P. Submitting equation (4.6) into equation (4.11) and ignore the high order terms, we have

$$|\mathbb{E}_{\boldsymbol{\theta}+\Delta\boldsymbol{\theta}}[P] - \mathbb{E}_{\boldsymbol{\theta}}[P]| \leq \|P\|_\infty \sqrt{\Delta\boldsymbol{\theta}^{\mathrm{T}}\mathcal{I}(\boldsymbol{\theta})\Delta\boldsymbol{\theta}}. \tag{4.12}$$

A tighter version of the Csiszár–Kullback–Pinsker inequality has recently been put forward [120]:

$$|\mathbb{E}_{\boldsymbol{\theta}+\Delta\boldsymbol{\theta}}[P] - \mathbb{E}_{\boldsymbol{\theta}}[P]| \leq \sqrt{\mathrm{Var}_{\boldsymbol{\theta}}[P]} \sqrt{\Delta\boldsymbol{\theta}^{\mathrm{T}}\mathcal{I}(\boldsymbol{\theta})\Delta\boldsymbol{\theta}}, \tag{4.13}$$

where $\mathrm{Var}_{\boldsymbol{\theta}}[P]$ denotes the variance of the observable P.

With FIM, we can obtain bounds on interatomic potential (IP) parameters and observables computed from IPs using the Cramér–Rao inequality and the Csiszár–Kullback–Pinsker inequality, respectively. These inequalities have analytical forms, making them relatively easy to evaluate. The downside of this approach, however, is that it only explores the parameter space in the vicinity of the best fit, which may fail to provide information on how well an IP behaves in general. An example of applying the FIM based method to quantify the uncertainty in the Stillinger–Weber (SW) potential discussed in section 2.2 is provided below in section 4.2.

4.1.2 Frequentist statistics

Frequentist statistics based on the F-statistic was adopted by Messerly *et al.* [14] to investigate the effects of parametric uncertainty in a Lennard–Jones (LJ) potential. The fundamental equation of this approach is [236]

$$S(\boldsymbol{\theta}) - S(\hat{\boldsymbol{\theta}}) < N_p s^2 F_{N_p,\nu,\alpha}, \tag{4.14}$$

where $S(\boldsymbol{\theta})$ is a sum of the squared errors for some properties evaluated using a specific parameter set $\boldsymbol{\theta}$ (a choice of S could be the loss defined in equation (2.10)), $S(\hat{\boldsymbol{\theta}})$ is that evaluated at the optimal parameter set $\hat{\boldsymbol{\theta}}$, N_p is the number of parameters, s^2 is an independent estimate of the inherent variance having ν degrees of freedom, and $F_{N_p,\nu,\alpha}$ is the F-statistic at the α confidence level with N_p and ν degrees of freedom. The parameter set $\boldsymbol{\theta}$ is acceptable at the $\alpha\%$ confidence level if equation (4.14) is satisfied.

The uncertainty in the LJ parameters is propagated to the saturated liquid density using a standard MC sampling approach [14]. To carry out the MC sampling, one needs a probability density function in the parameter space, and this is achieved by rearranging equation (4.14) to obtain

$$p(\boldsymbol{\theta}) = F^{-1}_{N_p,\nu}\left(\frac{S(\boldsymbol{\theta}) - S(\hat{\boldsymbol{\theta}})}{N_p s^2}\right), \tag{4.15}$$

where $F^{-1}_{N_p,\nu}$ is the inverse of the F-statistic with N_p and ν degrees of freedom. Once an ensemble of IP parameters are sampled, for any observable P that the can be computed from atomistic simulations, the ensemble mean $\langle P \rangle$ and variance σ_P^2 can then be obtained.

4.1.3 Bayesian statistics

Many work fall into the category of Bayesian statistics. To my best knowledge, in the field of uncertainty quantification in atomistic simulations, Bayesian statistics was first employed by Frederiksen *et al.* [237], where they directly assume the form of the

posterior distribution, conditioned on the data set $\mathcal{D}$ and the model M, to be

$$p(\boldsymbol{\theta}|\mathcal{D}, M) \propto \exp\left[-\frac{\mathcal{L}(\boldsymbol{\theta})}{T}\right], \tag{4.16}$$

where $\mathcal{L}(\boldsymbol{\theta})$ is a loss function (e.g. equation (2.10)), and T is a temperature introduced to formalize the weighting of different parameter sets. An ensemble of IP parameters can be generated from equation (4.16) given a temperature T, a data set $\mathcal{D}$, and a model M. Therefore, for any observable P, the ensemble mean $\langle P\rangle|_{T,D,M}$ and variance $\sigma_P^2|_{T,D,M}$ can then be obtained. To generate an ensemble according to equation (4.16), the authors also employed the MC sampling technique. They noticed that the curvatures of the loss function in different directions vary enormously in the region near the optimal parameter set and called for special care to use MC to get an efficient sampling. To this end, they rescaled the MC trail moves using the calculated Hessian in their test for the modified embedded atom method (MEAM) potential [237].

The methods by Messerly *et al.* [14] and Frederiksen *et al.* [237] resemble each other in the sense that a distribution of the IP parameters, $p(\boldsymbol{\theta})$, is directly obtained (or assumed) and then this distribution is sampled to generate an ensemble of parameter sets so as to propagate the uncertainty in IPs to simulation results. The Frederiksen *et al.* approach is more flexible though, because it incorporates a hyperparameter T to control the distribution.

Most uncertainty quantification and propagation (UQ+P) works undertake the full Bayesian approach that constructs the posterior distribution over IP parameters from a prior and a likelihood. We assume that the observed true data from first-principles calculations or experiments y is given by the model prediction $h(\boldsymbol{x};\boldsymbol{\theta})$ with an additive error ϵ, i.e.

$$y = h(\boldsymbol{x};\boldsymbol{\theta}) + \epsilon. \tag{4.17}$$

Here, we use general notation not specific to IPs, but one can just regard the function h as an IP mathematical form and $\boldsymbol{x}$ as the collections of coordinates $\boldsymbol{r}_1, \ldots, \boldsymbol{r}_{N_\mathrm{a}}$. A Gaussian distribution with zero mean is a reasonable choice for the error ϵ, consistent with the maximum entropy principle [10], $p(\epsilon) = \mathcal{N}(\epsilon|0, \sigma^2)$, where σ^2 is the covariance of the Gaussian distribution. Consequently, we can build a probabilistic model by which

an input $\boldsymbol{x}$ generates an output y given the parameters $\boldsymbol{\theta}$,

$$p(y|\boldsymbol{x}, \boldsymbol{\theta}) = \mathcal{N}(y|h(\boldsymbol{x}; \boldsymbol{\theta}), \sigma^2). \tag{4.18}$$

This is the *likelihood* distribution for a single data point. Now consider a data set $\mathcal{D} = (\boldsymbol{X}, \boldsymbol{y})$ with inputs $\boldsymbol{X} = [\boldsymbol{x}_1^{\mathrm{T}}; \cdots; \boldsymbol{x}_N^{\mathrm{T}}]$ and the corresponding outputs $\boldsymbol{y} = [y_1; \cdots; y_N]$, where N is the size of the data set. Make the assumption that the data points are drawn independently from the distribution in equation (4.18), we can obtain the *likelihood* for the data set:

$$p(\boldsymbol{y} \,|\, \boldsymbol{X}, \boldsymbol{\theta}) = \prod_{i=1}^{N} p(y_i \,|\, \boldsymbol{x}_i, \boldsymbol{\theta}). \tag{4.19}$$

Given the data set $\mathcal{D}$, we are interested in finding the parameters $\boldsymbol{\theta}$ that are most likely to have generated the outputs $\boldsymbol{y}$ from the inputs $\boldsymbol{X}$. Following the Bayesian approach, the model parameters $\boldsymbol{\theta}$ are considered to be uncertain and we assign a *priori* distribution over the parameter space, $p(\boldsymbol{\theta})$. This distribution represents our prior information as to which parameters are likely to have generated the outputs before observing any data, based on previous knowledge, experience, or physical limitations. We then look for the *posterior* distribution over the parameter space by invoking Bayes' theorem:

$$p(\boldsymbol{\theta} \,|\, \boldsymbol{X}, \boldsymbol{y}) = \frac{p(\boldsymbol{y} \,|\, \boldsymbol{X}, \boldsymbol{\theta}) p(\boldsymbol{\theta})}{p(\boldsymbol{y} \,|\, \boldsymbol{X})}, \tag{4.20}$$

where $p(\boldsymbol{y} \,|\, \boldsymbol{X})$ is the *model evidence*, given by

$$p(\boldsymbol{y} \,|\, \boldsymbol{X}) = \int p(\boldsymbol{y} \,|\, \boldsymbol{X}, \boldsymbol{\theta}) p(\boldsymbol{\theta}) \, \mathrm{d}\boldsymbol{\theta}. \tag{4.21}$$

The posterior in equation (4.20) tells us how to update our knowledge of the parameters $\boldsymbol{\theta}$ upon observation of the data.

Let $p(P \,|\, \boldsymbol{\theta})$ be the distribution of an observable P conditioned on the model parameters $\boldsymbol{\theta}$. We can propagate the distribution via the Markov equation by a weighted integration over all possible values of the parameters to get the predictive distribution [9]

$$p(P \,|\, \boldsymbol{y}, \boldsymbol{X}) = \int p(P \,|\, \boldsymbol{\theta}) p(\boldsymbol{\theta} \,|\, \boldsymbol{X}, \boldsymbol{y}) \, \mathrm{d}\boldsymbol{\theta}. \tag{4.22}$$

From this predictive distribution, we can readily obtain the mean $\langle P \rangle$ and variance σ_P^2 of the observable.

A key component of the full Bayesian approach outlined above is the evaluation of the model evidence in equation (4.21). This can be done analytically for simple models like linear regression; however, for more interesting and complicated models such as multilayer neural network (NN) discussed in section 3.3, the evaluation cannot be carried out analytically. For such models, evaluation of the posterior often relies on sampling techniques such as the MC algorithm and its variants.

Following the full Bayesian approach, Cailliez and Pernot [9] studied the uncertainty in some thermodynamical and transport properties for argon, and Angelikopoulos *et al.* [10] investigated the self-diffusion coefficient and viscosity, among others, for argon. Both work used the LJ potential, and a uniform distribution was adopted for the prior, representing the fact that no prior information on the values of the parameters is available. Cailliez and Pernot applied standard MC to sample the posterior parameter space, whereas Angelikopoulos *et al.* employed the transitional Markov chain Monte Carlo (TMCMC) technique to conduct the sampling, which was claimed to address the problem of choosing the right adaptive proposal in Markov chain Monte Carlo (MCMC) for accelerated convergence to the posterior.

Although sampling techniques to evaluate the posterior work, it is computational expensive. The most computationally intensive part is to evaluate the likelihood function in the resampling step, requiring multiple IP evaluations.[1] This is the reason why the demonstrations in most papers use the simplest LJ potential to reduce the computational cost. To address this, we can take advantage of surrogate models, while still achieving highly accurate approximations. The idea is to use some simple, yet faithful, meta model that can be quickly evaluated to represent the mapping between the input and the output, approximating the computationally expensive atomistic simulations. Widely used surrogate models include linear or polynomial regression, least-squares formulation, Gaussian process (GP), and polynomial chaos [10, 238, 239].

Besides the MC based sampling techniques, another powerful method to do Bayesian

[1] For molecular dynamics (MD) simulations, it requires multiple MD runs [10].

inference is the *variational inference* technique. In variational inference, instead of evaluating the posterior directly, one employs another distribution to approximate the posterior and then carries out the Bayesian inference using the easy-to-evaluate approximating distribution. Variational inference does not guarantee to produce (asymptotically) exactly the same samples as from the posterior (it can only find a density close to the posterior), but tends to be much faster than sampling the exact posterior directly [240].

In variational inference, another distribution $q(\boldsymbol{\theta};\boldsymbol{\omega})$, parameterized by $\boldsymbol{\omega}$, is used to approximate the posterior of the original model, $p(\boldsymbol{\theta}\,|\,\boldsymbol{X},\boldsymbol{y})$. The structure of the approximating distribution $q(\boldsymbol{\theta};\boldsymbol{\omega})$ should be known and it needs to be easy to evaluate. The goal of variational inference is to obtain an optimal approximating distribution $q^*(\boldsymbol{\theta};\boldsymbol{\omega})$ as close as possible to the posterior $p(\boldsymbol{\theta}\,|\,\boldsymbol{X},\boldsymbol{y})$, and then use $q^*(\boldsymbol{\theta};\boldsymbol{\omega})$ to replace $p(\boldsymbol{\theta}\,|\,\boldsymbol{X},\boldsymbol{y})$ to make predictions. This former can be achieved by minimizing the KL divergence [231–233] between $q(\boldsymbol{\theta};\boldsymbol{\omega})$ and $p(\boldsymbol{\theta}\,|\,\boldsymbol{X},\boldsymbol{y})$,

$$D_{\mathrm{KL}}(q(\boldsymbol{\theta};\boldsymbol{\omega})\,\|\,p(\boldsymbol{\theta}\,|\,\boldsymbol{X},\boldsymbol{y})) = \int q(\boldsymbol{\theta};\boldsymbol{\omega}) \log \frac{q(\boldsymbol{\theta};\boldsymbol{\omega})}{p(\boldsymbol{\theta}\,|\,\boldsymbol{X},\boldsymbol{y})}\,\mathrm{d}\boldsymbol{\theta}, \tag{4.23}$$

with respect to $\boldsymbol{\omega}$. The KL divergence is a measure of the similarity between the two distributions. After obtaining the optimal approximating distribution $q*(\boldsymbol{\theta};\boldsymbol{\omega})$, we can replace $p(\boldsymbol{\theta}\,|\,\boldsymbol{X},\boldsymbol{y})$ in equation (4.22) by it to obtain

$$p(P\,|\,\boldsymbol{y},\boldsymbol{X}) = \int p(P\,|\,\boldsymbol{\theta}) q^*(\boldsymbol{\theta};\boldsymbol{\omega})\,\mathrm{d}\boldsymbol{\theta}, \tag{4.24}$$

and then compute predictive mean and variance using this approximating predictive distribution.

The dropout neural network interatomic potential (DNNIP) to be discussed in section 4.3 adopts the variational inference approach to quantify the uncertainty, and details on how to use DNNIP to make predictions in atomistic simulations are provided there.

4.2 Fisher information based uncertainty quantification

We perform an uncertainty analysis of the SW-FM potential discussed in section 2.2 using the Fisher information theory extended to path-space distributions [241]. This

analysis has two objectives. First, it provides an estimate for the uncertainty in the SW-FM potential parameters, i.e. how well the parameters are identified from the training set. Second, it provides an estimate for the uncertainty in the predictions of the SW-FM potential for new properties.

For a molecular dynamics (MD) simulation under the NVT ensemble using the Langevin thermostat, the motion of atoms is governed by two stochastic differential equations [235]:

$$\begin{cases} \mathrm{d}\boldsymbol{q}_t = \boldsymbol{M}^{-1}\boldsymbol{p}_t \mathrm{d}t \\ \mathrm{d}\boldsymbol{p}_t = \boldsymbol{f}\mathrm{d}t - \boldsymbol{\gamma}\boldsymbol{M}^{-1}\boldsymbol{p}_t \mathrm{d}t + \boldsymbol{\sigma}\mathrm{d}\boldsymbol{W}_t, \end{cases} \tag{4.25}$$

where $\boldsymbol{q}_t \in \mathbb{R}^{3N_a}$ is the position vector of a configuration of N_a atoms in three-dimensional (3D) space, $\boldsymbol{p}_t \in \mathbb{R}^{3N_a}$ is the momentum vector, $\boldsymbol{f} \in \mathbb{R}^{3N_a}$ is the force vector, $\boldsymbol{M}$ is the $3N_a \times 3N_a$ diagonal mass matrix of the atoms, $\boldsymbol{\gamma}$ is the $3N_a \times 3N_a$ friction matrix, $\boldsymbol{\sigma}$ is the $3N_a \times 3N_a$ diffusion matrix, and $\boldsymbol{W}_t$ is a $3N_a$ vector describing the Brownian motion. According to the fluctuation-dissipation theorem, the friction and diffusion terms are related to the temperature T via [17, 235]

$$\boldsymbol{\sigma}\boldsymbol{\sigma}^{\mathrm{T}} = 2k_{\mathrm{B}}T\boldsymbol{\gamma}, \tag{4.26}$$

where k_B is the Boltzmann constant.

For the system under Langevin dynamics, it has been show in Ref. [235] that the Fisher information matrix (FIM) is

$$\begin{aligned} \boldsymbol{F}(\boldsymbol{\theta}) &= \mathbb{E}_{\mathrm{eq}}\left[\left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{\theta}}\right)^{\mathrm{T}} (\boldsymbol{\sigma}\boldsymbol{\sigma}^{\mathrm{T}})^{-1}\left(\frac{\partial \boldsymbol{f}}{\partial \boldsymbol{\theta}}\right)\right] \\ &\approx \frac{1}{N}\sum_{i=1}^{N}\left(\frac{\partial \boldsymbol{f}_i}{\partial \boldsymbol{\theta}}\right)^{\mathrm{T}} (\boldsymbol{\sigma}\boldsymbol{\sigma}^{\mathrm{T}})^{-1}\left(\frac{\partial \boldsymbol{f}_i}{\partial \boldsymbol{\theta}}\right), \end{aligned} \tag{4.27}$$

where $\boldsymbol{f}_i \in \mathbb{R}^{3N_a}$ is the force vector at the ith MD step. The expectation $\mathbb{E}_{\mathrm{eq}}[\cdot]$ denotes averaging with respect to an equilibrium (stationary) distribution of the observed dynamics. This is approximated by ergodic averaging on an equilibrated trajectory where N is the number of sampled configurations in the training set. We see that the path-space FIM in equation (4.27) has exactly the same form as equation (4.9), if we

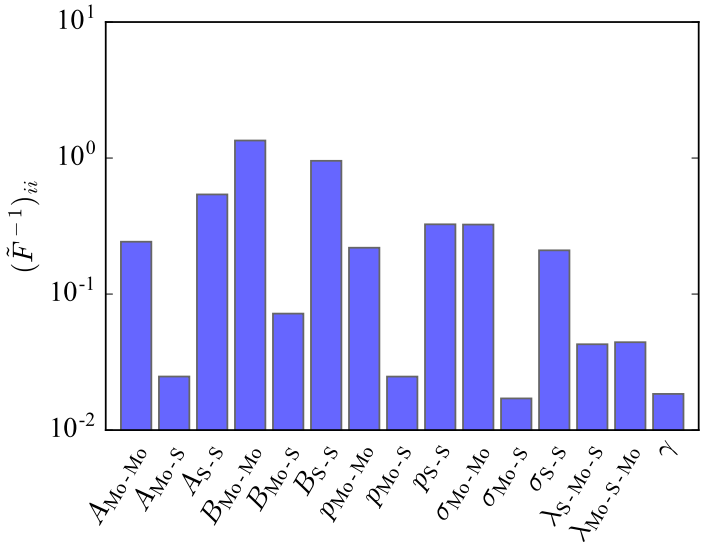


Figure 4.1: The diagonal elements of the inverse FIM in logarithmic space.

fit to forces only and regard $\boldsymbol{\sigma}\boldsymbol{\sigma}^{\mathrm{T}}$ in equation (4.27) as the covariance matrix $\boldsymbol{\Sigma}$ in equation (4.9) .

In real simulations, the friction matrix $\boldsymbol{\gamma}$ is typically a diagonal matrix of the form $\boldsymbol{\gamma} = \eta \boldsymbol{I}$, where η is a scalar and $\boldsymbol{I}$ is the identity matrix. Consequently, equation (4.27) can be simplified as

$$\boldsymbol{F}(\boldsymbol{\theta}) \approx \frac{1}{2k_{\mathrm{B}}T\eta}\frac{1}{N}\sum_{i=1}^{N}\left(\frac{\partial \boldsymbol{f}_i}{\partial \boldsymbol{\theta}}\right)^{\mathrm{T}}\left(\frac{\partial \boldsymbol{f}_i}{\partial \boldsymbol{\theta}}\right), \tag{4.28}$$

The scalar η is associated with the thermostat used to control temperature in the *ab initio* molecular dynamics (AIMD) simulation. In our simulations $\eta = 0.02$ and T =750 K. The path-space FIM was evaluated for the fitted parameters given in tables 2.1 and 2.2. The derivatives of the SW-FM force $\boldsymbol{f}$ with respect to the potential parameters $\boldsymbol{\theta}$ were calculated by finite difference using Ridders' method [242, 243].

4.2.1 Parameter uncertainty

In cases where the magnitudes of the parameters differ greatly (as in our case where the parameters range over more than an order of magnitude), it is helpful to perform

a relative parameter analysis by using the logarithm of the parameters instead of the parameters themselves. Defining $\tilde{\theta}_i = \log \theta_i$, it can be shown (see Appendix E) that the FIM in the logarithmic parameter space is

$$\tilde{F}_{ij} = \theta_i F_{ij} \theta_j. \tag{4.29}$$

In terms of the logarithmic parameter space FIM, the Cramér-Rao bound in equation (4.10) becomes (see Appendix E)

$$\mathrm{Var}_{\boldsymbol{\theta}}[\hat{\theta}_i/\theta_i] \geq [\tilde{\boldsymbol{F}}^{-1}]_{ii}. \tag{4.30}$$

This serves as an estimate for the uncertainty in the obtained parameters in a fractional sense.

The diagonal elements of the inverse FIM, $[\tilde{\boldsymbol{F}}^{-1}]_{ii}$, are plotted in figure 4.1. We see that all elements are within two orders of magnitude of each other and there are no parameters with extremely low values compared with the rest. This suggests that all parameters in the interatomic potential (IP) are identified, and there is no cause to simplify the model by removing undetermined parameters.

Examining the results more closely, we see that for the two-body interaction parameters (A, B, p and σ), the lower bounds for the standard deviation of the logarithmic parameters $\tilde{\boldsymbol{\theta}}$ associated with Mo–S interactions are smaller than their Mo–Mo and S–S counterparts, which loosely indicates that the Mo-S parameters are better determined. This is consistent with our knowledge of the bonding in MoS_2, where the Mo–S bonds are shortest and expected to be strongest [244].

4.2.2 Observable uncertainty

The diagonal elements of the FIM provide an upper bound on the uncertainty due to variations in parameters in any observable predicted by the model that is obtained by averaging with respect to an equilibrium distribution in phase space according to a sharper version of the Csiszár-Kullback-Pinsker inequality [120] (see equation (4.13)). If we only perturb one component of the parameter vector, equation (4.13) can be

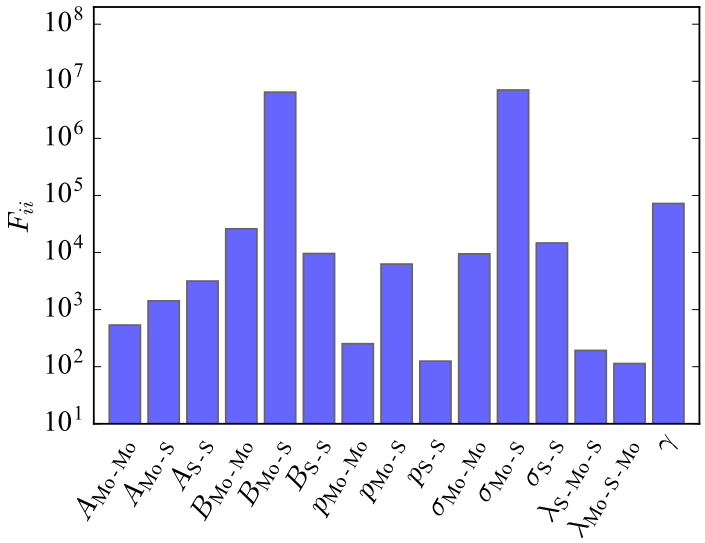


Figure 4.2: The diagonal elements of the FIM.

simplified as

$$|\mathbb{E}_{\boldsymbol{\theta}+\epsilon\boldsymbol{e}_i}[O] - \mathbb{E}_{\boldsymbol{\theta}}[O]| \leq \text{Std}_{\boldsymbol{\theta}}[O]|\epsilon|\sqrt{F_{ii}}, \tag{4.31}$$

where $\mathbb{E}_{\boldsymbol{\theta}}[O]$ is the expectation of an observable O using the parameter set $\boldsymbol{\theta}$, $\text{Std}_{\boldsymbol{\theta}}[O]$ is the corresponding standard deviation, $\boldsymbol{e}_i$ is a unit vector of dimension N_p (where N_p is size of the parameter vector $\boldsymbol{\theta}$) with the ith component equal to one and all others zero. The quantity $\epsilon\boldsymbol{e}_i$ suggests that only the ith component of the parameter vector $\boldsymbol{\theta}$ is perturbed and the others are kept intact. Thus the diagonal of the FIM provides an upper bound on the uncertainty of the predictions of the IP to its fitting parameters. The larger F_{ii}, the more sensitive the predictions are to parameter θ_i.

The diagonal elements of the FIM, F_{ii}, are plotted in figure 4.2. We see that the SW-FM potential is most sensitive to $\sigma_{\text{Mo-S}}$ and least sensitive to $\lambda_{\text{Mo-S-Mo}}$. The ratio of F_{ii} for these two parameters is on the order of 10^5. In particular for the two-body interaction parameters (B, p and σ), as we noted above, the potential is more sensitive to parameters associated with Mo–S interactions than to those associated with Mo–Mo and S–S interactions.

As an example for the bound in equation (4.31), we take the observable $\mathbb{E}_{\boldsymbol{\theta}}[O]$ to be

the mean thickness $\bar{t}$ of a MoS_2 sheet,

$$\bar{t} = \mathbb{E}_{\boldsymbol{\theta}} \left[\frac{1}{N_a} \left(\sum_{i=1}^{N_a} z_i^{\text{top}} - \sum_{i=1}^{N_a} z_i^{\text{bot}} \right) \right], \tag{4.32}$$

where N_a is the number of atoms in each sulfur layer, z_i^{top} is the coordinate perpendicular to the MoS_2 plane (lying in the xy place) of atom i in the top layer, z_i^{bot} is similarly defined for the bottom layer. The expected thickness $\bar{t}$ was computed by performing an MD simulation at $T = 750$ K using large-scale atomic/molecular massively parallel simulator (LAMMPS) with the same setup used for calculations of the LTEC in section 2.2.3, however under NVT conditions. The simulation was repeated ten times with different initialization of atom velocities to compute the standard deviation that appears on the right-hand side of equation (4.31).

Equation equation (4.31) was evaluated for the parameters associated with the maximum and minimum diagonal FIM elements (see figure 4.2), $\sigma_{\text{Mo-S}}$ and $\lambda_{\text{Mo-S-Mo}}$, respectively. These are the parameters with respect to which observables will be most and least sensitive.

To evaluate equation (4.31), the mean thickness in equation (4.32) was computed for the parameter set $\boldsymbol{\theta}$ in tables 2.1 and 2.2, and also for $\boldsymbol{\theta} + \epsilon \boldsymbol{e}_i$ for the two studied parameters with $\epsilon = 0.01$.

For parameter $\sigma_{\text{Mo-S}}$, we find

$$\begin{aligned} \text{Left of equation (4.31)} &= |3.21887 - 3.19900| = 0.01987 \\ \text{Right of equation (4.31)} &= 0.00087 \times (0.01 \times 2.17517) \times \sqrt{7.02436 \times 10^6} \\ &= 0.05016, \end{aligned} \tag{4.33}$$

and for parameter $\lambda_{\text{Mo-S-Mo}}$, we find

$$\begin{aligned} \text{Left of equation (4.31)} &= |3.19874 - 3.19900| = 0.00026 \\ \text{Right of equation (4.31)} &= 0.00087 \times (0.01 \times 8.15952) \times \sqrt{1.13313 \times 10^2} \\ &= 0.00076. \end{aligned} \tag{4.34}$$

We see that as expected equation (4.31) is satisfied for both parameters, and at least

in this case, the bounds are rather tight. Thus equation (4.31) can be used to estimate the reliability of a model in making new predictions.

The Fisher information theory based uncertainty analysis shows that all the parameters are well identified for the FM-SW potential. The IP is most sensitive to parameters associated with two-body Mo–S interactions, and less sensitive to Mo–Mo and S–S interactions. The analysis also provides an analytical upper bound on the uncertainty in any phase average predictions that the IP makes due to small changes in its parameters. This is demonstrated by example for the mean thickness of a MoS_2 sheet at finite temperature. The change in mean thickness computed by MD is found to be tightly bound by the analytical expression. The FIM based uncertainty analysis described in this section is general and can be applied to IPs for other materials as long as the training set for the force-matching method is obtained from a dynamical trajectory sampling a distribution.

4.3 Dropout neural network potential

As discussed in section 3.3, in recent years, machine learning interatomic potentials (IPs) [178–182, 191] that train general-purpose functions (containing a large number of parameters that can be on the order of 10,000) against a large amount of data have been shown to possess errors around $4 \sim 5$ meV/atom, approaching the accuracy of some first-principles methods such as density functional theory (DFT). However, such IPs generally have very low *transferability*, i.e. the ability of an IP to make appropriate predictions outside its training set. The reason is that the general-purpose mathematical forms of such IPs bear no physical information, leading to extremely high parametric uncertainty. The low transferable characteristics of such IPs suggest that prerequisite actions must be taken to determine whether an IP is applicable to a new problem of interest; otherwise, one cannot trust the simulation results because the employed IP may introduce a huge amount of error.

We propose a dropout neural network interatomic potential (DNNIP) that can be used easily in practice to determine the transferability of an IP to new problems and to quantify the parametric uncertainty in simulation results.[2] We show that a DNNIP is

[2]The bias-variance trade-off [245] tells us that a more flexible model would have lower structural

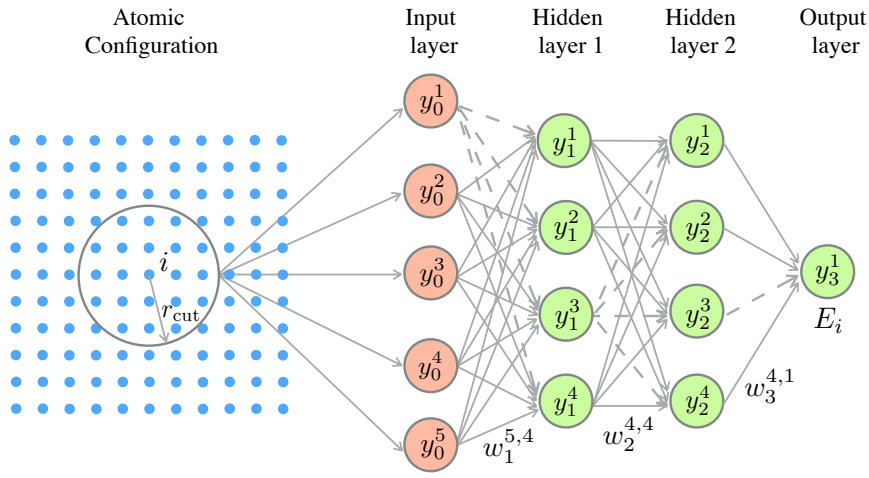


Figure 4.3: Schematic representation of an NN potential to compute the atomic energy E_i. The NN consists of an input layer, two hidden layers, and an output layer. The atomic neighborhoods information is transformed to the input for the NN, y_0^n ($n = 1, 2, \dots$), through a set of descriptors. Each arrow connecting two nodes between adjacent layers represents a weight. The fully-connected NN becomes a dropout NN when some connections are cut out (e.g. removing the dashed arrows). See text for explanation of the variables.

equivalent to a Bayesian neural network (NN) via variational inference and thus one can carry out uncertainty quantification using Bayesian statistics. We also provide a second interpretation of the DNNIP such that it can be used in a parameter ensemble approach to quantify uncertainty as in Ref. [237]. After introducing the DNNIP and the interpretations, we demonstrate the approaches to carry out transferability determination and uncertainty quantification using a model trained for condensed matter carbon.

4.3.1 Definition of model

The total interatomic potential energy of a configuration consisting of N_a atoms can be decomposed into the contributions of individual atoms

$$E = \sum_{i=1}^{N_\text{a}} E_i, \tag{4.35}$$

uncertainty but higher parametric uncertainty, and vice versa. The DNNIP is an extremely flexible model and therefore we only consider the parametric uncertainty.

where E_i is the atomic energy of atom i, represented by an NN as shown in figure 4.3. As discussed in section 3.3.1, the output of a fully-connected NN can be written as

$$E_i = \sigma[\sigma[\boldsymbol{y}_0\boldsymbol{W}_1 + \boldsymbol{b}_1]\boldsymbol{W}_2 + \boldsymbol{b}_2]\boldsymbol{W}_3 + \boldsymbol{b}_3, \tag{4.36}$$

in which the atomic environment descriptor to generate the inputs $\boldsymbol{y}_0$ for the NN used here is the same as that in section 3.3.1.

By cutting out some connections between layers (e.g. the dashed arrows of the NN shown in figure 4.3), we turn a fully-connected NN into a dropout NN [246,247]. Mathematically, equation (4.36) can be reformulated for a dropout NN as

$$E_i = \sigma[\sigma[\boldsymbol{y}_0(\boldsymbol{D}_1\boldsymbol{W}_1) + \boldsymbol{b}_1](\boldsymbol{D}_2\boldsymbol{W}_2) + \boldsymbol{b}_2](\boldsymbol{D}_3\boldsymbol{W}_3) + \boldsymbol{b}_3, \tag{4.37}$$

where the dropout matrix $\boldsymbol{D}_m$ ($m = 1, 2, 3$) is a diagonal matrix of binary integers of 0 or 1. Each diagonal element of $\boldsymbol{D}_m$ follows a Bernoulli distribution $\sim$ Bernoulli$(1-p)$ with a dropout ratio p. Redefining the weights $\widetilde{\boldsymbol{W}}_m \coloneqq \boldsymbol{D}_m\boldsymbol{W}_m$, we can view the dropout NN as a Bayesian model, because the parameters are stochastic now. Following the Bayesian approach, we denote $p(\boldsymbol{\theta})$ the *prior* distribution over the set of parameters $\boldsymbol{\theta} = \{\widetilde{\boldsymbol{W}}_1, \widetilde{\boldsymbol{W}}_2, \widetilde{\boldsymbol{W}}_3, \boldsymbol{b}_1, \boldsymbol{b}_2, \boldsymbol{b}_3\}$ and then look for the *posterior* distribution over the parameter space by invoking Bayes' theorem [248]:

$$p(\boldsymbol{\theta} \,|\, \boldsymbol{X}, \boldsymbol{Y}) \propto p(\boldsymbol{Y} \,|\, \boldsymbol{X}, \boldsymbol{\theta})p(\boldsymbol{\theta}), \tag{4.38}$$

where $p(\boldsymbol{Y} \,|\, \boldsymbol{X}, \boldsymbol{\theta})$ is the *likelihood* for the training set $(\boldsymbol{X}, \boldsymbol{Y})$ (see section 4.1.3 for a discussion of the Bayesian approach). With the posterior, we can obtain the *predictive* distribution for a new data point $(\boldsymbol{x}^*, \boldsymbol{y}^*)$,

$$p(\boldsymbol{y}^* \,|\, \boldsymbol{x}^*, \boldsymbol{X}, \boldsymbol{Y}) = \int p(\boldsymbol{y}^* \,|\, \boldsymbol{x}^*, \boldsymbol{\theta})p(\boldsymbol{\theta} \,|\, \boldsymbol{X}, \boldsymbol{Y})\, \mathrm{d}\boldsymbol{\theta}, \tag{4.39}$$

and then compute the predictive mean and variance for the new data point. The difficulty, however, is that the posterior for an NN with multiple hidden layers cannot be evaluated analytically [249]. To tackle this, we can take advantage of variational inference [250] that uses another distribution, $q(\boldsymbol{\theta})$, to approximate the posterior and replaces

$p(\boldsymbol{\theta} \mid \boldsymbol{X}, \boldsymbol{Y})$ by $q(\boldsymbol{\theta})$ in equation (4.39) to make predictions. Using this variational inference approach, Gal and Ghahramani [249,251] have recently shown that training an NN with the dropout technique approximates a Bayesian NN. Consequently, a dropout NN possesses all the properties of a probabilistic Bayesian model, from which uncertainty information can be extracted. In practice, for a new data point $(\boldsymbol{x}^*, \boldsymbol{y}^*)$, we only need to do multiple stochastic forward passes through the dropout NN (each with a different realization of the dropout matrices) to get multiple samples of the output $\boldsymbol{y}_1^*, \boldsymbol{y}_2^*, \ldots$ at the prediction stage. The average and variance of these samples can then be computed as the predictive mean and uncertainty, respectively.

The practical method to obtain the predictive mean and uncertainty can also be interpreted using the frequentist statistics. Applying dropout to a fully-connected NN amounts to sampling a "thinned" NN from it. The thinned NN consists of all the nodes that survived the dropout. An NN with a total number of n nodes can be considered as a collection of 2^n possible thinned NNs, and therefore training an NN with dropout can be seen as training a collection of 2^n thinned NNs with extensive weight sharing [247]. Consequently, using the dropout NN to make predictions can be seen as drawing samples from the ensemble of models.

To demonstrate how to determine the transferability of a DNNIP and quantify the uncertainty in atomistic simulations, we fit a DNNIP for carbon systems. The parameters in DNNIP, $\boldsymbol{\theta} = \{\boldsymbol{W}_1, \boldsymbol{b}_1, \boldsymbol{W}_2, \boldsymbol{b}_2, \ldots, \boldsymbol{W}_L, \boldsymbol{b}_L\}$ in which L is the number of NN layers (hidden layers plus output layer), are optimized by minimizing a loss function $\mathcal{L}(\boldsymbol{\theta})$ that quantifies the difference between the predictions of DNNIP and a training set. We construct a dataset of energies and forces for monolayer graphene, bilayer graphene, and graphite at various states. These include configurations with compressed and stretched cells and random perturbation of atoms for monolayer graphene, and configurations from *ab initio* molecular dynamics (AIMD) trajectories at temperatures 300, 900, and 1500 K for monolayer graphene, bilayer graphene, and graphite. The dataset consists of 4132 configurations, which is randomly divided into a training set of 3719 configurations (90%) and a test set of 413 configurations (10%). The data set is generated from DFT calculations using Vienna ab initio simulation package (VASP) in the same way as described in section 2.3.1.

The parameters are optimized by minimizing the loss function in equation (2.14),

i.e.

$$\mathcal{L}(\boldsymbol{\theta}) = \frac{1}{2}\sum_{i=1}^{N} w_i^{\mathrm{e}} \left[E_i(\boldsymbol{r}_i;\boldsymbol{\theta}) - \hat{E}_i\right]^2 + \frac{1}{2}\sum_{i=1}^{N} w_i^{\mathrm{f}} \|\boldsymbol{f}_i(\boldsymbol{r}_i;\boldsymbol{\theta}) - \hat{\boldsymbol{f}}_i\|^2, \tag{4.40}$$

where $N = 3719$ is the number of configurations in the training set, E_i and $\boldsymbol{f}_i(\boldsymbol{r}_i;\boldsymbol{\theta}) = -\left(\partial\mathcal{V}/\partial\boldsymbol{r}\right)|_{\boldsymbol{r}_i}$ are the potential energy and forces of configuration i, and $\hat{E}_i$ and $\hat{\boldsymbol{f}}_i$ are the corresponding reference energy and forces. We set the energy weight w_m^{e} to $1/(N_{\mathrm{a},i})^2$ and the force weight w_m^{f} to $1/(10(N_{\mathrm{a},i})^2)$, where $N_{\mathrm{a},i}$ is the number of atoms in configuration i.

Same as section 3.3.1, the symmetry functions [178, 187] are employed as the descriptor to transform the atomic environments to obtain the inputs $\boldsymbol{y}_0$ for the NN. Each feature (i.e. the descriptor values of different atoms obtained using the same descriptor) of the input is centered by subtracting the mean and then normalized by dividing the standard deviation before feeding to the NN. We select the hyperbolic tangent, $\tanh(x) = (e^x - e^{-x})/(e^x + e^{-x})$, as the nonlinear activation function, σ. We train and test DNNIP using KIM-based learning-integrated fitting framework (KLIFF) [69]. The optimization of the parameters $\boldsymbol{\theta}$ was carried out using the stochastic Adam optimizer [252] with a learning rate of 0.001 obtained by grid search. To accelerate the training process, we employ the mini-batch technique [253] with a mini-batch size of 100.

There are three hyperparameters that define the structure of the NN: (1) number of hidden layers, (2) number of nodes in each hidden layer, and (3) dropout ratio in the hidden layers and the output layer. In general, one can have different number of nodes and different dropout ratio for each layer, but for simplicity, we require that both of them to be the same across layers. The dropout ratio for the first layer (hidden layer 1) was set to 0, otherwise certain input information from the descriptors is totally lost, leading to deteriorated performance of DNNIP. In the following discussion, the dropout ratio refers to that in layers other than the first layer unless otherwise stated. The number of hidden layers and number of nodes in each layer were obtained by grid search over $[2, 3, 4, 5]$ and $[64, 128, 192, 256]$, respectively, at a dropout ratio of 0.1. We found that 3 hidden layers with 128 nodes in each hidden layer yields the smallest loss, equation (4.40), for the test set. Using 3 hidden layers with 128 nodes in each, we trained two more DNNIPs with dropout ratios 0.2 and 0.3.

Table 4.1: Energy and forces RMSEs for DNNIP using a dropout ratio of 0.1, 0.2 and 0.3.

Dropout ratio	Energy RMSE [meV/atom]	Forces RMSE [(meV/Å)/atom]
0.1 (training set)	4.2	43.4
0.1 (test set)	4.2	52.0
0.2 (training set)	4.5	46.6
0.2 (test set)	4.6	54.9
0.3 (training set)	4.9	56.4
0.3 (test set)	5.2	68.9

The energy root-mean-square error (RMSE) is defined as

$$\mathrm{RMSE}(E) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left(\bar{E}_i - \hat{E}_i\right)^2}, \tag{4.41}$$

and the force RMSE is defined as

$$\mathrm{RMSE}(\boldsymbol{f}) = \sqrt{\frac{1}{N}\sum_{i=1}^{N}\left\|\bar{\boldsymbol{f}}_i - \hat{\boldsymbol{f}}_i\right\|^2}, \tag{4.42}$$

where $\bar{E}_m$ and $\bar{\boldsymbol{f}}_m$ are the DNNIP mean energy and mean forces obtained by evaluating the DNNIP multiple times with different dropout matrices. The energy and forces RMSEs for the DNNIP with dropouts ratios 0.1, 0.2, and 0.3 are listed in table 4.1. We see that both the energy and forces RMSEs increase with dropout ratio. For a DNNIP with fixed number of hidden layers and number of nodes in each hidden layer, larger dropout ratio means fewer connections between layers, decreasing the capacity of the model and thus leads to increased RMSEs.

The RMSE is a measure of the accuracy of DNNIP to reproduce the training set and test set; to see the precision of the predictions made by DNNIP, we plot in figure 4.4 the uncertainty in atomic energy and force for both the training set and test set using DNNIP with dropout ratios 0.1, 0.2, and 0.3. The magnitude of the force on an atom

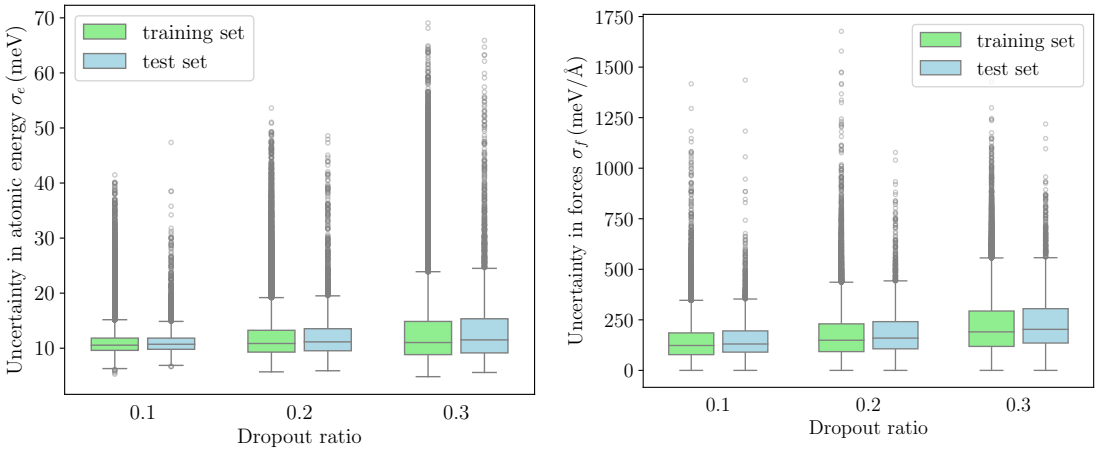


Figure 4.4: Uncertainty in (a) atomic energy and (b) forces on atom for the training set and test set using DNNIP with various dropout ratios. The lower and upper box edges represent the first and third quartiles of the data, respectively, the bar inside the box denotes the median, the ends of the whiskers represent the lowest datum and highest datum still within 1.5 interquartile range of the lower quartile and upper quartile, respectively, and the circles represent outliers.

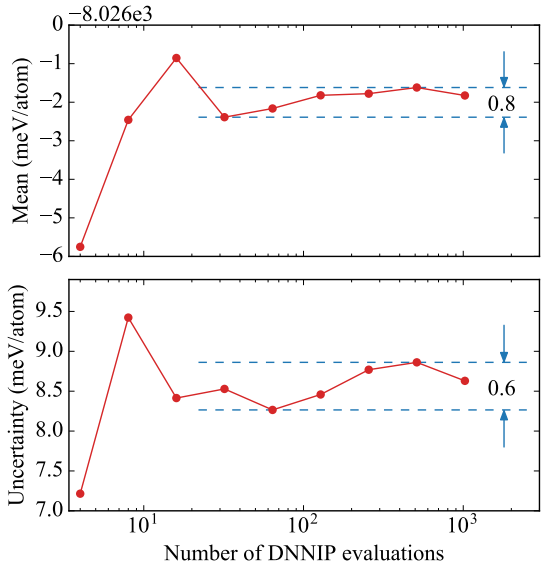


Figure 4.5: Predictive mean and uncertainty of the energy of a monolayer graphene as a function of the number of DNNIP evaluations.

is $f = \sqrt{f_x^2 + f_y^2 + f_z^2}$, the uncertainty in which can be estimated by[3]

$$\sigma_f = \frac{1}{f}\sqrt{f_x^2\sigma_{f_x}^2 + f_y^2\sigma_{f_y}^2 + f_z^2\sigma_{f_z}^2}, \tag{4.43}$$

where σ_{f_x}, σ_{f_y}, and σ_{f_z} are the uncertainty in f_x, f_y, and f_z, respectively. We see from figure 4.4 that at a given dropout ratio, the uncertainty for the test set is slightly larger than the training set based on the fact that the median, lower box edge, and upper box edge for the test set are higher than their counterparts for the training set. For both the training set and test set, we claim that the uncertainty in force increases with dropout ratio since the median, lower box edge, and upper box edge increase with dropout ratio. For the uncertainty in atomic energy, although the lower box edge slightly decreases with dropout ratio, we still see increased median and upper box edge as the dropout ratio becomes larger. All the results reported below are obtained using the DNNIP that has 3 hidden layers with 128 nodes in each hidden layer at a dropout ratio of 0.1.

The predictive mean and uncertainty are obtained by evaluating IP multiple times with different dropout matrices. An important question here is how many evaluations are needed for the predictive mean and uncertainty to converge. Figure 4.5 shows the predictive mean and uncertainty of the potential energy of a monolayer graphene as a function of the number of DNNIP evaluations. For both the mean and uncertainty, an evaluation number on the order of $\sim$100 leads the energy to converge into a band smaller than 1 meV/atom, which is on the accuracy level of our training set generated from DFT. All the predictive mean and uncertainty reported below are computed from 100 DNNIP evaluations unless otherwise stated.

4.3.2 Transferability determination

A direct method to determine the transferability of an IP is to measure the "distance" between the configurations characterizing the quantity of interest (QoI set) and the training set. If the distance is larger than some threshold value, we decide that the IP cannot be applied to study the new problem. The total potential energy is decomposed

[3] For a scalar function $f = f(x, y, z)$, the linearized uncertainty can be estimated using: $\sigma_f = \sqrt{(\partial f/\partial x)^2\sigma_x^2 + (\partial f/\partial y)^2\sigma_y^2 + (\partial f/\partial z)^2\sigma_z^2}$, where σ_x, σ_y, and σ_z are the uncertainty in x, y, and z, respectively [254].

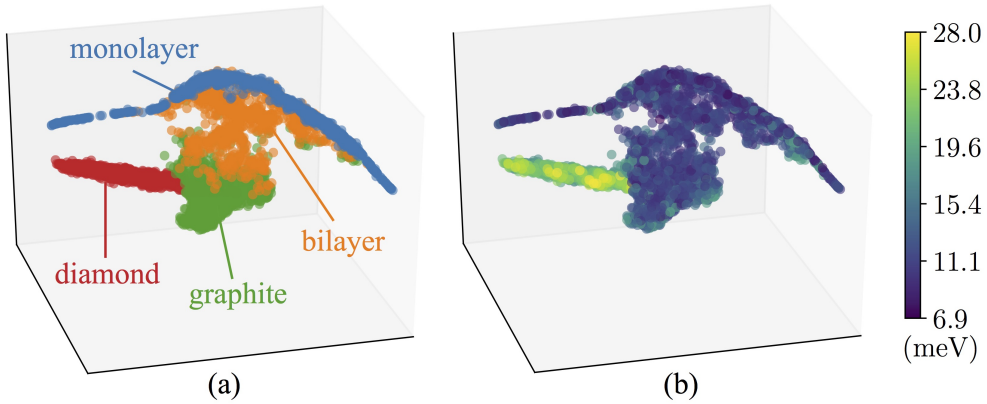


Figure 4.6: Representations of the carbon local atomic neighborhoods by UMAP. Each dot in the plot denotes one atom, and the representation is applied to the descriptor values (i.e. the input to the NN). The atoms are colored according to (a) the structure from which they come from and (b) the uncertainty in atomic energy. The training set consists of monolayer graphene (blue), bilayer graphene (orange), and graphite (green), but not diamond (red).

into the contributions of individual atoms (see equation (4.35)), which are obtained by mapping the descriptors that encode the local environments of individual atoms. Therefore, the descriptors could be a good candidate from which we measure the distance between the QoI set and the training set. The descriptors live in a high-dimensional space, so to visualize them, we apply the uniform manifold approximation and projection (UMAP) [255] dimensionality reduction technique to embed them to a three-dimensional (3D) space. The projected descriptors of the training set and a QoI set composed of diamond structures are plotted in figure 4.6. We see from panel (a) that the four carbon allotropes form clusters and separate from each other, although there is some overlap between them.[4] The clustering of similar atomic environments is the reason that many existing machine learning algorithms have successfully modeled the IP energy landscape of atomistic systems [256]. The QoI set (diamond (red)) is away from the training set (monolayer (blue), bilayer (orange), and graphite(green)), suggesting that the DNNIP parameterized against the training set is not suitable for diamond structures.

We plot the uncertainty in atomic energy (the energy of individual atoms) in panel

[4]It seems from the plot that diamond adjoins graphite, but actually they are noncontiguous in the out-of-paper direction.

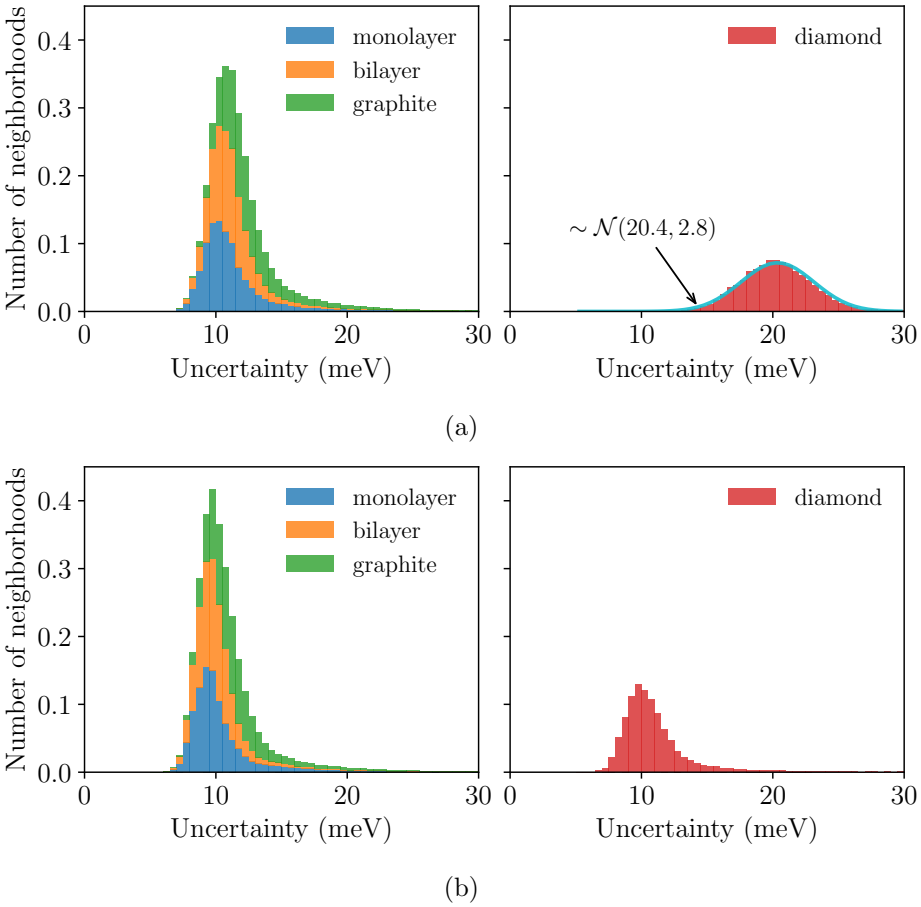


Figure 4.7: Histogram of the uncertainty in atomic energy. We randomly select 40,000 local atomic environments for each carbon allotrope, and the vertical axis is normalized by this value. In panel (a), the training set consists of monolayer graphene, bilayer graphene, and graphite, but not diamond, while in panel (b), the training set consists of all four types of carbon allotropes. The cyan curve in the right plot of panel (a) represents a normal distribution fitted to the histogram. The histograms on the left of both panel (a) and panel (b) are stacked.

(b) of figure 4.6. It is seen that the uncertainty in monolayer, bilayer, and graphite is low, whereas that in diamond is much higher.[5] The DNNIP is trained against monolayer, bilayer, and graphite without any representatives of the local atomic envrionments of diamond in the training set. Consequently, it has no idea of producing the correct predictions for diamond, thus resulting in large uncertainty in the prediction. The fact that the uncertainty is correlated to the distance between a QoI set and the training set is very useful. Instead of calculating the distance between a QoI set and the training set, we can compute the uncertainty in the QoI set and compare it with the uncertainty in the training set to determine whether the IP is suitable for the new problem of interest or not.

To show the uncertainty more quantitatively, we plot a histogram of the uncertainty in atomic energy for the training set and QoI set in figure 4.7a. The uncertainty in the training set is located around 10 meV, whereas that in the QoI set is much larger, around 20 meV. Again, this suggests that the IP is not applicable to diamond. We then add the QoI set to the training set, and refit the IP. The new histogram obtained using the refitted IP is plotted in figure 4.7b. The uncertainty in the original training set barely changes, but the uncertainty in diamond decreases significantly, to a level comparable to the other three carbon allotropes. This further confirms that the observed large uncertainty in diamond before it is added to the training set is simply because the IP is not trained against it. Also worth mentioning is the shape of the histogram. The histogram for diamond using the IP without diamond in the training set follows closely a normal distribution (figure 4.7a); however, it has a flatter tail on the larger uncertainty side than the smaller uncertainty side when diamond is added to the training set (figure 4.7b).

4.3.3 Uncertainty quantification

Now that we have a method to determine whether a DNNIP is transferable to a new problem or not, the next question to answer is how to quantify the uncertainty in a property obtained through atomistic simulations. We provide two ways to compute

[5] Arguably, a better way is to compare the relative uncertainty (the uncertainty normalized by the predictive mean), but here it is not a problem because the energy scale of all four carbon allotropes is around 8 eV/atom.

the uncertainty: a direct method and an indirect method. In the direct method, we compute the property multiple times, each with different but fixed dropout matrices in the IP, and then calculate the average and standard deviation of the outputs from these multiple runs as the predictive mean and uncertainty, respectively. This method applies to any property. But if a property has a "simple" relation with the IP energy and/or forces, the indirect method can be employed to propagate the uncertainty in the energy and/or forces obtained from the IP to the property.

As an example, we compute the potential part of the virial stress in a monolayer graphene using molecular dynamics (MD) simulations. The potential part of the virial stress (stress for short below) can be expressed as [17, 257]

$$s_{ij} = \frac{1}{VT} \sum_{t=1}^{T} \sum_{\alpha=1}^{N_\mathrm{a}} r_{i,t}^{\alpha} f_{j,t}^{\alpha}, \tag{4.44}$$

where $i, j \in \{1, 2, 3\}$ are Cartesian components, $r_{i,t}^{\alpha}$ is the ith component of the position of atom α at MD time step t, $f_{j,t}^{\alpha}$ is the jth component of the force on atom α at MD step t, N_a is the total number of atoms in the system, T is the total number of MD steps, and V is the volume of the system defined as the area of graphene multiplied with the van der Waals thickness, 3.4 Å in the present case.

In the indirect method, we rewrite equation (4.44) in a matrix form

$$\boldsymbol{s} = \frac{1}{VT} \boldsymbol{R} \boldsymbol{f}, \tag{4.45}$$

where the stress $\boldsymbol{s}$, in Voigt notation, is a column vector of 6 components, the coordinates $\boldsymbol{R}$ is a $6 \times 3N_\mathrm{a}T$ matrix, and the forces $\boldsymbol{f}$ is a column vector of length $3N_\mathrm{a}T$. See appendix F for a method to construct $\boldsymbol{R}$ and $\boldsymbol{f}$. Unlike the direct method where we run multiple MD simulations to compute the average and standard deviation of the outputs, only one MD trajectory is generated in the indirect method. At each MD step, we evaluate the forces multiple times with different dropout matrices and then update the positions of atoms by integrating the equations of motion using the average forces from the multiple evaluations. Therefore, we can assume that $\boldsymbol{R}$ is a coefficient matrix

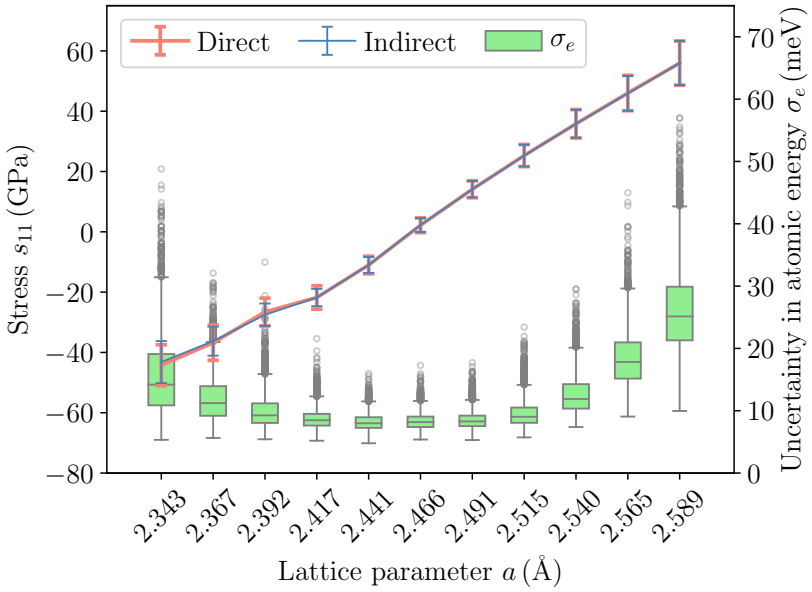


Figure 4.8: The potential part of the virial stress s_{11} and uncertainty in atomic energy σ_e in monolayer graphene at various lattice parameters. The left y axis is for the error bar plot of s_{11}, where we show the predictive mean and uncertainty obtained using both the direct and indirect methods. The right y axis is for the box and whisker plot of σ_e, where the bar inside the box denotes the median, the ends of the whiskers represent the lowest datum and highest datum still within 1.5 interquartile range of the lower quartile and upper quartile, respectively, and the circles represent outliers.

without any uncertainty.[6] Then the covariance of $\boldsymbol{s}$ can be estimated as [254]

$$\Sigma_{\boldsymbol{s}} = \frac{1}{V^2 T^2} \boldsymbol{R} \Sigma_{\boldsymbol{f}} \boldsymbol{R}^{\mathrm{T}}, \tag{4.46}$$

where $\Sigma_{\boldsymbol{s}}$ and $\Sigma_{\boldsymbol{f}}$ denote the covariance matrices of $\boldsymbol{s}$ and $\boldsymbol{f}$, respectively, and the square root of the 6 diagonal elements of $\Sigma_{\boldsymbol{s}}$ give the uncertainty in the stress. The force covariance matrix, $\Sigma_{\boldsymbol{f}}$, can be obtained from the multiple evaluations of the IP at each MD step.

Using both the direct and indirect methods, we computed the stress in a monolayer

[6] For a random variable x, the standard deviation of its sample mean $\bar{x} = \left(\sum_{i=1}^{n} x_i\right)/n$ is $\sigma_{\bar{x}} = \sigma/\sqrt{n}$, where σ is the standard deviation of x, and n is the sample size. We assume that the number of dropout evaluations is large enough such that the standard deviation of the mean of the forces is 0, thus introducing no uncertainty to the positions of atoms.

graphene at various in-plane lattice parameters. We construct a rectangular monolayer graphene consisting of 96 atoms using in-plane lattice parameter ranging from 2.343 to 2.589 Å. The zigzag and armchair edges of the graphene are aligned with the first and second Cartesian directions, respectively. Periodic boundary conditions are applied to both in-plane directions. The equations of motion were integrated using a velocity-Verlet algorithm with a time step of $\Delta t = 1$ fs. The system was thermalized at a constant temperature of $T = 300$ K under the canonical ensemble (NVT) using a Langevin thermostat. For both the direct and indirect methods, we ignore the first 10,000 unstable steps and then sample 1 out of 10 steps to obtain a total number of 1,000 steps to compute the stress.

The stress in the x direction s_{11} and its uncertainty $\sigma_{s_{11}}$ are plotted in figure 4.8. It is seen that the direct and indirect methods yield almost the same stress and uncertainty.[7] The stress s_{11} in the graphene has the smallest magnitude at the equilibrium lattice parameter $a = 2.466$ Å, and the magnitude increases as the graphene moves away from its equilibrium structure. The uncertainty in the stress $\sigma_{s_{11}}$ follows the same trend as the stress s_{11} (i.e. small near $a = 2.466$ Å and getting larger when moving away from it); however, the underlying mechanism is totally different. For s_{11}, this is purely due to the physical law that governs the material behavior: we get larger and larger tensile (compressive) stress when a material is constantly stretched (squeezed). But for $\sigma_{s_{11}}$, moving away from the equilibrium lattice parameter means making predictions for configurations deviating from the training set,[8] and thus we would expect higher uncertainty in the predictions. This is in agreement with the uncertainty in atomic energy, which measures the distance between these configurations and the training set as discussed in section 4.3.2. The uncertainty in atomic energy is presented as box and whisker plots in figure 4.8.

As a second example, we consider the phonon dispersions in a monolayer graphene, which provides a comprehensive view of the elastic vibrational behavior of IPs. Unlike the stress, there is not a simple linear relation between the phonon frequency and

[7] The slight difference originates from the fact that the stress and uncertainty are obtained from a single MD trajectory in the indirect method, whereas multiple distinct MD trajectories different from the one used in the indirect method have to be used in the direct method.

[8] For monolayer graphene, our training set only includes *ab initio* molecular dynamics trajectories using an initial lattice parameter of $a = 2.466$ Å and slightly stretched and compressed configurations using a lattice parameter $a \in [2.40, 2.52]$ Å.

the IP energy (or forces), so we compute the phonon dispersions using only the direct method. The phonon dispersions was calculated using the finite difference method as implemented in the phonopy package [209]. The phonon dispersions along some high-symmetry points in the first Brillouin zone are plotted in figure 4.9. We see that the predictive mean (dashed line) is in excellent agreement with DFT results (solid line). Specifically, it correctly captures the characteristics of the flexural acoustic (ZA) branch (e.g. the quadratic nature near the Γ point) that is associated with out-of-plane vibrations, which provides the dominant contribution to lattice thermal conductivity in graphene [218, 219]. The uncertainty in the phonon frequency is small for acoustic branches and becomes larger for optical branches as the absolute phonon frequency increases. Also plotted in figure 4.9 is the prediction obtained using the reactive empirical bond order (REBO) potential [83], which performs the best among a number of physics-based potentials such as the Tersoff [113], adaptive intermolecular reactive empirical bond order (AIREBO) [148], long-range carbon bond order potential (LCBOP) [84], and reactivate force field (ReaxFF) [201] models. See figure 3.5 for a comparison. The REBO potential performs comparably well as our DNNIP for the low-frequency acoustic branches, whereas its predictions for the high-frequency TO and LO branches deviate significantly from DFT results, much worse than DNNIP.

4.3.4 Precision and accuracy

Given a set of predictions obtained by varying an IP's parameters, *accuracy* refers to the difference between the average prediction and exact value (e.g. DFT results), and *precision* refers to the spread in the predictions. The predictions by an IP can only be trusted for properties for which it has high precision (i.e. low uncertainty), but this does not ensure that the predictions are accurate. To study the accuracy and precision in the predictions of DNNIP, we investigate the energy of a monolayer graphene, E, at different in-plane lattice parameter, a. At a given a, we do multiple DNNIP evaluations with different dropout matrices to obtain a set of predictions for E, the mean and standard deviation (our uncertainty) of which are plotted in figure 4.10. We see that the accuracy and precision are correlated: both of them decreases as the lattice parameter moves away from the equilibrium value $a = 2.466$ Å. This type of behavior has been observed elsewhere in empirical IPs [237] as well as exchange-correlation functionals

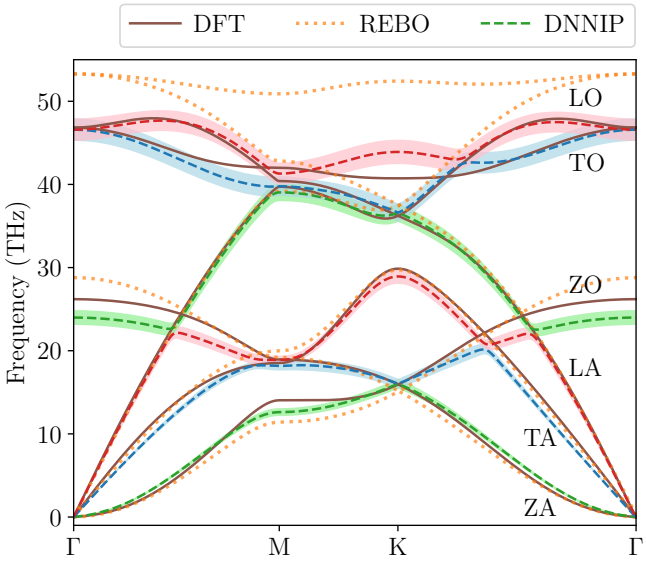


Figure 4.9: Phonon dispersions in monolayer graphene along some high-symmetry points in the first Brillouin zone. The dashed line and the band around it represent the predictive mean and uncertainty obtained from the DNNIP, respectively. The branches are colored according to the vibrational modes: green for flexural (Z), blue for transverse (T), and red for longitudinal (L). "A" and "O" stand for acoustic and optical, respectively.

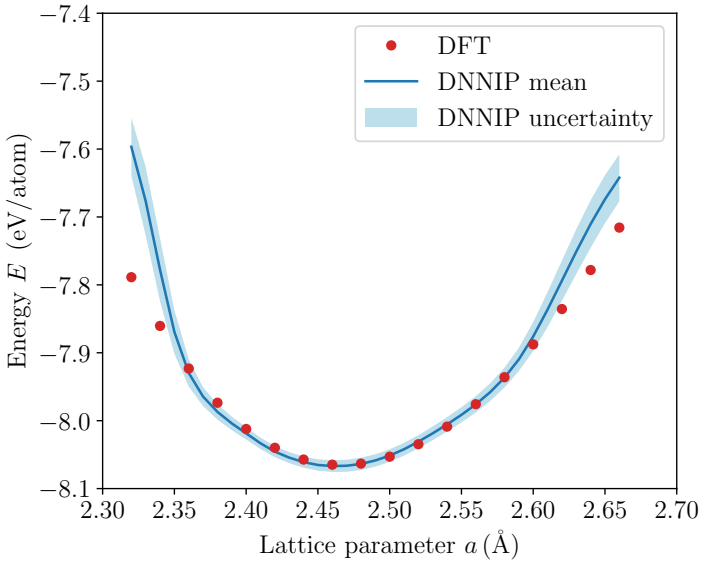


Figure 4.10: Energy of a monolayer graphene versus the in-plane lattice parameter obtained using the dropout NN potential and DFT.

used in DFT [258]. These models directly assume the form of the posterior where a temperature is introduced to formalize the weighting of different parameter sets. This leads to the empirical observation that the difference between the predictive mean and the exact value typically fall within the uncertainty band, and therefore the precision can be used as an estimate of the accuracy. Using a full Bayesian approach, our DNNIP does not seem to have this property, which can be seen clearly from figure 4.10, especially the predictions at small lattice parameters.

4.3.5 Summary

We propose a dropout neural network interatomic potential to model the interactions between atoms in materials. This IP can be used easily to determine the transferability to new problems of interest and to quantify the uncertainty in properties obtained from atomistic simulations, thus making the results trustable. In practice, we simply do multiple evaluations of the IP with difference dropout matrices to get multiple samples of the property of interest and then compute the average and standard deviation of these samples as the predictive mean and uncertainty, respectively. This approach is

justified by both the Bayesian statistics and the frequentist statistics.

Using a DNNIP for carbon systems as an example, we demonstrated how to determine the transferability and quantify the uncertainty and also investigated the relation between precision and accuracy. With transferability determination and uncertainty quantification, we believe that atomistic simulations with machine learning IPs could lead to better quantitative understanding of materials on a microscopic level.

Chapter 5

KLIFF: KIM-based Learning-Integrated Fitting Framework

The interatomic potentials (IPs) discussed in chapters 2 to 4 are all generated using the open-source KIM-based learning-integrated fitting framework (KLIFF) for interatomic potentials [69]. This package provides a unified interface to train both physics-based models and machine learning models built on various atomic environment descriptors. KLIFF is constructed using a modular approach and has a pure Python interface, making it easy to add new functionality. It integrates closely with the knowledgebase of interatomic models (KIM) ecosystem. A trained model can be readily deployed with the KIM application programming interface (API) [259] and then be used in major simulation codes such as LAMMPS [71], ASE [72], DL_POLY [74], and GULP [73] among others. The KLIFF package, together with its documentation, is publicly available at `https://kliff.readthedocs.io`.

5.1 Features and capabilities of KLIFF

KIM-based learning-integrated fitting framework (KLIFF) has a number of features and capabilities. Here, we introduce a few of them that distinguish KLIFF from other

interatomic potential (IP) fitting packages.

5.1.1 Integration with KIM

As indicated by the name, KLIFF is deeply integrated with the knowledgebase of interatomic models (KIM) ecosystem. KIM strives to make molecular simulations reliable, reproducible, and portable. KLIFF interacts with KIM in various aspects.

First, KLIFF supports the training of physics-based potentials archived in the open knowledgebase of interatomic models (OpenKIM) repository. An IP is called a model in the KIM nomenclature. A KIM portable model is an independent computer implementation of an IP that conforms to the KIM application programming interface (API) portable model interface (PMI) standard. It can be a stand-alone model or a model driver that reads in different parameter files to define different models.[1] All contents (including models) in the OpenKIM repository are archived subject to strict provenance control with digital object identifiers (DOIs) assigned. This makes it possible to access the exact IP used in a publication at any later date to reproduce the calculations—an ability lacking prior to OpenKIM archiving. A large number of IPs are archived in the OpenKIM repository with their correctness and code quality tested, such as the SW potential [79, 260], the Tersoff potential [261–264], the environment-dependent interatomic potential (EDIP) [265–268], and the embedded atom method (EAM) potential [81, 88, 269] to name a few. Therefore, users of KLIFF oftentimes can use these models directly without bothering to implement one from scratch, which is extremely error prone.

Second, IPs trained with KLIFF can be easily tested via KIM. KLIFF can automatically generate models that are compatible with the KIM API, thus allowing a trained IP to run against KIM verification checks (VCs) and KIM tests. KIM verification checks (VCs) are programs that explore the integrity of an IP implementation. They check for programming errors (e.g. memory leak [270]), failures to satisfy required behaviors (e.g. inversion [271] and permutation [272] symmetries), and general characteristics of the

[1] KIM also supports a second type of model called simulator model. While a portable model will work seamlessly with any simulation code that supports the KIM API/PMI standard, a simulator model only specifies how to setup and run a model that is implemented as an integrated part of a specific simulation code.

IP functional form (e.g. are the forces returned by the model consistent with those obtained through numerical differentiation of the energy [273]). As opposed to KIM VCs, KIM tests check the accuracy of an IP by computing a variety of physical properties of interest to researchers, such as stacking fault energy [274], elastic constants [275], and linear thermal expansion coefficient [276], etc. The information provided by KIM VCs and KIM tests can save a researcher a great deal of time by identifying limitations of an IP that can lead to subtle problems in simulations (e.g. poor convergence during energy minimization due to incorrect or discontinuous forces) and giving a general idea of how well an IP behaves on canonical physical properties.

Third, IPs trained with KLIFF can be deployed via KIM. Up to today, most IP development papers only report the functional form of the IPs and the associated parameters, without mentioning or providing any computer implementation. After obtaining a satisfied IP, IP developers either lack interest or expertise to make the IP implementation publicly available by transplanting it to a simulation code. Even if an IP developer is willing to go through this time-consuming and error-prone process to transplant the implementation, the IP will end up in only one or two simulation codes that the IP developer thinks important and worth the time. If users do not use the same simulation code, they need to implement the IP themselves in the simulation code they want to use or have to wait until someone else to implement it. This creates a significant barrier for the universal usability of the IP. The implementation is as important as (if not more important than) the mathematical form and the associated parameters of an IP, because in some cases the same parameter file can lead to different results when read in by different simulation codes. (See [65] For a discussion of this effect for tabulated EAM potentials.) As mentioned above, KLIFF can automatically create models that are compatible with the KIM API. The KIM API enables any IP conforming to this standard to work seamlessly with any KIM-compliant simulation code including large-scale atomic/molecular massively parallel simulator (LAMMPS) [71,137], ASE [72,277], DL_POLY [74,278], GULP [73,132] and ASAP [279] among others. The final production IP can be contributed to the OpenKIM repository for deployment. In such, the trained IP obtains all sorts of advantages of a KIM portable model as discussed above (e.g. provenance control), and other researchers can easily get access to it and then carry out simulations.

5.1.2 Sensitivity and uncertainty analysis

KLIFF provides a number of tools to analyze the quality of IPs and to train IPs with the ability to conduct uncertainty quantification in atomistic simulations. The Fisher information theory based approach discussed in sections 4.1.1 and 4.2 is available for computing the Fisher information matrix (FIM), from which the sensitivity in IP parameters and the uncertainty in observable upon perturbation of IP parameters can be obtained. KLIFF also supports the training of dropout neural network interatomic potential (DNNIP) discussed in section 4.3.

5.1.3 A wide range of support

As discussed in section 5.1.1, KLIFF integrates with KIM to fit physics-based potentials archived in the OpenKIM repository. Currently, the OpenKIM repository has 32 model drivers, including the widely-used SW [79, 260], Tersoff [261–264], EDIP [265–268], and EAM [81, 88, 269] potentials, etc. For machine learning IPs, the foremost ingredient is the descriptor that transforms atomic environments to vector representations to which machine learning regression models are then applied. Currently, KLIFF has support for the symmetry functions [178, 187], the bispectrum [179, 183], and the Coulomb matrix representations [180]. Other descriptors such as the many-body tensor [191] representation are under development.

For "simple" machine learning models such as linear regression and kernel ridge regression, KLIFF has its own implementations to conduct the training. Deep learning with neural network (NN) is such a highly growing area that new techniques are proposed every few months. It seems impractical (if not impossible) to implement these techniques in KLIFF timely and elegantly. Therefore, to avoid reinvent the wheels, KLIFF takes advantage of PyTorch [280] to build and train NN potentials. PyTorch is an open-source deep learning platform that provides a seamless path from research prototyping to production deployment. The NN model in KLIFF wraps PyTorch in such a way that the user interface has no difference from other models in KLIFF (more on the uniformity of KLIFF in section 5.1.4), but still retains the flexibility of PyTorch to create customizable NN structures and then train with state-of-the-art deep learning techniques.

As discussed in section 2.1.2, IP parameters are obtained by minimizing a loss function that quantifies the difference between IP predictions and the training set. The optimizer used to carry out the minimization directly determines the values of the parameters and thus the quality of the IP. It is impossible to tell which optimizer is "the" best or one optimizer is better than another, although some optimizers (e.g. the L-BFGS-B method [205]) work well for a wide range of problems in general. KLIFF takes advantage of the optimization algorithms in SciPy [281] and PyTorch [280] to train models when minimization of a loss function is necessary. The `scipy.optimize.minimize` module provides a large number of general minimization algorithms, while the `scipy.optimize.least_squares` module provides algorithms specific for nonlinear least-squares minimization problem with a loss function of the form in equation (2.14) (e.g, the Levenberg–Marquardt (Levenberg–Marquardt (LM)) method [105, 106]). The optimizers in PyTorch are targeted for training NN models, including the stochastic gradient descent (SGD) method [282, 283] and its variants such as the Adam method [252]. Besides, KLIFF also supports the geodesic LM algorithm [103, 284, 285], which has been shown to work extremely well for "sloppy" IPs whose predictions are insensitive to certain parameters or certain combinations of their parameters. (See [66] for a comparison of using the geodesic LM method and other methods for fitting the EDIP potential.)

5.1.4 Uniformity, modularity, and extensibility

KLIFF is designed to be as uniform, modular, and extensible as possible. It is implemented using an object-oriented programming (OOP) paradigm and provides a pure Python user interface. All the atomic environment descriptors, models, loss functions, etc. are subclassed from individual superclasses. A subclass only provides or modifies specific implementations of superclass methods when necessary without changing their names and arguments, guaranteeing a uniform interface among subclasses. As mentioned in section 5.1.3, KLIFF takes advantage of the optimization algorithms in SciPy [281] and PyTorch [280] to train models when minimization of a loss function is necessary. Although vanilla SciPy and PyTorch have different APIs to call these optimization algorithms, KLIFF provides a uniform interface that wraps SciPy and PyTorch optimization algorithms under the hood.

The task to train an IP can be divided into several subtasks. For example, to train a machine learning IP we typically: (1) select a descriptor to transform the atomic environments to vector representations; (2) build a regression model that takes the vector representations as input to calculate a set of predictions (energy, forces, stresses, etc.); (3) construct a loss function based on the predictions of the model and the corresponding reference data in the training set and then minimize the loss function to obtain the optimal parameter set; and (4) analyze the quality of the trained model. Using the OOP paradigm, each atomic environment descriptor, regression model, loss function, and analyzer are constructed as an individual class in such a way that any descriptor should work seamlessly with any regression model, any regression model should work with any loss function and so forth.

Extending KLIFF is made easy due to the way it is set up. New descriptor, regression model, loss function, optimization algorithm, analyzer, etc., can be added to work with existing modules in KLIFF. A new physics-based IP can be implemented either as a KIM model or a KLIFF model. The benefits of KIM models are discussed in section 5.1.1; however, this may be a bit burdensome in cases where fast research prototyping of new mathematical forms of an IP is preferred, because one needs to get familiar with the KIM API and then implement in low-level languages such as C, C++, or Fortran. To this end, KLIFF allows the creation of new models within its own framework using pure Python and it provides all sorts of utilities like neighbor list to aid the implementation of an IP. With these utilities, typically, the only thing left is to use Python to code up the mathematical form of the IP by replacing this (not implemented) part in a superclass for an IP model. The newly created model can then be utilized for training with any loss function and optimization algorithm that are supported by KLIFF. Extending other parts of KLIFF is similar to adding new models, but generally simpler and easier.

5.1.5 Data parallelization

Computationally expensive parts such as generating the neighbor list and transforming atomic environments into descriptor values are implemented in C++ to accelerate the calculation. Even with this, the computational requirements can still become quite demanding as the size of the training set increases. Fortunately, evaluation of the loss function (equation (2.14)) can be easily divided into sub-problems, and thus many

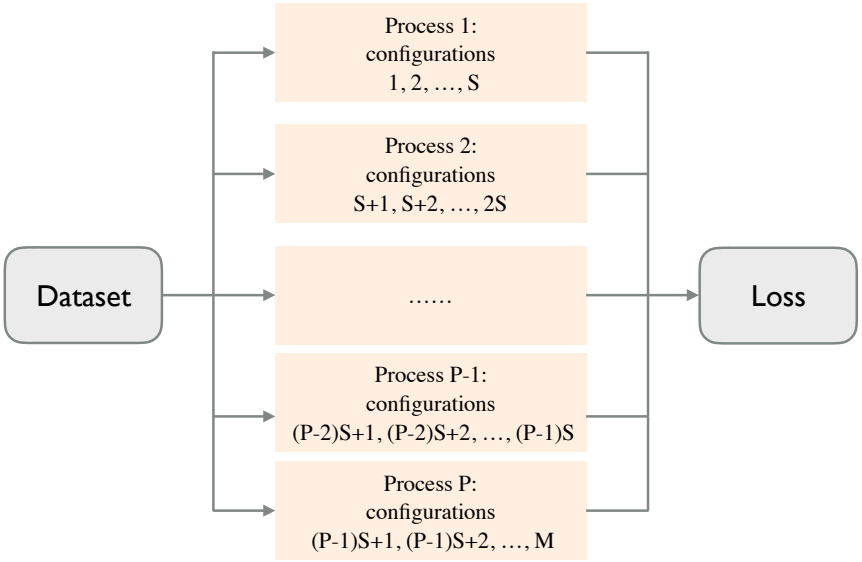


Figure 5.1: Data parallelization scheme used by KLIFF.

computer cores can work together to solve the problem with each core independently focusing on its own sub-problem. KLIFF adopts the simple parallelization over data scheme as illustrated in figure 5.1. Atomic configurations in the dataset are distributed to different processes. Each process may not get the same number of configurations as shown in figure 5.1 (when configurations do not have an equal number of atoms), but instead the total number of atoms of the configurations distributed to each process is approximately the same. This balances the load of each process since the computational cost of IPs scales linearly with the number of atoms. Each process computes the sub-loss according to equation (2.14) for the configurations assigned to it, and the total loss is then obtained as the sum of the sub-losses from all the processes. Besides the evaluation of the loss function, other tasks operating on the dataset such as generating the neighbor list and computing the descriptor values are also parallelized in this way.

KLIFF supports running in parallel mode on both shared-memory multicore desktop machines and high performance computing (HPC) clusters composed of multiple standalone machines connected by a network. Internally, we implement the data parallelization using both the OpenMP-style `multiprocessing` module of native Python and the MPI-style `mpi4py` [286] third-party package. These two are mutually exclusive.

The former can only work on desktop machines, while the latter works on both desktop machines and HPC clusters.

5.2 Implementation details: the KLIFF code

In this section, we discuss how the KIM-based learning-integrated fitting framework (KLIFF) package is set up. Detailed and update-to-date documentation of the instructions as well as the KLIFF package reference are available at: `https://kliff.readthedocs.io`.

KLIFF is built using Python with several computationally expensive parts internally implemented in C++. However, Python bindings of these C++ parts are used to create the user interface, so users are expected to interact with KLIFF purely through Python. It adopts a modular approach as discussed in section 5.1.4 and a flowchart representing the procedures to use some of the modules in KLIFF to train interatomic potentials (IPs) is schematically demonstrated in figure 5.2. The task to train an IP using KLIFF breaks down to the interaction and information exchange between these parts: `Dataset`, `Model`, `Calculator`, `Loss`, `Optimizer`, and `Analyzer`. In the following paragraphs, we briefly discuss each of these modules and how they interact with each other.

Dataset. A dataset is comprised of a set of atomic configurations, acting as training data to optimize IP parameters or examine data to test the quality of an IP. An atomic configuration should have three lattice vectors of the simulation cell, flags to indicate whether periodic boundary conditions (PBCs) are used, and atomic species and coordinates of all atoms in the configuration. These collectively define a configuration and are, typically, considered as the inputs to an IP model in terms of IP fitting. These information must be read in from disk. KLIFF adopts the extended XYZ file format, and each configuration is stored in a separate file. Internally, each atomic configuration is associated with a `Configuration` object and a `Dataset` is essentially a set of `Configuration` objects. The reference output values (e.g. energy, forces, and stress) associated with the inputs of an atomic configuration are also read in from the extended XYZ file and stored in the `Configuration` object. The reference outputs are typically obtained from more accurate first-principles calculations or experimental results.

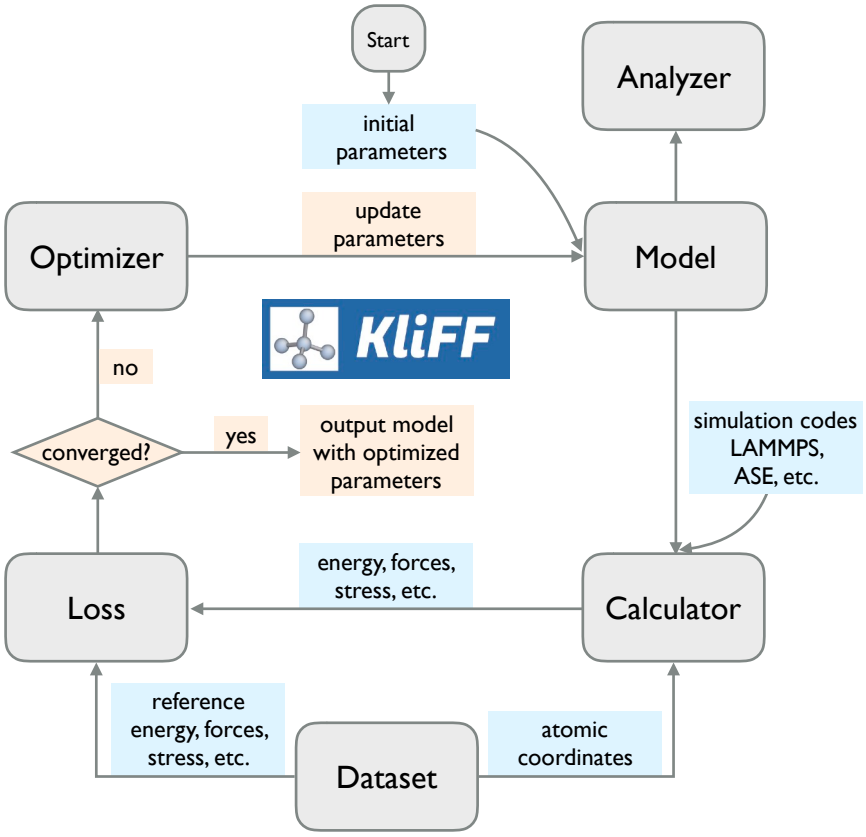


Figure 5.2: Flowchart of the procedures of using KLIFF to train an IP.

Model. To start fitting, a model (representing an IP) is first instantiated. Depending on the nature of the model, various operations can be applied to the model. For a physics-based model, KLIFF can provide information on what parameters are available for fitting, together with a description of each parameter and the data structure and data type of each parameter. Based on these information, we can then select a subset (proper or improper) of the parameters to fit and specify their initial values or simply use the default initial values. Lower and upper bounds on parameter values can also be supplied to restrict them in a range. KLIFF provides two ways to specific the initial guesses of parameters, either setting values by calling Python functions or reading in from a file. For a neural network (NN) model, the descriptor to transform atomic environments to the inputs for the NN needs to be provided at initialization. Then the NN can be constructed using whatever number of layers, number of nodes in each layer, and activation functions that the user thinks appropriate. Unlike physics-based models, KLIFF automatically initializes the parameters in the network. For example, the He initializer [287] is used to initialize the weights and biases in linear layers. We also make some other default choices and restrictions based on our experience. In such, we hope KLIFF makes it easier for users to create machine learning IPs without diving into the subtle side of machine learning.

Calculator. The created model is then attached to a calculator. Using the model, the calculator computes the predictions corresponding to the reference outputs for atomic configurations in the training set. A calculator can use any molecular simulation codes to compute the predictions, but for "simple" properties such as energy, forces, and stress, KLIFF has built-in implementations to quickly evaluate them.

Loss. The predictions computed by the calculator and the corresponding reference output values stored in the training set are then used to construct a loss function (e.g. equation (2.14)) that quantifies the difference between the model predictions and the references. KLIFF predefines a set of widely-used loss functions that are flexible enough to allow the assignment of a different weight for each configuration, making it possible to let "important" configurations weigh more during the optimization. One can even assign a different weight for each component of the prediction of the same configuration. If the available loss functions do not satisfy the need, a user-defined one can be supplied to KLIFF provided it follows the function prototype.

Optimizer. Based on the value of the loss function and/or other stopping criteria (e.g. the maximum allowed minimization step), KLIFF determines whether to terminate the training process. If not, the loss function is then used by the optimizer to adjust the IP parameters so as to minimize the value of the loss function. The optimizers supported by KLIFF can be broadly categorized into two classes: the *batch optimizers* and the *mini-batch optimizers*. The former (e.g. the L-BFGS-B and Levenberg–Marquardt (LM) methods) require the evaluation of the whole training set to carry out one minimization step, whereas the latter (e.g. the stochastic gradient descent (SGD) and Adam methods) typically only use a subset of the whole training set to carry out one minimization step. A batch optimizer guarantees monotonic decrease of the loss function as the minimization proceeds, and it typically yields smaller final loss function value compared with a mini-batch optimizer. Mini-batch optimizers are quite useful when a huge training set is used (usually the case for machine learning IPs), because evaluation of the whole set requires an enormous amount of time, impractical in many cases. For NN models that contain a large number of parameters, SGD-based optimizers can find a reasonable solution in the parameter space that minimizes the loss function to a certain level. KLIFF uses batch optimizer for physics-based IPs, which typically have no more than tens of parameters and thus only need a relatively small training set. For NN potentials, which normally involve tens of thousands of parameters or even more so that a large training set is needed, SGD-based mini-batch optimizers are recommended. But, of course, batch optimizers are still available to be used if one prefers.

The optimizer then update the model with the new set of parameters such that they will be used the next time the calculator computes the predictions. The optimization loop involving the `Dataset`, `Model`, `Calculator`, `Loss`, and `Optimizer` continues until the loss function is converged or other stopping criterion is hit. At exist, the fitted IP can be written out as a knowledgebase of interatomic models (KIM) model that conforms to the KIM application programming interface (API), which can then be run against KIM verification checks and KIM tests or be used with any KIM-compliant simulation codes as discussed in section 5.1.1. Also, the model can be attached to a `Analyzer` to carry out post-processing analysis, such as computing the Fisher information matrix (FIM) as discussed in section 4.2.

Command line tool. KLIFF also provides a command line tool named `kliff` that

facilitates the execution of many common tasks. For example, inquire a physics-based IP to obtain the available parameters that can be optimized and their associated metadata (e.g. data type and data size), and print a synopsis of the atomic configurations in the dataset or split a dataset into multiple subsets.

5.3 Demonstration

In this section we give an example to demonstrate how to use KIM-based learning-integrated fitting framework (KLIFF) to train a Stillinger–Weber (SW) potential for silicon. Note, the codes shown in this section is compatible with KLIFF v0.1.1, which may not work for later versions of KLIFF. But the up-to-date example is available at the KLIFF documentation: `https://kliff.readthedocs.io`.

Here, we train a SW potential for silicon that is archived in the open knowledgebase of interatomic models (OpenKIM) repository. Before getting started to train the model, let's first install it:

```
$ kim-api-collections-management install user \
    SW_StillingerWeber_1985_Si__MO_405512056662_005
```

We are going to fit the model to a training set of energies and forces from compressed and stretched diamond silicon structures as well as configurations drawn from molecular dynamics (MD) trajectories at different temperatures. The training set is stored in the extended XYZ format. A tarball of the training set can be downloaded from `https://raw.githubusercontent.com/mjwen/kliff/master/examples/Si_training_set.tar.gz`. To extract the training set, do

```
$ tar xzf Si_training_set.tar.gz
```

Note that the `Si_training_set` is just a toy data set for the purpose to demonstrate how to use KLIFF to train models, so by no means should it be suitable for the training of interatomic potentials (IPs) for real simulations.

Model

We first instantiate a knowledgebase of interatomic models (KIM) model for the SW potential and print out all its available parameters that can be optimized (we call this *model parameters*):

```
from kliff.models import KIM

model = KIM(model_name="SW_StillingerWeber_1985_Si__MO_405512056662_005")
model.echo_model_params()
```

The output generated by the last line reads:

```
#=============================================================================
# Available parameters to optimize.
#
# Model: SW_StillingerWeber_1985_Si__MO_405512056662_005
#=============================================================================

name: A
value: [15.28484792]
size: 1
dtype: Double
description: Multiplicative factors on the two-body energy function as a
    whole for each binary species combination. In terms of the original SW
    parameters, each quantity is equal to A*epsilon for the corresponding
    species combination. This array corresponds to a lower-triangular matrix
     (of size N=1) in row-major storage. Ordering is according to
    SpeciesCode values. For example, to find the parameter related to
    SpeciesCode 'i' and SpeciesCode 'j' (i >= j), use (zero-based) index = (
    j*N + i - (j*j + j)/2).

name: B
value: [0.60222456]
size: 1
dtype: Double
```

```
description: Multiplicative factors on the repulsive term in the two-body
    energy function for each binary species combination. This array
    corresponds to a lower-triangular matrix (of size N=1) in row-major
    storage. Ordering is according to SpeciesCode values. For example, to
    find the parameter related to SpeciesCode 'i' and SpeciesCode 'j' (i >=
    j), use (zero-based) index = (j*N + i - (j*j + j)/2).


...
```

which shows the name, value, size, data type and a description of each parameter. In fact, there are other model parameters in the SW potential available for optimization (e.g. `p`, `sigma`, `gamma`, `lambda`, etc.), but we omit them here for the sake of space.

Now that we know what model parameters are available for fitting, we optimize a subset of them to reproduce the training set.

```
model.set_fitting_params(
    gamma=[[1.5]],
    B=[["default"]],
    sigma=[[2.0951, "fix"]],
    A=[[5, 1, 20]])

model.echo_fitting_params()
```

Here, we tell KLIFF to fit four parameters `gamma`, `B`, `sigma`, and `A` of the SW potential. The information for each fitting parameter should be provided as a list of list, where the size of the outer list should be equal to the size of the parameter given by `model.echo_model_params()`. For each inner list, we can provide either one, two, or three items.

- One item. We can use a numerical value to provide an initial guess of the parameter. For example, `gamma`. Alternatively, the string `default` can be provided to use the default value in the model. For example, `B`.
- Two items. The first item should be a numerical value and the second item should be the string `fix`, which tells KLIFF to use the first item as the value of the parameter but do not optimize it. For example, `sigma`.

- Three items. The first item can be a numerical value or the string `default`, having the same meanings as the one item case. The second and third items are the lower and upper bounds for the parameters, respectively. A bound can be provided as either a numerical value or `None`, with the latter indicating no bound is applied. For example, `A`.

The call of `model.echo_fitting_params()` prints to stdout the fitting parameters that we require KLIFF to optimize:

```
#=========================================================================
# Model parameters that are optimized.
#=========================================================================

A 1
  5.0000000000000000e+00 1.0000000000000000e+00 2.0000000000000000e+01

B 1
  6.0222455840000000e-01

sigma 1
  2.0951000000000000e+00 fix

gamma 1
  1.5000000000000000e+00
```

where the number `1` after the name of each parameter indicates the size of the parameter. Parameters that are not included as fitting parameters are fixed to their default values in the model during the optimization.

Training set

KLIFF has a `Dataset` class to deal with the training data (and test data). For the silicon training set, we can read and process the extended XYZ files by:

```
from kliff.dataset import Dataset

dataset_name = "Si_training_set"
```

```
tset = Dataset()
tset.read(dataset_name)
configs = tset.get_configs()
```

The `configs` in the last line is a list of `Configuration`. Each `Configuration` is an internal representation of a processed extended XYZ file, consisting of the species, coordinates, energy, forces, and other related information of a system of atoms.

Calculator

Calculator is the central agent that exchanges information and orchestrate the operation of the fitting process. It computes a set of predictions using the model and provides this information to the loss function (discussed below) to compute the loss value. It also grabs the new parameters from the optimizer and update the parameters in the model so that the up-to-date parameters are used the next time the model is evaluated. The calculator can be created by:

```
from kliff.calculator import Calculator

calc = Calculator(model)
calc.create(configs)
```

where `calc.create(configs)` does necessary initializations for each configuration in the training set such as creating the neighbor list.

Loss function

KLIFF uses a loss function to quantify the difference between model predictions and the corresponding reference data in the training set and then uses optimization algorithms to reduce the loss as much as possible. For physics-based IPs, any algorithm listed on scipy.optimize.minimize and scipy.optimize.least_squares can be used. In the following code snippet, we create a loss function and then use the `L-BFGS-B` algorithm to minimize the loss. The minimization will run with 1 process and a max number of 100 iterations are allowed.

```
from kliff.loss import Loss
```

```
steps = 100
loss = Loss(calc, nprocs=1)
loss.minimize(method="L-BFGS-B", options={"disp": True, "maxiter": steps})
```

The output reads:

```
RUNNING THE L-BFGS-B CODE

          * * *

Machine precision = 2.220D-16
 N =            3     M =           10

At X0         0 variables are exactly at the bounds

At iterate   0    f=  1.65618D+07   |proj g|= 1.63611D+07

At iterate   1    f=  4.50459D+06   |proj g|= 7.90884D+06
.
.
.
At iterate  25    f=  3.25435D+03   |proj g|= 1.16308D+02

At iterate  26    f=  3.25435D+03   |proj g|= 3.06113D+00

At iterate  27    f=  3.25435D+03   |proj g|= 6.61066D-01

          * * *

Tit   = total number of iterations
Tnf   = total number of function evaluations
Tnint = total number of segments explored during Cauchy searches
Skip  = number of BFGS updates skipped
Nact  = number of active bounds at final generalized Cauchy point
Projg = norm of the final projected gradient
F     = final function value
```

```
         * * *

   N    Tit     Tnf  Tnint  Skip  Nact     Projg        F
    3     27     36     28     0     0   6.611D-01  3.254D+03
  F =   3254.3480974009767

CONVERGENCE: REL_REDUCTION_OF_F_<=_FACTR*EPSMCH

 Cauchy               time 0.000E+00 seconds.
 Subspace minimization time 0.000E+00 seconds.
 Line search          time 0.000E+00 seconds.
```

As seen, the minimization converges after running for 27 steps.

Save trained model

After training, we'd better save the model to disk so that it can be loaded later for retraining, evaluation, or other analysis. If we are satisfied with the fitted model, we can also write it as a KIM model; then it can be used with simulation codes that conform to the KIM application programming interface (API).

```
model.echo_fitting_params()
model.save("kliff_model.pkl")
model.write_kim_model()
```

The first line of the above code generates:

```
#=========================================================================
# Model parameters that are optimized.
#=========================================================================

A 1
  1.5008554501462323e+01 1.0000000000000000e+00 2.0000000000000000e+01

B 1
  5.9537800948866415e-01
```

```
sigma 1
  2.0951000000000000e+00 fix

gamma 1
  2.4122637121188939e+00
```

A comparison with the original parameters before carrying out the minimization shows that we recover the original parameters quite reasonably. The second line saves the fitted model to disk with a file name of `kliff_model.pkl`, and the third line writes out a KIM model named `SW_StillingerWeber_1985_Si__MO_405512056662_005_kliff_trained`.

So far, we have successfully trained the physics-based SW potential for silicon. For machine learning IPs, the procedures would be largely the same. Again, refer to the KLIFF documentation for other examples.

Chapter 6

Conclusions and Future Work

Atomistic simulation with empirical interatomic potentials (IPs) is a useful computational tool to investigate materials on a microscopic level. IPs that describe the interactions between atoms and thus produce the forces governing atomic motion and deformation are arguably the most important element that determines the quality of atomistic simulations.

During my doctoral studies, I have developed both physics-based and machine learning IPs for two-dimensional (2D) materials and heterostructures and applied these IPs to study their mechanical and thermal properties. In particular, I have created a Stillinger–Weber (SW) potential for monolayer MoS_2, a registry-dependent potential for the interlayer interactions in graphene, as well as a neural network (NN) potential for multilayer graphene structures. These IPs are either built based on existing models to improve/correct their behaviors or built from scratch to capture the physics deemed to be important for 2D materials and heterostructures.

Atomistic simulation with IPs are viewed as a tool limited to provide only qualitative insight. A key reason is that in such simulations there are many sources of uncertainty that are difficult to quantify, thus failing to give confidence interval on simulation results. A novel contribution of my work is the development and application of techniques to quantify the uncertainty in IPs themselves and simulation results obtained using IPs. For physics-based IPs, I demonstrate how to analyze the parameter sensitivity of an IP and the uncertainty in its predictions using the Fisher information theory extended to path space. For machine learning NN potentials, I show how to apply the

dropout technique to train a NN and then obtain the predictive mean and variance (the uncertainty) as ensemble averages. Besides, I have discovered the correlation between the uncertainty in atomic energies and the distance between the training set and the configurations characterizing a new problem of interesting, and proposed a practical method to determine the transferability of NN potentials.

I have also developed an open-source KIM-based learning-integrated fitting framework (KLIFF) to train both physics-based and machine learning IPs. KLIFF integrates closely with the knowledgebase of interatomic models (KIM) echosystem to either use the physics-based IPs archived in the open knowledgebase of interatomic models (OpenKIM) repository or deploy the trained model via the KIM application programming interface (API). It supports a variety of atomic environment descriptors and machine learning regression methods, and specifically PyTorch is used internally to provide state-of-the-art deep learning techniques for NN models. In addition, KLIFF provides a number of tools to assess the quality of IPs such as computing the Fisher information matrix (FIM). I hope other researchers will find KLIFF useful when developing their own IPs.

There are many open directions for future research:

Potentials for other 2D materials and heterostructures. In this thesis, I present NN potentials for multilayer graphene structures. In the future, I intend to build NN potentials and perform large scale simulations for other 2D materials and heterostructures. One interesting problem is to investigate how heterostructures behave when different types of 2D materials are stacked on top of each other and whether we can create new physics by stacking them in different manners.

Active learning. The IPs in this thesis are fit to pre-determined training sets that are believed to be important for the problems of interest. This is, indeed, a nontrivial task, especially for machine learning models, because we may not know a priori the configurations that carry the information needed by a problem of interest. With the uncertainty quantification capability of dropout NN, we are interested in applying active learning to generate the training set automatically. First, we train an NN model to a preliminary training set. Next, the trained NN model is applied to make predictions for the problem of interest, and at the same time we measure the associated uncertainty in the predictions. Then, we add back to the training set the configurations with large

uncertainty and retrain the model against the updated training set. We do this *training–uncertainty quantification–enriching the training set* process until the uncertainty in the predictions is below a threshold value. With this active learning technique, training an NN potential can be largely automated, if not all.

Faster atomic environment descriptors. Although significantly faster than first-principles approaches, machine learning IPs are more computationally expensive than physics-based IPs. For example, our NN potential that uses the symmetry functions as the atomic environment descriptor is about 100 times slower than a Tersoff [113] potential. Actually, the largest portion of time of a machine learning IP is spent on evaluating the atomic environment descriptor. We have initialized an effort to design new atomic environment descriptors that not only satisfy all the requirements discussed in section 3.1, but are also computationally far less expensive. Preliminary results show that Gabor transformation [288] of atomic density function seems a promising method.

Extending KLIFF. For now, KLIFF only allows the use of energy, forces, and stress as the training target. We are planning to extend the framework such that any materials property can be employed in the training, for example, equilibrium lattice parameters, elastic moduli, and phonon dispersion curves to name a few. This is extremely useful for physics-based IPs where the number of parameters is not too large such that we can afford to compute these properties during the training process. We also hope to support more physics-based IPs and machine learning regression methods.

Bibliography

[1] N. R. Council. *Materials and Man's Needs: Materials Science and Engineering – Volume I, The History, Scope, and Nature of Materials Science and Engineering*. The National Academies Press, Washington, DC, 1975.

[2] L. A. Dobrzański. Significance of materials science for the future development of societies. *J. Mater. Process. Technol.*, 175(1-3):133–148, 2006.

[3] N. A. Spaldin. Fundamental materials research and the course of human civilization. *arXiv preprint arXiv:1708.01325*, 2017.

[4] D. L. Schodek, P. Ferreira, and M. F. Ashby. *Nanomaterials, nanotechnologies and design: an introduction for engineers and architects*. Butterworth-Heinemann, 2009.

[5] J. Ramsden. *Nanotechnology: An Introduction*. Micro and Nano Technologies. Elsevier Science, 2011.

[6] W. H. Hunt. Nanomaterials: Nomenclature, novelty, and necessity. *JOM*, 56(10):13–18, 2004.

[7] G. Binnig and H. Rohrer. Scanning tunneling microscopy. *Surf. Sci.*, 126(1-3):236–244, 1983.

[8] G. Binnig, C. F. Quate, and C. Gerber. Atomic force microscope. *Phys. Rev. Lett.*, 56(9):930, 1986.

[9] F. Cailliez and P. Pernot. Statistical approaches to forcefield calibration and prediction uncertainty in molecular simulation. *J. Chem. Phys.*, 134(5):054124, 2011.

[10] P. Angelikopoulos, C. Papadimitriou, and P. Koumoutsakos. Bayesian uncertainty quantification and propagation in molecular dynamics simulations: a high performance computing framework. *J. Chem. Phys.*, 137(14):144103, 2012.

[11] R. Z. Khaliullin, H. Eshet, T. D. Kühne, J. Behler, and M. Parrinello. Nucleation mechanism for the direct graphite-to-diamond phase transition. *Nat. Mater.*, 10(9):693, 2011.

[12] K. Chenoweth, A. C. Van Duin, and W. A. Goddard. Reaxff reactive force field for molecular dynamics simulations of hydrocarbon oxidation. *J. Phys. Chem. A*, 112(5):1040–1053, 2008.

[13] S. Piana, K. Lindorff-Larsen, and D. E. Shaw. Protein folding kinetics and thermodynamics from atomistic simulation. *Proc. Natl. Acad. Sci.*, 109(44):17845–17850, 2012.

[14] R. A. Messerly, T. A. Knotts, and W. V. Wilding. Uncertainty quantification and propagation of errors of the lennard-jones 12-6 parameters for n-alkanes. *J. Chem. Phys.*, 146(19):194110, 2017.

[15] K. S. Novoselov, A. K. Geim, S. V. Morozov, D. Jiang, Y. Zhang, S. V. Dubonos, I. V. Grigorieva, and A. A. Firsov. Electric field effect in atomically thin carbon films. *Science*, 306(5696):666–669, 2004.

[16] D. Griffiths. *Introduction to Quantum Mechanics*. Pearson international edition. Pearson Prentice Hall, 2005.

[17] E. B. Tadmor and R. E. Miller. *Modeling materials: continuum, atomistic and multiscale techniques*. Cambridge University Press, 2011.

[18] M. Ishigami, J. Chen, W. Cullen, M. Fuhrer, and E. Williams. Atomic structure of graphene on sio2. *Nano Lett.*, 7(6):1643–1648, 2007.

[19] E. Fradkin. Critical behavior of disordered degenerate semiconductors. ii. spectrum and transport properties in mean-field theory. *Phys. Rev. B*, 33(5):3263, 1986.

[20] The Nobel Prize in physics 2010. https://www.nobelprize.org/prizes/physics/2010/summary/. Retrieved: 2019-03-08.

[21] Y. Hernandez, V. Nicolosi, M. Lotya, F. M. Blighe, Z. Sun, S. De, I. McGovern, B. Holland, M. Byrne, Y. K. Gun'Ko, et al. High-yield production of graphene by liquid-phase exfoliation of graphite. *Nat. Nanotechnol.*, 3(9):563, 2008.

[22] K. V. Emtsev, A. Bostwick, K. Horn, J. Jobst, G. L. Kellogg, L. Ley, J. L. McChesney, T. Ohta, S. A. Reshanov, J. Röhrl, et al. Towards wafer-size graphene layers by atmospheric pressure graphitization of silicon carbide. *Nat. Mater.*, 8(3):203, 2009.

[23] X. Li, W. Cai, J. An, S. Kim, J. Nah, D. Yang, R. Piner, A. Velamakanni, I. Jung, E. Tutuc, et al. Large-area synthesis of high-quality and uniform graphene films on copper foils. *Science*, 324(5932):1312–1314, 2009.

[24] W. Kern and G. L. Schnable. Low-pressure chemical vapor deposition for very large-scale integration processing—a review. *IEEE Trans. Electron Devices*, 26(4):647–657, 1979.

[25] A. H. C. Neto, F. Guinea, N. M. R. Peres, K. S. Novoselov, and A. K. Geim. The electronic properties of graphene. *Rev. Mod. Phys.*, 81(1):109–162, 2009.

[26] C. Berger, Z. Song, X. Li, X. Wu, N. Brown, C. Naud, D. Mayou, T. Li, J. Hass, A. N. Marchenkov, et al. Electronic confinement and coherence in patterned epitaxial graphene. *Science*, 312(5777):1191–1196, 2006.

[27] Y. Zhang, Y.-W. Tan, H. L. Stormer, and P. Kim. Experimental observation of the quantum hall effect and berry's phase in graphene. *Nature*, 438(7065):201–204, nov 2005.

[28] S.-E. Zhu, S. Yuan, and G. Janssen. Optical transmittance of multilayer graphene. *EPL (Europhysics Letters)*, 108(1):17007, 2014.

[29] R. R. Nair, P. Blake, A. N. Grigorenko, K. S. Novoselov, T. J. Booth, T. Stauber, N. M. Peres, and A. K. Geim. Fine structure constant defines visual transparency of graphene. *Science*, 320(5881):1308–1308, 2008.

[30] D. E. Sheehy and J. Schmalian. Optical transparency of graphene as determined by the fine-structure constant. *Phys. Rev. B*, 80(19):193411, 2009.

[31] W. Cai, A. L. Moore, Y. Zhu, X. Li, S. Chen, L. Shi, and R. S. Ruoff. Thermal transport in suspended and supported monolayer graphene grown by chemical vapor deposition. *Nano Lett.*, 10(5):1645–1651, 2010.

[32] C. Faugeras, B. Faugeras, M. Orlita, M. Potemski, R. R. Nair, and A. Geim. Thermal conductivity of graphene in corbino membrane geometry. *ACS Nano*, 4(4):1889–1892, 2010.

[33] X. Xu, L. F. Pereira, Y. Wang, J. Wu, K. Zhang, X. Zhao, S. Bae, C. T. Bui, R. Xie, J. T. Thong, et al. Length-dependent thermal conductivity in suspended single-layer graphene. *Nat. Commun.*, 5:3689, 2014.

[34] J.-U. Lee, D. Yoon, H. Kim, S. W. Lee, and H. Cheong. Thermal conductivity of suspended pristine graphene measured by raman spectroscopy. *Phys. Rev. B*, 83(8):081419, 2011.

[35] A. Yousefzadi Nobakht and S. Shin. Anisotropic control of thermal transport in graphene/si heterostructures. *J. Appl. Phys.*, 120(22):225111, 2016.

[36] J. Los, K. Zakharchenko, M. Katsnelson, and A. Fasolino. Melting temperature of graphene. *Phys. Rev. B*, 91(4):045415, 2015.

[37] List of thermal conductivities. https://en.wikipedia.org/wiki/List_of_thermal_conductivities. Retrieved: 2019-03-08.

[38] C. Lee, X. Wei, J. W. Kysar, and J. Hone. Measurement of the elastic properties and intrinsic strength of monolayer graphene. *Science*, 321(5887):385–388, 2008.

[39] H. C. Schulitz, W. Sobek, and K. J. Habermann. *Steel Contruction Manual*. Walter de Gruyter, 2012.

[40] A. K. Geim and I. V. Grigorieva. Van der waals heterostructures. *Nature*, 499(7459):419–425, 2013.

[41] A. Jain, S. P. Ong, G. Hautier, W. Chen, W. D. Richards, S. Dacek, S. Cholia, D. Gunter, D. Skinner, G. Ceder, and K. a. Persson. The Materials Project: A materials genome approach to accelerating materials innovation. *APL Mater.*, 1(1):011002, 2013.

[42] M. Ashton, J. Paul, S. B. Sinnott, and R. G. Hennig. Topology-scaling identification of layered solids and stable exfoliated 2d materials. *Phys. Rev. Lett.*, 118(10):106101, 2017.

[43] K. S. Novoselov, A. Mishchenko, A. Carvalho, and A. H. C. Neto. 2d materials and van der waals heterostructures. *Science*, 353(6298):aac9439, 2016.

[44] Y. Zhang, T.-T. Tang, C. Girit, Z. Hao, M. C. Martin, A. Zettl, M. F. Crommie, Y. R. Shen, and F. Wang. Direct observation of a widely tunable bandgap in bilayer graphene. *Nature*, 459(7248):820–823, 2009.

[45] Y. Cao, V. Fatemi, S. Fang, K. Watanabe, T. Taniguchi, E. Kaxiras, and P. Jarillo-Herrero. Unconventional superconductivity in magic-angle graphene superlattices. *Nature*, 556(7699):43–50, 2018.

[46] L. Ponomarenko, F. Schedin, M. Katsnelson, R. Yang, E. Hill, K. Novoselov, and A. Geim. Chaotic dirac billiard in graphene quantum dots. *Science*, 320(5874):356–358, 2008.

[47] J. Kedzierski, P.-L. Hsu, P. Healey, P. W. Wyatt, C. L. Keast, M. Sprinkle, C. Berger, and W. A. De Heer. Epitaxial graphene transistors on sic substrates. *IEEE Trans. Electron Devices*, 55(8):2078–2085, 2008.

[48] Y.-M. Lin, C. Dimitrakopoulos, K. A. Jenkins, D. B. Farmer, H.-Y. Chiu, A. Grill, and P. Avouris. 100-ghz transistors from wafer-scale epitaxial graphene. *Science*, 327(5966):662–662, 2010.

[49] L. Yang, T. Hu, R. Hao, C. Qiu, C. Xu, H. Yu, Y. Xu, X. Jiang, Y. Li, and J. Yang. Low-chirp high-extinction-ratio modulator based on graphene–silicon waveguide. *Opt. Lett.*, 38(14):2512–2515, 2013.

[50] X. Li, M. Zhu, M. Du, Z. Lv, L. Zhang, Y. Li, Y. Yang, T. Yang, X. Li, K. Wang, H. Zhu, and Y. Fang. High detectivity graphene-silicon heterojunction photodetector. *Small*, 12(5):595–601, 2015.

[51] X. Li, H. Zhu, K. Wang, A. Cao, J. Wei, C. Li, Y. Jia, Z. Li, X. Li, and D. Wu. Graphene-on-silicon schottky junction solar cells. *Adv. Mater.*, 22(25):2743–2748, 2010.

[52] S. Hu, M. Lozada-Hidalgo, F. Wang, A. Mishchenko, F. Schedin, R. Nair, E. Hill, D. Boukhvalov, M. Katsnelson, R. Dryfe, et al. Proton transport through one-atom-thick crystals. *Nature*, 516(7530):227, 2014.

[53] C. Singh, S. Nikhil, A. Jana, A. K. Mishra, and A. Paul. Proton conduction through oxygen functionalized few-layer graphene. *Chem. Commun.*, 52(85):12661–12664, 2016.

[54] M. D. Stoller, S. Park, Y. Zhu, J. An, and R. S. Ruoff. Graphene-based ultracapacitors. *Nano Lett.*, 8(10):3498–3502, 2008.

[55] F. Yao, F. Gunes, H. Q. Ta, S. M. Lee, S. J. Chae, K. Y. Sheem, C. S. Cojocaru, S. S. Xie, and Y. H. Lee. Diffusion mechanism of lithium ion through basal plane of layered graphene. *J. Am. Chem. Soc.*, 134(20):8646–8654, 2012.

[56] Y. Dan, Y. Lu, N. J. Kybert, Z. Luo, and A. C. Johnson. Intrinsic response of graphene vapor sensors. *Nano Lett.*, 9(4):1472–1475, 2009.

[57] Y. Xu, Z. Guo, H. Chen, Y. Yuan, J. Lou, X. Lin, H. Gao, H. Chen, and B. Yu. In-plane and tunneling pressure sensors based on graphene/hexagonal boron nitride heterostructures. *Appl. Phys. Lett.*, 99(13):133109, 2011.

[58] C. S. Boland, U. Khan, C. Backes, A. O'Neill, J. McCauley, S. Duane, R. Shanker, Y. Liu, I. Jurewicz, A. B. Dalton, et al. Sensitive, high-strain, high-rate bodily motion sensors based on graphene–rubber composites. *ACS Nano*, 8(9):8819–8830, 2014.

[59] X. Li, T. Yang, Y. Yang, J. Zhu, L. Li, F. E. Alam, X. Li, K. Wang, H. Cheng, C.-T. Lin, et al. Large-area ultrathin graphene films by single-step marangoni

self-assembly for highly sensitive strain sensing application. *Adv. Funct. Mater.*, 26(9):1322–1329, 2016.

[60] G. Lalwani, A. M. Henslee, B. Farshid, L. Lin, F. K. Kasper, Y.-X. Qin, A. G. Mikos, and B. Sitharaman. Two-dimensional nanostructure-reinforced biodegradable polymeric nanocomposites for bone tissue engineering. *Biomacromolecules*, 14(3):900–909, 2013.

[61] A. Khaliq, R. Kafafy, H. M. Salleh, and W. F. Faris. Enhancing the efficiency of polymerase chain reaction using graphene nanoflakes. *Nanotechnology*, 23(45):455106, 2012.

[62] N. Mohanty and V. Berry. Graphene-based single-bacterium resolution biodevice and dna transistor: interfacing graphene derivatives with nanoscale and microscale biocomponents. *Nano Lett.*, 8(12):4469–4476, 2008.

[63] R. Tkacz, R. Oldenbourg, S. B. Mehta, M. Miansari, A. Verma, and M. Majumder. ph dependent isotropic to nematic phase transitions in graphene oxide dispersions reveal droplet liquid crystalline phases. *Chem. Commun.*, 50(50):6668–6671, 2014.

[64] G. Lalwani, M. D'Agati, A. M. Khan, and B. Sitharaman. Toxicology of graphene-based nanomaterials. *Adv. Drug Delivery Rev.*, 105:109–144, 2016.

[65] M. Wen, S. M. Whalen, R. S. Elliott, and E. B. Tadmor. Interpolation effects in tabulated interatomic potentials. *Modell. Simul. Mater. Sci. Eng.*, 23(7):074008, 2015.

[66] M. Wen, J. Li, P. Brommer, R. S. Elliott, J. P. Sethna, and E. B. Tadmor. A kim-compliant potfit for fitting sloppy interatomic potentials: application to the edip model for silicon. *Modell. Simul. Mater. Sci. Eng.*, 25(1):014001, 2017.

[67] M. Wen, S. N. Shirodkar, P. Plecháč, E. Kaxiras, R. S. Elliott, and E. B. Tadmor. A force-matching stillinger-weber potential for MoS2: Parameterization and fisher information theory based sensitivity analysis. *J. Appl. Phys.*, 122(24):244301, 2017.

[68] M. Wen, S. Carr, S. Fang, E. Kaxiras, and E. B. Tadmor. Dihedral-angle-corrected registry-dependent interlayer potential for multilayer graphene structures. *Phys. Rev. B*, 98(23), 2018.

[69] KLIFF: Kim-based learning-integrated fitting framework. https://kliff.readthedocs.io, 2019. Retrieved: 2019-04-15.

[70] Open knowledgebase of interatomic models (OpenKIM). https://openkim.org, 2019. Retrieved: 2019-04-15.

[71] Large-scale atomic/molecular massively parallel simulator (LAMMPS). http://lammps.sandia.gov, 2019. Retrieved: 2019-04-15.

[72] ASE: The atomic simulation environment—a python library for working with atoms. https://wiki.fysik.dtu.dk/ase/, 2019. Retrieved: 2019-04-15.

[73] General utility lattice program (GULP). http://nanochemistry.curtin.edu.au/gulp/, 2019. Retrieved: 2019-04-15.

[74] DL_POLY classic molecular simulation package. https://www.scd.stfc.ac.uk/Pages/DL_POLY.aspx, 2019. Retrieved: 2019-04-15.

[75] J. E. Jones. On the determination of molecular fields. I. from the variation of the viscosity of a gas with temperature. *Proc. Roy. Soc. A*, 106(738):441–462, 1924.

[76] J. E. Jones. On the determination of molecular fields. II. from the equation of state of a gas. *Proc. Roy. Soc. A*, 106(738):463–477, 1924.

[77] J. E. Lennard-Jones. Cohesion. *Proc. Phys. Soc.*, 43(5):461–482, 1931.

[78] C. Kittel. *Introduction to Solid State Physics*. Wiley, 8 edition, 2004.

[79] F. H. Stillinger and T. A. Weber. Computer simulation of local order in condensed phases of silicon. *Phys. Rev. B*, 31(8):5262, 1985.

[80] J. A. Moriarty. Density-functional formulation of the generalized pseudopotential theory. iii. transition-metal interatomic potentials. *Phys. Rev. B*, 38(5):3199, 1988.

[81] M. S. Daw and M. I. Baskes. Embedded-atom method: Derivation and application to impurities, surfaces, and other defects in metals. *Phys. Rev. B*, 29:6443–6453, Jun 1984.

[82] J. Tersoff. New empirical model for the structural properties of silicon. *Phys. Rev. Lett.*, 56(6):632, 1986.

[83] D. W. Brenner, O. A. Shenderova, J. A. Harrison, S. J. Stuart, B. Ni, and S. B. Sinnott. A second-generation reactive empirical bond order (REBO) potential energy expression for hydrocarbons. *J. Phys.: Condens. Matter*, 14(4):783–802, 2002.

[84] J. H. Los and A. Fasolino. Intrinsic long-range bond-order potential for carbon: Performance in monte carlo simulations of graphitization. *Phys. Rev. B*, 68(2):024107, 2003.

[85] A. C. T. van Duin, S. Dasgupta, F. Lorant, and W. A. Goddard. ReaxFF: a reactive force field for hydrocarbons. *J. Phys. Chem. A*, 105(41):9396–9409, 2001.

[86] Z. Fan, L. F. C. Pereira, H.-Q. Wang, J.-C. Zheng, D. Donadio, and A. Harju. Force and heat current formulas for many-body potentials in molecular dynamics simulations with applications to thermal conductivity calculations. *Phys. Rev. B*, 92(9):094301, 2015.

[87] M. S. Daw. Model of metallic cohesion: The embedded-atom method. *Phys. Rev. B*, 39:7441–7452, Apr 1989.

[88] M. S. Daw, S. M. Foiles, and M. I. Baskes. The embedded-atom method: a review of theory and applications. *Mater. Sci. Rep.*, 9(7):251–310, 1993.

[89] M. Finnis and J. Sinclair. A simple empirical n-body potential for transition metals. *Philos. Mag. A*, 50(1):45–55, 1984.

[90] K. W. Jacobsen, J. K. Nørskov, and M. J. Puska. Interatomic interactions in the effective-medium theory. *Phys. Rev. B*, 35:7423–7442, 1987.

[91] M. I. Baskes. Modified embedded-atom potentials for cubic materials and impurities. *Phys. Rev. B*, 46(5):2727–2742, 1992.

[92] Y. Mishin, M. J. Mehl, and D. A. Papaconstantopoulos. Phase stability in the fe–ni system: Investigation by first-principles calculations and atomistic simulations. *Acta Mater.*, 53(15):4029–4041, 2005.

[93] F. Apostol and Y. Mishin. Interatomic potential for the al-cu system. *Phys. Rev. B*, 83:054116, Feb 2011.

[94] J. M. Bowman. Self-consistent field energies and wavefunctions for coupled oscillators. *J. Chem. Phys.*, 68(2):608–610, 1978.

[95] R. A. MacDonald and W. M. MacDonald. Thermodynamic properties of fcc metals at high temperatures. *Phys. Rev. B*, 24:1715–1724, Aug 1981.

[96] C. M. Bishop. *Pattern recognition and machine learning*. springer, 2006.

[97] Y. Bar-Shalom, X. R. Li, and T. Kirubarajan. *Estimation with applications to tracking and navigation: theory algorithms and software*. John Wiley & Sons, 2004.

[98] A. Nichol and G. J. Ackland. Property trends in simple metals: An empirical potential approach. *Phys. Rev. B*, 93:184101, 2016.

[99] F. Ercolessi and J. B. Adams. Interatomic potentials from first-principles calculations: the force-matching method. *Europhys. Lett.*, 26(8):583–588, 1994.

[100] P. Brommer and F. Gähler. Effective potentials for quasicrystals from ab-initio data. *Philos. Mag.*, 86(6-8):753–758, 2006.

[101] P. Brommer and F. Gähler. Potfit: effective potentials from ab initio data. *Modell. Simul. Mater. Sci. Eng.*, 15(3):295, 2007.

[102] J. J. Waterfall, F. P. Casey, R. N. Gutenkunst, K. S. Brown, C. R. Myers, P. W. Brouwer, V. Elser, and J. P. Sethna. Sloppy model universality class and the Vandermonde matrix. *Phys. Rev. Lett.*, 97:150601, 2006.

[103] M. K. Transtrum, B. B. Machta, and J. P. Sethna. Geometry of nonlinear least squares with applications to sloppy models and optimization. *Phys. Rev. E*, 83(3):036701, 2011.

[104] M. K. Transtrum, B. B. Machta, and J. P. Sethna. Why are nonlinear fits to data so challenging? *Phys. Rev. Lett.*, 104(6):060201, 2010.

[105] K. Levenberg. A method for the solution of certain non-linear problems in least squares. *Quart. Appl. Math.*, 2(2):164–168, 1944.

[106] D. W. Marquardt. An algorithm for least-squares estimation of nonlinear parameters. *J. Soc. Ind. Appl. Math.*, 11(2):431–441, 1963.

[107] J. F. Justo, M. Z. Bazant, E. Kaxiras, V. Bulatov, and S. Yip. Interatomic potential for silicon defects and disordered phases. *Phys. Rev. B*, 58(5):2539, 1998.

[108] M. J. D. Powell. A method for minimizing a sum of squares of non-linear functions without calculating derivatives. *Comp. J.*, 7(4):303–307, 1965.

[109] N. Wakabayashi, H. G. Smith, and R. M. Nicklow. Lattice dynamics of hexagonal mos_2 studied by neutron scattering. *Phys. Rev. B*, 12(2):659, 1975.

[110] T. Liang, S. R. Phillpot, and S. B. Sinnott. Parametrization of a reactive many-body potential for mo–s systems. *Phys. Rev. B*, 79(24):245110, 2009.

[111] G. C. Abell. Empirical chemical pseudopotential theory of molecular and metallic bonding. *Phys. Rev. B*, 31(10):6184–6196, 1985.

[112] J. Tersoff. New empirical approach for the structure and energy of covalent systems. *Phys. Rev. B*, 37(12):6991, 1988.

[113] J. Tersoff. Empirical interatomic potential for silicon with improved elastic properties. *Phys. Rev. B*, 38(14):9902, 1988.

[114] D. W. Brenner. Empirical potential for hydrocarbons for use in simulating the chemical vapor deposition of diamond films. *Phys. Rev. B*, 42(15):9458–9471, 1990.

[115] J.-W. Jiang, H. S. Park, and T. Rabczuk. Molecular dynamics simulations of single-layer molybdenum disulphide (mos2): Stillinger-weber parametrization,

mechanical properties, and thermal conductivity. *J. Appl. Phys.*, 114(6):064307, 2013.

[116] J.-W. Jiang. Parametrization of Stillinger–Weber potential based on valence force field model: application to single-layer mos2 and black phosphorus. *Nanotechnology*, 26(31):315706, 2015.

[117] A. Ostadhossein, A. Rahnamoun, Y. Wang, P. Zhao, S. Zhang, V. H. Crespi, and A. C. van Duin. Reaxff reactive force-field study of molybdenum disulfide (mos2). *J. Phys. Chem. Lett.*, 8(3):631–640, 2017.

[118] R. A. Fisher. On the mathematical foundations of theoretical statistics. *Philos. Trans. R. Soc. London, Ser. A*, 222:309–368, 1922.

[119] Y. Pantazis and M. A. Katsoulakis. A relative entropy rate method for path space sensitivity analysis of stationary complex stochastic dynamics. *J. Chem. Phys.*, 138(5):054115, 2013.

[120] P. Dupuis, M. A. Katsoulakis, Y. Pantazis, and P. Plecháč. Path-space information bounds for uncertainty quantification and sensitivity analysis of stochastic dynamics. *SIAM/ASA J. Uncertainty Quantif.*, 4(1):80–111, 2016.

[121] E. Kalligiannaki, V. Harmandaris, M. A. Katsoulakis, and P. Plecháč. The geometry of generalized force matching and related information metrics in coarse-graining of molecular systems. *J. Chem. Phys.*, 143(8):084105, 2015.

[122] V. Harmandaris, E. Kalligiannaki, M. Katsoulakis, and P. Plecháč. Path-space variational inference for non-equilibrium coarse-grained systems. *J. Comput. Phys.*, 314:355–383, 2016.

[123] P. Español and I. Zuniga. Obtaining fully dynamic coarse-grained models from MD. *Phys. Chem. Chem. Phys.*, 13:10538–10545, 2011.

[124] L. Lu, J. F. Dama, and G. A. Voth. Fitting coarse-grained distribution functions through an iterative force-matching method. *J. Chem. Phys.*, 139(12):121906, 2013.

[125] A. van den Bos. *Parameter estimation for scientists and engineers*. John Wiley & Sons, Hoboken, 2007.

[126] H. Cramér. *Mathematical Methods of Statistics*. Princeton University Press, Princeton, 1999.

[127] Z. Jian, Z. Kaiming, and X. Xide. Modification of Stillinger-Weber potentials Si and Ge. *Phys. Rev. B*, 41(18):12915–12918, 1990.

[128] M. Ichimura. Stillinger-Weber potential for III-V compound semiconductors and their application to the critical thickness calculation for InAs/GaAs. *Phys. Stat. Sol. A*, 153(2):431–437, 1996.

[129] X. W. Zhou, D. K. Ward, J. E. Martin, F. B. van Swol, J. L. Cruz-Campa, and D. Zubia. Stillinger-weber potential for the ii-vi elements zn-cd-hg-s-se-te. *Phys. Rev. B*, 88(8):085309, 2013.

[130] J. Li. Atomeye: an efficient atomistic configuration viewer. *Modell. Simul. Mater. Sci. Eng.*, 11(2):173, 2003.

[131] J. M. Soler, E. Artacho, J. D. Gale, A. García, J. Junquera, P. Ordejón, and D. Sánchez-Portal. The siesta method for ab initio order-n materials simulation. *J. Phys.: Condens. Matter*, 14(11):2745, 2002.

[132] J. D. Gale. GULP: A computer program for the symmetry-adapted simulation of solids. *J. Chem. Soc.-Farad. Trans.*, 93(4):629–637, 1997.

[133] D. R. Hamann, M. Schlüter, and C. Chiang. Norm-conserving pseudopotentials. *Phys. Rev. Lett.*, 43(20):1494–1497, 1979.

[134] N. Troullier and J. L. Martins. Efficient pseudopotentials for plane-wave calculations. *Phys. Rev. B*, 43(3):1993–2006, 1991.

[135] J. P. Perdew, K. Burke, and M. Ernzerhof. Generalized gradient approximation made simple. *Phys. Rev. Lett.*, 77(18):3865–3868, 1996.

[136] L. F. Huang, P. L. Gong, and Z. Zeng. Correlation between structure, phonon spectra, thermal expansion, and thermomechanics of single-layer mos2. *Phys. Rev. B*, 90(4):045409, jul 2014.

[137] S. Plimpton. Fast parallel algorithms for short-range molecular dynamics. *J. Comput. Phys.*, 117(1):1–19, 1995.

[138] C. Ataca, H. Sahin, E. Akturk, and S. Ciraci. Mechanical and electronic properties of MoS2 nanoribbons and their defects. *J. Phys. Chem. C*, 115(10):3934–3941, 2011.

[139] Y. Ding, Y. Wang, J. Ni, L. Shi, S. Shi, and W. Tang. First principles study of structural, vibrational and electronic properties of graphene-like MX2 (m=mo, nb, w, ta; x=s, se, te) monolayers. *Physica B*, 406(11):2254–2260, may 2011.

[140] R. C. Cooper, C. Lee, C. A. Marianetti, X. Wei, J. Hone, and J. W. Kysar. Nonlinear elastic behavior of two-dimensional molybdenum disulfide. *Phys. Rev. B*, 87(3):035423, jan 2013.

[141] D. Çakır, F. M. Peeters, and C. Sevik. Mechanical and thermal properties of h-MX2 (m = cr, mo, w; x = o, s, se, te) monolayers: A comparative study. *Appl. Phys. Lett.*, 104(20):203110, 2014.

[142] M. P. Allen and D. J. Tildesley. *Computer simulation of liquids*. Oxford University Press, Oxford, 1989.

[143] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, A. Passos, D. Cournapeau, M. Brucher, M. Perrot, and E. Duchesnay. Scikit-learn: Machine learning in Python. *J. Mach. Learn. Res.*, 12:2825–2830, 2011.

[144] Scikit-learn: Machine learning in Python. http://scikit-learn.org, 2019. Retrieved: 2019-04-15.

[145] P. Anees, M. Valsakumar, and B. Panigrahi. Anharmonicity of optic modes in monolayer mos2. *Appl. Phys. Lett.*, 108(10):101902, 2016.

[146] C. Sevik. Assessment on lattice thermal properties of two-dimensional honeycomb structures: Graphene, h-bn, h-mos 2, and h-mose 2. *Phys. Rev. B*, 89(3):035422, 2014.

[147] J. Tersoff. Modeling solid-state chemistry: Interatomic potentials for multicomponent systems. *Phys. Rev. B*, 39(8):5566–5568, 1989.

[148] S. J. Stuart, A. B. Tutein, and J. A. Harrison. A reactive potential for hydrocarbons with intermolecular interactions. *J. Chem. Phys.*, 112(14):6472–6486, 2000.

[149] K. Zhang and E. B. Tadmor. Energy and moiré patterns in 2d bilayers in translation and rotation: A study using an efficient discrete–continuum interlayer potential. *Extreme Mech. Lett.*, 14:16–22, 2017.

[150] A. N. Kolmogorov and V. H. Crespi. Registry-dependent interlayer potential for graphitic systems. *Phys. Rev. B*, 71(23):235415, 2005.

[151] I. Leven, I. Azuri, L. Kronik, and O. Hod. Inter-layer potential for hexagonal boron nitride. *J. Chem. Phys.*, 140(10):104106, 2014.

[152] I. Leven, T. Maaravi, I. Azuri, L. Kronik, and O. Hod. Interlayer potential for graphene/h-BN heterostructures. *J. Chem. Theory Comput.*, 12(6):2896–2905, 2016.

[153] P. M. Morse. Diatomic molecules according to the wave mechanics. II. vibrational levels. *Phys. Rev.*, 34(1):57–64, 1929.

[154] I. V. Lebedeva, A. A. Knizhnik, A. M. Popov, Y. E. Lozovik, and B. V. Potapkin. Interlayer interaction and relative vibrations of bilayer graphene. *Phys. Chem. Chem. Phys.*, 13(13):5687, 2011.

[155] T. Maaravi, I. Leven, I. Azuri, L. Kronik, and O. Hod. Interlayer potential for homogeneous graphene and hexagonal boron nitride systems: Reparametrization for many-body dispersion effects. *J. Phys. Chem. C*, 121(41):22826–22835, 2017.

[156] G. Kresse and J. Furthmüller. Efficient iterative schemes for ab initio total-energy calculations using a plane-wave basis set. *Phys. Rev. B*, 54(16):11169–11186, 1996.

[157] G. Kresse and J. Furthmüller. Efficiency of ab-initio total energy calculations for metals and semiconductors using a plane-wave basis set. *Comput. Mater. Sci*, 6(1):15–50, 1996.

[158] S. Grimme. Semiempirical GGA-type density functional constructed with a long-range dispersion correction. *J. Comput. Chem.*, 27(15):1787–1799, 2006.

[159] S. Grimme, J. Antony, S. Ehrlich, and H. Krieg. A consistent and accurate ab initio parametrization of density functional dispersion correction (DFT-d) for the 94 elements h-pu. *J. Chem. Phys.*, 132(15):154104, 2010.

[160] A. Tkatchenko and M. Scheffler. Accurate molecular van der waals interactions from ground-state electron density and free-atom reference data. *Phys. Rev. Lett.*, 102(7):073005, 2009.

[161] T. Bučko, S. Lebègue, J. Hafner, and J. G. Ángyán. Improved density dependent correction for the description of london dispersion forces. *J. Chem. Theory Comput.*, 9(10):4293–4299, 2013.

[162] A. Tkatchenko, R. A. DiStasio, R. Car, and M. Scheffler. Accurate and efficient method for many-body van der waals interactions. *Phys. Rev. Lett.*, 108(23):236402, 2012.

[163] S. N. Steinmann and C. Corminboeuf. A generalized-gradient approximation exchange hole model for dispersion coefficients. *J. Chem. Phys.*, 134(4):044117, 2011.

[164] S. Zhou, J. Han, S. Dai, J. Sun, and D. J. Srolovitz. van der waals bilayer energetics: Generalized stacking-fault energy of graphene, boron nitride, and graphene/boron nitride bilayers. *Phys. Rev. B*, 92(15):155438, 2015.

[165] S. Lebègue, J. Harl, T. Gould, J. G. Ángyán, G. Kresse, and J. F. Dobson. Cohesive properties and asymptotics of the dispersion interaction in graphite by the random phase approximation. *Phys. Rev. Lett.*, 105(19):196401, 2010.

[166] Y. Baskin and L. Meyer. Lattice constants of graphite at low temperatures. *Phys. Rev.*, 100(2):544–544, 1955.

[167] L. A. Girifalco and R. A. Lad. Energy of cohesion, compressibility, and the potential energy functions of the graphite system. *J. Chem. Phys.*, 25(4):693–697, 1956.

[168] L. X. Benedict, N. G. Chopra, M. L. Cohen, A. Zettl, S. G. Louie, and V. H. Crespi. Microscopic determination of the interlayer binding energy in graphite. *Chem. Phys. Lett.*, 286(5-6):490–496, 1998.

[169] R. Zacharia, H. Ulbricht, and T. Hertel. Interlayer cohesive energy of graphite from thermal desorption of polyaromatic hydrocarbons. *Phys. Rev. B*, 69(15):155406, 2004.

[170] A. M. Popov, I. V. Lebedeva, A. A. Knizhnik, Y. E. Lozovik, and B. V. Potapkin. Barriers to motion and rotation of graphene layers based on measurements of shear mode frequencies. *Chem. Phys. Lett.*, 536:82–86, 2012.

[171] O. L. Blakslee, D. G. Proctor, E. J. Seldin, G. B. Spence, and T. Weng. Elastic constants of compression-annealed pyrolytic graphite. *J. Appl. Phys.*, 41(8):3373–3382, 1970.

[172] A. Bosak, M. Krisch, M. Mohr, J. Maultzsch, and C. Thomsen. Elasticity of single-crystalline graphite: Inelastic x-ray scattering study. *Phys. Rev. B*, 75(15):153408, 2007.

[173] S. Shallcross, S. Sharma, E. Kandelaki, and O. A. Pankratov. Electronic structure of turbostratic graphene. *Phys. Rev. B*, 81(16):165105, 2010.

[174] G. A. Tritsaris, S. N. Shirodkar, E. Kaxiras, P. Cazeaux, M. Luskin, P. Plecháč, and E. Cancès. Perturbation theory for weakly coupled two-dimensional layers. *J. Mater. Res.*, 31(07):959–966, 2016.

[175] K. Zhang and E. B. Tadmor. Structural and electron diffraction scaling of twisted graphene bilayers. *J. Mech. Phys. Solids*, 112:225–238, 2018.

[176] L. A. Gonzalez-Arraga, J. L. Lado, F. Guinea, and P. San-Jose. Electrically controllable magnetism in twisted bilayer graphene. *Phys. Rev. Lett.*, 119(10):107201, 2017.

[177] K. P. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012.

[178] J. Behler and M. Parrinello. Generalized neural-network representation of high-dimensional potential-energy surfaces. *Phys. Rev. Lett.*, 98(14):146401, 2007.

[179] A. P. Bartók, M. C. Payne, R. Kondor, and G. Csányi. Gaussian Approximation Potentials: The accuracy of quantum mechanics, without the electrons. *Phys. Rev. Lett.*, 104(13):136403, 2010.

[180] M. Rupp, A. Tkatchenko, K.-R. Müller, and O. A. Von Lilienfeld. Fast and accurate modeling of molecular atomization energies with machine learning. *Phys. Rev. Lett.*, 108(5):058301, 2012.

[181] A. V. Shapeev. Moment tensor potentials: A class of systematically improvable interatomic potentials. *Multiscale Model. Simul.*, 14(3):1153–1173, 2016.

[182] S. Hajinazar, J. Shao, and A. N. Kolmogorov. Stratified construction of neural network based interatomic models for multicomponent materials. *Phys. Rev. B*, 95(1):014114, 2017.

[183] A. P. Bartók, R. Kondor, and G. Csányi. On representing chemical environments. *Phys. Rev. B*, 87(18):184115, 2013.

[184] G. Montavon, M. Rupp, V. Gobre, A. Vazquez-Mayagoitia, K. Hansen, A. Tkatchenko, K.-R. Müller, and O. A. Von Lilienfeld. Machine learning of molecular electronic properties in chemical compound space. *New J. Phys.*, 15(9):095003, 2013.

[185] M. Rupp, R. Ramakrishnan, and O. A. von Lilienfeld. Machine learning for quantum mechanical properties of atoms in molecules. *J. Phys. Chem. Lett.*, 6(16):3309–3313, 2015.

[186] J. E. Moussa. Comment on "fast and accurate modeling of molecular atomization energies with machine learning". *Phys. Rev. Lett.*, 109(5):059801, 2012.

[187] J. Behler. Atom-centered symmetry functions for constructing high-dimensional neural network potentials. *J. Chem. Phys.*, 134(7):074106, 2011.

[188] E. P. Wigner. *Group theory and its application to the quantum mechanics of atomic spectra*. Academic Press, 1959.

[189] D. Varshalovich, D. Varshalovich, A. Moskalev, V. Khersonskiĭ, and W. S. (Singapur). *Quantum Theory of Angular Momentum: Irreducible Tensors, Spherical Harmonics, Vector Coupling Coefficients, 3nj Symbols*. World Scientific Pub., 1988.

[190] A. Bartók-Pkirtay. *The Gaussian Approximation Potential: An Interatomic Potential Derived from First Principles Quantum Mechanics*. Springer Science & Business Media, 2010.

[191] K. Hansen, F. Biegler, R. Ramakrishnan, W. Pronobis, O. A. Von Lilienfeld, K.-R. Müller, and A. Tkatchenko. Machine learning predictions of molecular properties: Accurate many-body potentials and nonlocality in chemical space. *J. Phys. Chem. Lett.*, 6(12):2326–2331, 2015.

[192] H. Huo and M. Rupp. Unified representation of molecules and crystals for machine learning. *arXiv preprint arXiv:1704.06439*, 2017.

[193] A. P. Thompson, L. P. Swiler, C. R. Trott, S. M. Foiles, and G. J. Tucker. Spectral neighbor analysis method for automated generation of quantum-accurate interatomic potentials. *J. Comput. Phys.*, 285:316–330, 2015.

[194] P. Rowe, G. Csányi, D. Alfè, and A. Michaelides. Development of a machine learning potential for graphene. *Phys. Rev. B*, 97(5):054303, 2018.

[195] A. P. Bartók, J. Kermode, N. Bernstein, and G. Csányi. Machine learning a general-purpose interatomic potential for silicon. *Phys. Rev. X*, 8(4):041048, 2018.

[196] G. E. Hinton, D. Rumelhart, and R. J. Williams. Learning representations by back-propagating errors. *Nature*, 323(9):533–536, 1986.

[197] I. Goodfellow, Y. Bengio, and A. Courville. *Deep learning*. MIT press, 2016.

[198] K. Zhang and E. B. Tadmor. Energy and moiré patterns in 2D bilayers in translation and rotation: A study using an efficient discrete–continuum interlayer potential. *Extreme Mech. Lett.*, 14:16–22, 2017.

[199] H. Yoo, R. Engelke, S. Carr, S. Fang, K. Zhang, P. Cazeaux, S. H. Sung, R. Hovden, A. Tsen, T. Taniguchi, K. Watanabe, G.-C. Yi, M. Kim, M. Luskin, E. B.

Tadmor, E. Kaxiras, and P. Kim. Atomic and electronic reconstruction at the van der waals interface in twisted bilayer graphene. *Nat. Mater.*, 18(5):448–453, 2019.

[200] T. C. O'Connor, J. Andzelm, and M. O. Robbins. AIREBO-m: A reactive model for hydrocarbons at extreme pressures. *J. Chem. Phys.*, 142(2):024903, 2015.

[201] S. G. Srinivasan, A. C. T. van Duin, and P. Ganesh. Development of a ReaxFF potential for carbon condensed phases and its application to the thermal fragmentation of a large fullerene. *J. Phys. Chem. A*, 119(4):571–580, 2015.

[202] L. Liu, J. Gao, X. Zhang, T. Yan, and F. Ding. Vacancy inter-layer migration in multi-layered graphene. *Nanoscale*, 6(11):5729–5734, 2014.

[203] V. L. Deringer and G. Csányi. Machine learning based interatomic potential for amorphous carbon. *Phys. Rev. B*, 95(9):094203, 2017.

[204] R. Z. Khaliullin, H. Eshet, T. D. Kühne, J. Behler, and M. Parrinello. Graphite-diamond phase coexistence study employing a neural-network mapping of the ab initio potential energy surface. *Phys. Rev. B*, 81(10):100103, 2010.

[205] C. Zhu, R. H. Byrd, P. Lu, and J. Nocedal. Algorithm 778: L-BFGS-b: Fortran subroutines for large-scale bound-constrained optimization. *ACM Trans. Math. Software*, 23(4):550–560, 1997.

[206] L. Lin and S. Zhang. Creating high yield water soluble luminescent graphene quantum dots via exfoliating and disintegrating carbon nanotubes and graphite flakes. *Chem. Commun.*, 48(82):10177, 2012.

[207] D. R. Cooper, B. D'Anjou, N. Ghattamaneni, B. Harack, M. Hilke, A. Horth, N. Majlis, M. Massicotte, L. Vandsburger, E. Whiteway, and V. Yu. Experimental review of graphene. *ISRN Condensed Matter Physics*, 2012:1–56, 2012.

[208] F. Birch. Finite elastic strain of cubic crystals. *Phys. Rev.*, 71(11):809–824, 1947.

[209] A. Togo and I. Tanaka. First principles phonon calculations in materials science. *Scr. Mater.*, 108:1–5, 2015.

[210] J.-A. Yan, W. Y. Ruan, and M. Y. Chou. Phonon dispersions and vibrational properties of monolayer, bilayer, and trilayer graphene: Density-functional perturbation theory. *Phys. Rev. B*, 77(12):125401, 2008.

[211] L. Wirtz and A. Rubio. The phonon dispersion of graphite revisited. *Solid State Commun.*, 131(3-4):141–152, 2004.

[212] E. B. Tadmor, R. S. Elliott, J. P. Sethna, R. E. Miller, and C. A. Becker. The potential of atomistic simulations and the knowledgebase of interatomic models. *JOM*, 63(7):17–17, 2011.

[213] QUIP: a collection of software tools to carry out molecular dynamics simulations. http://www.libatoms.org/Home/Software, 2019. Retrieved: 2019-04-15.

[214] A. A. Balandin. Thermal properties of graphene and nanostructured carbon materials. *Nat. Mater.*, 10(8):569–581, 2011.

[215] L. F. C. Pereira and D. Donadio. Divergence of the thermal conductivity in uniaxially strained graphene. *Phys. Rev. B*, 87(12):125424, 2013.

[216] S. Ghosh, I. Calizo, D. Teweldebrhan, E. P. Pokatilov, D. L. Nika, A. A. Balandin, W. Bao, F. Miao, and C. N. Lau. Extremely high thermal conductivity of graphene: Prospects for thermal management applications in nanoelectronic circuits. *Appl. Phys. Lett.*, 92(15):151911, 2008.

[217] T. Y. Kim, C.-H. Park, and N. Marzari. The electronic thermal conductivity of graphene. *Nano Lett.*, 16(4):2439–2443, 2016.

[218] L. Lindsay, D. A. Broido, and N. Mingo. Flexural phonons and thermal transport in graphene. *Phys. Rev. B*, 82(11):115427, 2010.

[219] H. Zhang, G. Lee, and K. Cho. Thermal transport in graphene and effects of vacancy defects. *Phys. Rev. B*, 84(11):115460, 2011.

[220] M. Tuckerman. *Statistical mechanics: theory and molecular simulation*. Oxford University Press, Oxford, 2010.

[221] P. K. Schelling, S. R. Phillpot, and P. Keblinski. Comparison of atomic-level simulation methods for computing thermal conductivity. *Phys. Rev. B*, 65(14):144306, 2002.

[222] N. C. Admal and E. B. Tadmor. Stress and heat flux for arbitrary multibody potentials: A unified framework. *J. Chem. Phys.*, 134(18):184106, 2011.

[223] F. Banhart, J. Kotakoski, and A. V. Krasheninnikov. Structural defects in graphene. *ACS Nano*, 5(1):26–41, 2010.

[224] S. T. Skowron, I. V. Lebedeva, A. M. Popov, and E. Bichoutskaia. Energetics of atomic scale structure changes in graphene. *Chem. Soc. Rev.*, 44(10):3143–3176, 2015.

[225] M. H. Gass, U. Bangert, A. L. Bleloch, P. Wang, R. R. Nair, and A. K. Geim. Free-standing graphene at atomic resolution. *Nat. Nanotechnol.*, 3(11):676–681, 2008.

[226] J. Haskins, A. Kınacı, C. Sevik, H. Sevinçli, G. Cuniberti, and T. Çağın. Control of thermal and electronic transport in defect-engineered graphene nanoribbons. *ACS Nano*, 5(5):3779–3787, 2011.

[227] R. H. Telling, C. P. Ewels, A. A. El-Barbary, and M. I. Heggie. Wigner defects bridge the graphite gap. *Nat. Mater.*, 2(5):333–337, 2003.

[228] L. Vicarelli, S. J. Heerema, C. Dekker, and H. W. Zandbergen. Controlling defects in graphene for optimizing the electrical properties of graphene nanodevices. *ACS Nano*, 9(4):3428–3435, 2015.

[229] G. Teobaldi, H. Ohnishi, K. Tanimura, and A. L. Shluger. The effect of van der waals interactions on the properties of intrinsic defects in graphite. *Carbon*, 48(14):4145–4161, 2010.

[230] B. J. Alder and T. E. Wainwright. Studies in molecular dynamics. i. general method. *J. Chem. Phys.*, 31(2):459–466, 1959.

[231] T. M. Cover and J. A. Thomas. *Elements of information theory*. John Wiley & Sons, 2012.

[232] S. Kullback and R. A. Leibler. On information and sufficiency. *Ann. Math. Stat.*, 22(1):79–86, 1951.

[233] D. J. MacKay and D. J. Mac Kay. *Information theory, inference and learning algorithms*. Cambridge university press, 2003.

[234] A. D. Wilson, J. A. Schultz, and T. D. Murphey. Trajectory synthesis for fisher information maximization. *IEEE Trans. Robot.*, 30(6):1358–1370, 2014.

[235] A. Tsourtis, Y. Pantazis, M. A. Katsoulakis, and V. Harmandaris. Parametric sensitivity analysis for stochastic molecular systems using information theoretic metrics. *J. Chem. Phys.*, 143(1):014116, 2015.

[236] E. Beale. Confidence regions in non-linear estimation. *J. R. Stat. Soc.*, 22(1):41–76, 1960.

[237] S. L. Frederiksen, K. W. Jacobsen, K. S. Brown, and J. P. Sethna. Bayesian ensemble approach to error estimation of interatomic potentials. *Phys. Rev. Lett.*, 93(16):165501, 2004.

[238] F. Rizzi, H. N. Najm, B. J. Debusschere, K. Sargsyan, M. Salloum, H. Adalsteinsson, and O. M. Knio. Uncertainty quantification in MD simulations. part i: Forward propagation. *Multiscale Model. Simul.*, 10(4):1428–1459, 2012.

[239] F. Rizzi, H. N. Najm, B. J. Debusschere, K. Sargsyan, M. Salloum, H. Adalsteinsson, and O. M. Knio. Uncertainty quantification in MD simulations. part II: Bayesian inference of force-field parameters. *Multiscale Model. Simul.*, 10(4):1460–1492, 2012.

[240] D. M. Blei, A. Kucukelbir, and J. D. McAuliffe. Variational inference: A review for statisticians. *J. Am. Stat. Assoc.*, 112(518):859–877, 2017.

[241] M. A. Katsoulakis and P. Plecháč. Information-theoretic tools for parametrized coarse-graining of non-equilibrium extended systems. *J. Chem. Phys.*, 139(7):074115, 2013.

[242] C. Ridders. Accurate computation of $F'(x)$ and $F'(x)F''(x)$. *Adv. Eng. Software*, 4(2):75–76, 1982.

[243] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery. *Numerical Recipes: The Art of Scientific Computing*. Cambridge University Press, Cambridge, third edition, 2007.

[244] Y. Li, Y.-L. Li, C. M. Araujo, W. Luo, and R. Ahuja. Single-layer mos2 as an efficient photocatalyst. *Catal. Sci. Technol.*, 3(9):2214–2220, 2013.

[245] G. James, D. Witten, T. Hastie, and R. Tibshirani. *An introduction to statistical learning*, volume 112. Springer, 2013.

[246] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *CoRR*, abs/1207.0580, 2012, 1207.0580.

[247] N. Srivastava, G. Hinton, A. Krizhevsky, I. Sutskever, and R. Salakhutdinov. Dropout: a simple way to prevent neural networks from overfitting. *J. Mach. Learn. Res.*, 15(1):1929–1958, 2014.

[248] T. Bayes, R. Price, and J. Canton. An essay towards solving a problem in the doctrine of chances. *Philos. Trans.*, 53(0):370–418, 1763.

[249] Y. Gal. *Uncertainty in Deep Learning*. PhD thesis, University of Cambridge, 2016.

[250] A. Gelman, J. Carlin, H. Stern, D. Dunson, A. Vehtari, and D. Rubin. *Bayesian Data Analysis*. Chapman & Hall/CRC Texts in Statistical Science. CRC Press, 2013.

[251] Y. Gal and Z. Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the 33rd International Conference on Machine Learning (ICML-16)*, 2016.

[252] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

[253] M. Li, T. Zhang, Y. Chen, and A. J. Smola. Efficient mini-batch training for stochastic optimization. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 661–670. ACM, 2014.

[254] K. O. Arras. An introduction to error propagation: Derivation, meaning and examples of equation $c_y = f_x c_x f_x^t$. Technical Report EPFL-ASL-TR-98-01 R3, Swiss Federal Institute of Technology Lausanne (EPFL), 1998.

[255] L. McInnes, J. Healy, N. Saul, and L. Grossberger. Umap: Uniform manifold approximation and projection. *J. Open Source Softw.*, 3(29):861, 2018.

[256] E. D. Cubuk, B. D. Malone, B. Onat, A. Waterland, and E. Kaxiras. Representations in neural network based empirical potentials. *J. Chem. Phys.*, 147(2):024104, 2017.

[257] A. P. Thompson, S. J. Plimpton, and W. Mattson. General formulation of pressure and stress tensor for arbitrary many-body interaction potentials under periodic boundary conditions. *J. Chem. Phys.*, 131(15):154107, 2009.

[258] J. J. Mortensen, K. Kaasbjerg, S. L. Frederiksen, J. K. Nørskov, J. P. Sethna, and K. W. Jacobsen. Bayesian error estimation in density-functional theory. *Phys. Rev. Lett.*, 95(21):216401, 2005.

[259] KIM application programming interface (API). https://openkim.org/kim-api, 2019. Retrieved: 2019-04-15.

[260] M. Wen. Stillinger-Weber (SW) Model Driver v004. OpenKIM, https://doi.org/10.25950/f3abd2d6, 2018. Online; accessed: 2019-07-08.

[261] T. Brink. Model driver for Tersoff-style potentials ported from LAMMPS v003. OpenKIM, https://doi.org/10.25950/55b7b34e, 2019. Online; accessed: 2019-07-08.

[262] J. Tersoff. New empirical approach for the structure and energy of covalent systems. *Phys. Rev. B*, 37(12):6991–7000, 1988.

[263] J. Tersoff. Modeling solid-state chemistry: Interatomic potentials for multicomponent systems. *Phys. Rev. B*, 39:5566–5568, 1989.

[264] J. Nord, K. Albe, P. Erhart, and K. Nordlund. Modelling of compound semiconductors: analytical bond-order potential for gallium, nitrogen and gallium nitride. *J. Phys.: Condens. Matter*, 15:5649, 2003.

[265] D. S. Karls. Environment-Dependent Interatomic Potential (EDIP) model driver v002. OpenKIM, https://doi.org/10.25950/75c4686e, 2018. Online; accessed: 2019-07-08.

[266] M. Z. Bazant and E. Kaxiras. Modeling of covalent bonding in solids by inversion of cohesive energy curves. *Phys. Rev. Lett.*, 77:4370–4373, 1996.

[267] M. Z. Bazant, E. Kaxiras, and J. F. Justo. Environment-dependent interatomic potential for bulk silicon. *Phys. Rev. B*, 56:8542–8552, 1997.

[268] J. a. F. Justo, M. Z. Bazant, E. Kaxiras, V. V. Bulatov, and S. Yip. Interatomic potential for silicon defects and disordered phases. *Phys. Rev. B*, 58:2539–2550, 1998.

[269] R. S. Elliott. EAM Model Driver for tabulated potentials with cubic Hermite spline interpolation as used in LAMMPS v005. OpenKIM, https://doi.org/10.25950/68defa36, 2018. Online; accessed: 2019-07-08.

[270] E. Tadmor. Verification Check for Memory Leaks using Valgrind v001. OpenKIM, https://doi.org/10.25950/ba474f45, 2018. Online; accessed: 2019-07-08.

[271] E. Tadmor. Verification Check of Invariance with respect to the Inversion Operation (Inversion Symmetry) v001. OpenKIM, https://doi.org/10.25950/63a96579, 2018. Online; accessed: 2019-07-08.

[272] E. Tadmor. Verification Check of Invariance with respect to Atom Permutations (Permutation Symmetry) v001. OpenKIM, https://doi.org/10.25950/dfbf8222, 2018. Online; accessed: 2019-07-08.

[273] E. Tadmor. Verification Check of Forces via Numerical Differentiation (Richardson Extrapolation Technique) v002. OpenKIM, https://doi.org/10.25950/9be59b8d, 2018. Online; accessed: 2019-07-08.

[274] S. Pattamatta. Stacking and twinning fault energies of an fcc lattice at zero temperature and pressure v001. OpenKIM, https://doi.org/10.25950/d6ffade7, 2018. Online; accessed: 2019-07-08.

[275] J. Li and E. Tadmor. Elastic constants for cubic crystals at zero temperature and pressure v005. OpenKIM, https://doi.org/10.25950/49c5c255, 2019. Online; accessed: 2019-07-08.

[276] M. Wen. Linear thermal expansion coefficient of a cubic crystal structure at a given temperature and pressure v000. OpenKIM, https://openkim.org/cite/TD_522633393614_000, 2016. Online; accessed: 2019-07-08.

[277] A. H. Larsen, J. J. Mortensen, J. Blomqvist, I. E. Castelli, R. Christensen, M. Dułak, J. Friis, M. N. Groves, B. Hammer, C. Hargus, E. D. Hermes, P. C. Jennings, P. B. Jensen, J. Kermode, J. R. Kitchin, E. L. Kolsbjerg, J. Kubal, K. Kaasbjerg, S. Lysgaard, J. B. Maronsson, T. Maxson, T. Olsen, L. Pastewka, A. Peterson, C. Rostgaard, J. Schiøtz, O. Schütt, M. Strange, K. S. Thygesen, T. Vegge, L. Vilhelmsen, M. Walter, Z. Zeng, and K. W. Jacobsen. The atomic simulation environment—a python library for working with atoms. *J. Phys.: Condens. Matter*, 29(27):273002, 2017.

[278] W. Smith and T. R. Forester. DL_POLY_2.0: A general-purpose molecular dynamics simulation package. *J. Mol. Graph.*, 14:136–141, 1996.

[279] Asap: a calculator for doing large-scale classical molecular dynamics. https://wiki.fysik.dtu.dk/asap/, 2019. Retrieved: 2019-04-15.

[280] The PyTorch deep learning platform. https://pytorch.org, 2019. Retrieved: 2019-04-15.

[281] SciPy: a Python-based ecosystem of open-source software for mathematics, science, and engineering. https://www.scipy.org, 2019. Retrieved: 2019-04-15.

[282] H. Robbins and S. Monro. A stochastic approximation method. *Ann. Math. Stat.*, pages 400–407, 1951.

[283] J. Kiefer, J. Wolfowitz, et al. Stochastic estimation of the maximum of a regression function. *Ann. Math. Stat.*, 23(3):462–466, 1952.

[284] M. K. Transtrum and J. P. Sethna. Geodesic acceleration and the small-curvature approximation for nonlinear least squares. *arXiv preprint arXiv:1207.4999*, 2012.

[285] M. K. Transtrum and J. P. Sethna. Improvements to the Levenberg–Marquardt algorithm for nonlinear least-squares minimization. *arXiv preprint arXiv:1201.5885*, 2012.

[286] Mpi for python package. https://mpi4py.readthedocs.io, 2019. Retrieved: 2019-04-15.

[287] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. *2015 IEEE International Conference on Computer Vision (ICCV)*, Dec 2015.

[288] D. Gabor. Theory of communication. part 1: The analysis of information. *Journal of the Institution of Electrical Engineers-Part III: Radio and Communication Engineering*, 93(26):429–441, 1946.

[289] N. Artrith and J. Behler. High-dimensional neural network potentials for metal surfaces: A prototype study for copper. *Phys. Rev. B*, 85(4):045439, jan 2012.

Appendix A

Thermal expansion using the fluctuation method

The partition function for the isothermal-isobaric (NPT) ensemble is [220]

$$Q(N,p,T) = C\int_0^\infty \mathrm{d}V \int_\Gamma \mathrm{d}\boldsymbol{p}\,\mathrm{d}\boldsymbol{q}\, e^{-\beta(\mathcal{H}+pV)}, \tag{A.1}$$

where C is a normalization constant, $\beta = 1/(k_\mathrm{B}T)$, k_B is the Boltzmann constant, $\mathcal{H}$ is the Hamiltonian, p is pressure, V is volume, and the two integrations are over volume space and phase space Γ. The macroscopic observable associated with a phase function A can be obtained as

$$\langle A\rangle = \frac{C}{Q}\int_0^\infty \mathrm{d}V \int_\Gamma \mathrm{d}\boldsymbol{p}\,\mathrm{d}\boldsymbol{q}\, e^{-\beta(\mathcal{H}+pV)}A = CQ^{-1}\int_{\boldsymbol{x}} \mathrm{d}\boldsymbol{x}\, e^{-\beta(\mathcal{H}+pV)}A, \tag{A.2}$$

where in the last equality, we have rewritten the integration over volume space and phase space as $\int_{\boldsymbol{x}}$ for brevity.

Taking volume V as the phase function A, the derivative of the observable $\langle V\rangle$ with

respect to β is

$$\begin{aligned}
\left.\frac{\partial\langle V\rangle}{\partial\beta}\right|_{N,p} =& \frac{\partial(CQ^{-1})}{\partial\beta}\int_{\boldsymbol{x}} \mathrm{d}\boldsymbol{x}\, e^{-\beta(\mathcal{H}+pV)}V + CQ^{-1}\frac{\partial\int_{\boldsymbol{x}}\mathrm{d}\boldsymbol{x}\,\exp\left[-\beta(\mathcal{H}+pV)\right]V}{\partial\beta} \\
=& -CQ^{-2}\frac{\partial Q}{\partial\beta}\int_{\boldsymbol{x}} \mathrm{d}\boldsymbol{x}\, e^{-\beta(\mathcal{H}+pV)}V + CQ^{-1}\int_{\boldsymbol{x}}\mathrm{d}\boldsymbol{x}\,\frac{\partial\exp\left[-\beta(\mathcal{H}+pV)\right]}{\partial\beta}V \\
=& C^2Q^{-2}\int_{\boldsymbol{x}} \mathrm{d}\boldsymbol{x}\, e^{-\beta(\mathcal{H}+pV)}(\mathcal{H}+pV)\int_{\boldsymbol{x}} \mathrm{d}\boldsymbol{x}\, e^{-\beta(\mathcal{H}+pV)}V \\
& - CQ^{-1}\int_{\boldsymbol{x}} \mathrm{d}\boldsymbol{x}\, e^{-\beta(\mathcal{H}+pV)}(\mathcal{H}+pV)V \\
=& \left[CQ^{-1}\int_{\boldsymbol{x}} \mathrm{d}\boldsymbol{x}\, e^{-\beta(\mathcal{H}+pV)}(\mathcal{H}+pV)\right]\left[CQ^{-1}\int_{\boldsymbol{x}} \mathrm{d}\boldsymbol{x}\, e^{-\beta(\mathcal{H}+pV)}V\right] \\
& - CQ^{-1}\int_{\boldsymbol{x}} \mathrm{d}\boldsymbol{x}\, e^{-\beta(\mathcal{H}+pV)}(\mathcal{H}V+pV^2) \\
=& (\langle\mathcal{H}\rangle+\langle pV\rangle)\langle V\rangle - \langle\mathcal{H}V\rangle - \langle pV^2\rangle \\
=& \langle\mathcal{H}\rangle\langle V\rangle - \langle\mathcal{H}V\rangle + \langle pV\rangle\langle V\rangle - \langle pV^2\rangle,
\end{aligned} \tag{A.3}$$

where in the third equality, we used $\partial Q/\partial\beta = -C\int_{\boldsymbol{x}}\mathrm{d}\boldsymbol{x}\, e^{-\beta(\mathcal{H}+pV)}(\mathcal{H}+pV)$, and in the second to last equality, we used equation (A.2). The volumetric thermal expansion coefficient is,

$$\alpha_V = \frac{1}{V}\left.\frac{\partial V}{\partial T}\right|_{N,p} = \frac{1}{\langle V\rangle}\left.\frac{\partial\langle V\rangle}{\partial\beta}\right|_{N,p}\frac{\partial\beta}{\partial T} = -k_{\mathrm{B}}\beta^2\frac{1}{\langle V\rangle}\left.\frac{\partial\langle V\rangle}{\partial\beta}\right|_{N,p}. \tag{A.4}$$

At $p=0$, plugging equation (A.3) into equation (A.4), we obtain

$$\alpha_V = k_{\mathrm{B}}\beta^2\frac{1}{\langle V\rangle}\left[\langle\mathcal{H}V\rangle - \langle\mathcal{H}\rangle\langle V\rangle\right]. \tag{A.5}$$

It is seen that the volumetric thermal expansion coefficient α_V is related to the covariance of the Hamiltonian $\mathcal{H}$ and the volume V.

Next, we get the linear thermal expansion coefficient (LTEC) α_L from α_V. For a two-dimensional (2D) material (e.g. MoS_2), assume $V = hL^2$, where L is the in-place dimension, and h is the out-of-place dimension independent of L. Then the LTEC α_L is

$$\alpha_L = \frac{1}{L}\left.\frac{\partial L}{\partial T}\right|_{N,p} = \left(\frac{V}{h}\right)^{-1/2}\left.\frac{\partial L}{\partial V}\frac{\partial V}{\partial T}\right|_{N,p} = \frac{1}{2}\frac{1}{V}\left.\frac{\partial V}{\partial T}\right|_{N,p} = \frac{1}{2}\alpha_V. \tag{A.6}$$

Appendix B

Hyperparameters in symmetry functions

The symmetry functions are used as the atomic environment descriptor for the neural network interatomic potentials (IPs) discussed in sections 3.3 and 4.3. The hyperparameters η and R_s in equation (3.3) and ζ, λ, and η in equation (3.5) are listed in tables B.1 and B.2, respectively. The hyperparameters used here are taken from Ref. [289].

Table B.1: Hyperparameters in the radical descriptor G_i^2.

No.	η (Bohr^{-2})	R_s (Bohr)
1	0.001	0
2	0.01	0
3	0.02	0
4	0.035	0
5	0.06	0
6	0.1	0
7	0.2	0
8	0.4	0

Table B.2: Hyperparameters in the angular descriptor G_i^4.

No.	ζ	λ	η (Bohr^{-2})	No.	ζ	λ	η (Bohr^{-2})
1	1	−1	0.0001	23	2	−1	0.025
2	1	1	0.0001	24	2	1	0.025
3	2	−1	0.0001	25	4	−1	0.025
4	2	1	0.0001	26	4	1	0.025
5	1	−1	0.003	27	16	−1	0.025
6	1	1	0.003	28	16	1	0.025
7	2	−1	0.003	29	1	−1	0.045
8	2	1	0.003	30	1	1	0.045
9	1	−1	0.008	31	2	−1	0.045
10	1	1	0.008	32	2	1	0.045
11	2	−1	0.008	33	4	−1	0.045
12	2	1	0.008	34	4	1	0.045
13	1	−1	0.015	35	16	−1	0.045
14	1	1	0.015	36	16	1	0.045
15	2	−1	0.015	37	1	−1	0.08
16	2	1	0.015	38	1	1	0.08
17	4	−1	0.015	39	2	−1	0.08
18	4	1	0.015	40	2	1	0.08
19	16	−1	0.015	41	4	−1	0.08
20	16	1	0.015	42	4	1	0.08
21	1	−1	0.025	43	16	1	0.08
22	1	1	0.025				

Appendix C

Fisher information for Gaussian likelihood

For a Gaussian likelihood (i.e. equation (2.12)),

$$p(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{\theta}) = \mathcal{N}(\boldsymbol{y}\,|\,\boldsymbol{h}(\boldsymbol{x};\boldsymbol{\theta}),\boldsymbol{\Sigma}) = \frac{1}{\sqrt{(2\pi)^M|\boldsymbol{\Sigma}|}}\exp\left[-\frac{1}{2}(\boldsymbol{y}-\boldsymbol{h})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\boldsymbol{y}-\boldsymbol{h})\right], \quad \text{(C.1)}$$

the score is

$$\begin{aligned}\boldsymbol{s} &= \frac{\partial \ln p(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{\theta})}{\partial\boldsymbol{\theta}} \\ &= \frac{\partial}{\partial\boldsymbol{\theta}}\left[\ln\left(\frac{1}{\sqrt{(2\pi)^M|\boldsymbol{\Sigma}|}}\right) - \frac{1}{2}(\boldsymbol{y}-\boldsymbol{h})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}(\boldsymbol{y}-\boldsymbol{h})\right] \\ &= (\boldsymbol{y}-\boldsymbol{h})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\frac{\partial\boldsymbol{h}}{\partial\boldsymbol{\theta}},\end{aligned} \quad \text{(C.2)}$$

where the last step takes advantage of the fact that $\boldsymbol{\Sigma}^{-1}$ is symmetric and it is independent of $\boldsymbol{\theta}$.

The Fisher information is,

$$\begin{aligned}\mathcal{I}(\boldsymbol{\theta}) &= -\mathbb{E}_{\boldsymbol{y}}\left[\frac{\partial^2 \ln p(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{\theta})}{\partial\boldsymbol{\theta}\partial\boldsymbol{\theta}}\right] \\ &= -\mathbb{E}_{\boldsymbol{y}}\left[\frac{\partial}{\partial\boldsymbol{\theta}}\left(\frac{\partial \ln p(\boldsymbol{y}|\boldsymbol{x},\boldsymbol{\theta})}{\partial\boldsymbol{\theta}}\right)\right] \\ &= -\mathbb{E}_{\boldsymbol{y}}\left[\frac{\partial}{\partial\boldsymbol{\theta}}\left((\boldsymbol{y}-\boldsymbol{h})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\frac{\partial\boldsymbol{h}}{\partial\boldsymbol{\theta}}\right)\right]\end{aligned}$$

$$= -\mathbb{E}_{\boldsymbol{y}}\left[(\boldsymbol{y}-\boldsymbol{h})^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\frac{\partial^2\boldsymbol{h}}{\partial\boldsymbol{\theta}^2} - \left(\frac{\partial\boldsymbol{h}}{\partial\boldsymbol{\theta}}\right)^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\frac{\partial\boldsymbol{h}}{\partial\boldsymbol{\theta}}\right]$$

$$= -\mathbb{E}_{\boldsymbol{y}}\left[(\boldsymbol{y}-\boldsymbol{h})^{\mathrm{T}}\right]\boldsymbol{\Sigma}^{-1}\frac{\partial^2\boldsymbol{h}}{\partial\boldsymbol{\theta}^2} + \mathbb{E}_{\boldsymbol{y}}\left[\left(\frac{\partial\boldsymbol{h}}{\partial\boldsymbol{\theta}}\right)^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\frac{\partial\boldsymbol{h}}{\partial\boldsymbol{\theta}}\right] \tag{C.3}$$

$$= \mathbb{E}_{\boldsymbol{y}}\left[\left(\frac{\partial\boldsymbol{h}}{\partial\boldsymbol{\theta}}\right)^{\mathrm{T}}\boldsymbol{\Sigma}^{-1}\frac{\partial\boldsymbol{h}}{\partial\boldsymbol{\theta}}\right]. \tag{C.4}$$

From equation (C.3) to equation (C.4), we use the fact that $\mathbb{E}_{\boldsymbol{y}}[\boldsymbol{y}-\boldsymbol{h}] = \mathbf{0}$, since $\boldsymbol{y}$ is a Gaussian distribution with $\boldsymbol{h}$ as its mean, i.e. $\boldsymbol{y} \sim \mathcal{N}(\boldsymbol{y}\,|\,\boldsymbol{h}(\boldsymbol{x};\boldsymbol{\theta}),\boldsymbol{\Sigma})$ as can be seen from equation (C.1).

Appendix D

Dataset for the neural network potential

The data set consists of energies and forces for pristine and defected monolayer graphene, bilayer graphene, and graphite at various states. The configurations in the data set are generated in two ways: (1) crystals with distortions (compression and stretch of cells and random perturbations of atoms), and (2) configurations drawn from *ab initio* molecular dynamics (AIMD) trajectories at temperatures 300, 900, and 1500 K. A detailed summary of the data set is listed below.

For monolayer graphene, the configurations include:

- pristine
  - In-plane compressed and stretched monolayers
  - AIMD trajectories
- defected
  - Relaxation of monolayer with a single vacancy
  - AIMD trajectories of monolayers with a single vacancy

For bilayer graphene, the configurations include:

- pristine

  - AB-stacked bilayers with compression and stretch in the basal plane
  - Bilayers with different translational registry (e.g. AA, AB, and SP) at various layer separations
  - Twisted bilayers with different twisting angles at various layer separations
  - AIMD trajectories of twisted bilayers and bilayers in AB and AA stackings

- defected
  - Relaxation of bilayer with a single vacancy in each layer
  - AIMD trajectories of bilayer with a single vacancy in one layer and the other layer is pristine
  - AIMD trajectories of bilayer with a single vacancy in each layer; Initial configuration without interlayer bonds
  - AIMD trajectories of bilayer with a single vacancy in each layer; Initial configuration with interlayer bonds formed

For graphite, the configurations include:

- pristine
  - Graphite with compression and stretch in the basal plane
  - Graphite with compression and stretch in the c-axis
  - AIMD trajectories

Appendix E

Sensitivity analysis in logarithmic parameter space

Defining $\tilde{\theta}_i = \log \theta_i$, we have

$$\partial \boldsymbol{f} / \partial \tilde{\theta}_i = \partial \boldsymbol{f} / \partial \theta_i \cdot \partial \theta_i / \partial \tilde{\theta}_i = \theta_i \partial \boldsymbol{f} / \partial \theta_i. \tag{E.1}$$

The FIM in logarithm parameter space is

$$\begin{aligned} \tilde{F}_{ij}(\tilde{\boldsymbol{\theta}}) &= \frac{1}{2k_{\mathrm{B}}T\eta} \mathbb{E}_{\mathrm{eq}} \left[\frac{\partial \boldsymbol{f}(\boldsymbol{r};\boldsymbol{\theta})}{\partial \tilde{\theta}_i} \cdot \frac{\partial \boldsymbol{f}(\boldsymbol{r};\boldsymbol{\theta})}{\partial \tilde{\theta}_j} \right] \\ &= \theta_i \left(\frac{1}{2k_{\mathrm{B}}T\eta} \mathbb{E}_{\mathrm{eq}} \left[\frac{\partial \boldsymbol{f}(\boldsymbol{r};\boldsymbol{\theta})}{\partial \theta_i} \cdot \frac{\partial \boldsymbol{f}(\boldsymbol{r};\boldsymbol{\theta})}{\partial \theta_j} \right] \right) \theta_j \\ &= \theta_i F_{ij} \theta_j, \end{aligned} \tag{E.2}$$

where in the second and third equality, we used equation (E.1) and equation (4.28), respectively.

For an unbiased estimator $\hat{\boldsymbol{\theta}}$ of the model parameters, the Cramér-Rao bound is expressed as

$$\mathrm{Cov}_{\boldsymbol{\theta}}[\hat{\boldsymbol{\theta}}] \geq \boldsymbol{F}^{-1}(\boldsymbol{\theta}). \tag{E.3}$$

Let $\boldsymbol{D} = \mathrm{diag}(\boldsymbol{\theta})$ be the diagonal matrix generated from vector $\boldsymbol{\theta}$. Pre-multiplying both

sides of equation (E.3) by $\boldsymbol{D}^{-1}$ and post-multiplying by $\boldsymbol{D}^{-\mathrm{T}}$ gives,

$$\begin{aligned}\boldsymbol{D}^{-1}\mathrm{Cov}_{\boldsymbol{\theta}}[\hat{\boldsymbol{\theta}}]\boldsymbol{D}^{-\mathrm{T}} &= \boldsymbol{D}^{-1}\mathbb{E}\left[(\hat{\boldsymbol{\theta}}-\mathbb{E}[\hat{\boldsymbol{\theta}}])(\hat{\boldsymbol{\theta}}-\mathbb{E}[\hat{\boldsymbol{\theta}}])^{\mathrm{T}}\right]\boldsymbol{D}^{-\mathrm{T}} \\ &= \mathbb{E}\left[\boldsymbol{D}^{-1}(\hat{\boldsymbol{\theta}}-\mathbb{E}[\hat{\boldsymbol{\theta}}])(\hat{\boldsymbol{\theta}}-\mathbb{E}[\hat{\boldsymbol{\theta}}])^{\mathrm{T}}\boldsymbol{D}^{-\mathrm{T}}\right] \\ &= \mathbb{E}\left[\left(\boldsymbol{D}^{-1}(\hat{\boldsymbol{\theta}}-\mathbb{E}[\hat{\boldsymbol{\theta}}])\right)\left(\boldsymbol{D}^{-1}(\hat{\boldsymbol{\theta}}-\mathbb{E}[\hat{\boldsymbol{\theta}}])\right)^{\mathrm{T}}\right] \\ &= \mathrm{Cov}_{\boldsymbol{\theta}}[\boldsymbol{D}^{-1}\hat{\boldsymbol{\theta}}],\end{aligned} \tag{E.4}$$

and

$$\boldsymbol{D}^{-1}\boldsymbol{F}^{-1}(\boldsymbol{\theta})\boldsymbol{D}^{-\mathrm{T}} = \left(\boldsymbol{D}^{\mathrm{T}}\boldsymbol{F}(\boldsymbol{\theta})\boldsymbol{D}\right)^{-1} = \tilde{\boldsymbol{F}}(\tilde{\boldsymbol{\theta}})^{-1}, \tag{E.5}$$

where we used equation (E.2).

Because $\boldsymbol{D}$ is a non-negative matrix (i.e. each element of $\boldsymbol{D}$ is non-negative), the inequality equation (E.3) still holds when pre-multiplied by D^{-1} and post-multiplied by $\boldsymbol{D}^{-\mathrm{T}}$. Therefore

$$\boldsymbol{D}^{-1}\mathrm{Cov}_{\boldsymbol{\theta}}[\hat{\boldsymbol{\theta}}]\boldsymbol{D}^{-\mathrm{T}} \geq \boldsymbol{D}^{-1}\boldsymbol{F}^{-1}(\boldsymbol{\theta})\boldsymbol{D}^{-\mathrm{T}}, \tag{E.6}$$

and using equation (E.4) and equation (E.5), we have

$$\mathrm{Cov}_{\boldsymbol{\theta}}[\boldsymbol{D}^{-1}\hat{\boldsymbol{\theta}}] \geq \tilde{\boldsymbol{F}}(\tilde{\boldsymbol{\theta}})^{-1}. \tag{E.7}$$

The ith diagonal element of this inequality is

$$\mathrm{Var}_{\boldsymbol{\theta}}[\hat{\theta}_i/\theta_i] \geq [\tilde{\boldsymbol{F}}^{-1}]_{ii}. \tag{E.8}$$

Appendix F

Stress in matrix form

The potential part of the virial stress can be written in a matrix form:

$$\boldsymbol{s} = \frac{1}{VT}\boldsymbol{R}\boldsymbol{f}. \tag{F.1}$$

In Voigt notation, the stress $\boldsymbol{s}$ is a column vector with 6 elements:

$$\boldsymbol{s} = \left[s_{11}, s_{22}, s_{33}, s_{23}, s_{13}, s_{12}\right]^{\mathrm{T}}, \tag{F.2}$$

where $s_{11}, \ldots, s_{12}$ are the stress components in the full matrix notation. Given $r_{i,t}^{\alpha}$, where $i = 1, 2, 3$, $\alpha = 1, \ldots, N$, and $t = 1, \ldots, T$, the coordinate matrix $\boldsymbol{R}$ (of size $6 \times 3NT$) can be constructed as follows:

- $r_{i,t}^{\alpha} \mapsto R_{1,3N(t-1)+3\alpha-2}$ for $i = 1$,
- $r_{i,t}^{\alpha} \mapsto \begin{cases} R_{2,3N(t-1)+3\alpha-1} \\ R_{6,3N(t-1)+3\alpha-2} \end{cases}$ for $i = 2$,
- $r_{i,t}^{\alpha} \mapsto \begin{cases} R_{3,3N(t-1)+3\alpha} \\ R_{4,3N(t-1)+3\alpha-1} \\ R_{5,3N(t-1)+3\alpha-2} \end{cases}$ for $i = 3$,

where $r_{i,t}^{\alpha} \mapsto R_{p,q}$ means setting $R_{p,q}$ to $r_{i,t}^{\alpha}$. The components of $\boldsymbol{R}$ that are not set above take a value of 0. The force vector $\boldsymbol{f}$ (of size $3NT$) can be constructed by the

mapping: $f_{i,t}^{\alpha} \mapsto \boldsymbol{f}_{3N(t-1)+3\alpha-2}$.

For example, assume we have a system of $N = 2$ atoms running a total number of $T = 2$ steps, then the coordinate matrix $\boldsymbol{R}$ and the force vector $\boldsymbol{f}$ are

$$\boldsymbol{R} = \begin{bmatrix} r_{1,1}^1 & 0 & 0 & r_{1,1}^2 & 0 & 0 & r_{1,1}^1 & 0 & 0 & r_{1,1}^2 & 0 & 0 \\ 0 & r_{2,1}^1 & 0 & 0 & r_{2,1}^2 & 0 & 0 & r_{2,1}^1 & 0 & 0 & r_{2,1}^2 & 0 \\ 0 & 0 & r_{3,1}^1 & 0 & 0 & r_{3,1}^2 & 0 & 0 & r_{3,1}^1 & 0 & 0 & r_{3,1}^2 \\ 0 & r_{3,1}^1 & 0 & 0 & r_{3,1}^2 & 0 & 0 & r_{3,1}^1 & 0 & 0 & r_{3,1}^2 & 0 \\ r_{3,1}^1 & 0 & 0 & r_{3,1}^2 & 0 & 0 & r_{3,1}^1 & 0 & 0 & r_{3,1}^2 & 0 & 0 \\ r_{2,1}^1 & 0 & 0 & r_{2,1}^2 & 0 & 0 & r_{2,1}^1 & 0 & 0 & r_{2,1}^2 & 0 & 0 \end{bmatrix} \tag{F.3}$$

and

$$\boldsymbol{f} = \left[f_{1,1}^1, f_{2,1}^1, f_{3,1}^1, f_{1,1}^2, f_{2,1}^2, f_{3,1}^2, f_{1,2}^1, f_{2,2}^1, f_{3,2}^1, f_{1,2}^2, f_{2,2}^2, f_{3,2}^2 \right]^{\mathrm{T}}, \tag{F.4}$$

respectively.