Gaussian process regression for nonparametric force fields

Zeni, Claudio

Awarding institution:
King's College London

Gaussian Process Regression for Nonparametric Force Fields

Claudio Zeni

A dissertation submitted in partial fulfilment

of the requirements for the degree of

Doctor of Philosophy

of

King's College London.

Department of Physics

King's College London

May 10, 2020

If people did not sometimes do silly things,

nothing intelligent would ever get done.

Ludwig Wittgenstein

I, Claudio Zeni, confirm that the work presented in this thesis is my own. Where information has been derived from other sources, this has been indicated in the work.

I, Claudio Zeni, furthermore declare to have contributed to the following research outputs during the years of my doctorate studies:

- Claudio Zeni, Aldo Glielmo, Ádám Fekete, Henry Lambert, Francesca Baletto. MFF: a Python package for the automatic construction of extremely fast nonparametric force fields. *In writing*.
- Claudio Zeni, Theodoros Pavloudis, Francesca Baletto, Richard Palmer. Melting Au nanoparticles using nonparametric machine learning force fields. *In preparation*.
- Robert M. Ziolek, Joel Nulsen, Claudio Zeni, Franca Fraternali, Chiara Cammarota, Chris D. Lorenz. Bespoke implicit solvent models for soft matter molecular dynamics simulations. *In preparation*.
- Claudio Zeni, Ádám Fekete, Aldo Glielmo. Machine learning nonparametric force fields (MFF) Python package. *kcl-tscm/mff v1.0* `http://doi.org/10.5281/zenodo.1475959`, 2019.
- Claudio Zeni, Kevin Rossi, Aldo Glielmo, and Francesca Baletto. On machine learning force fields for metallic nanoparticles. *Advances in Physics: X*, 4(1):1654919, 2019.
- Claudio Zeni, Kevin Rossi, Aldo Glielmo, Ádám Fekete, Nicola Gaston, Francesca Baletto and Alessandro De Vita. Building Machine Learning Force Fields for Nanoparticles. *Journal of Chemical Physics* 148, 241739, 2018.
- Aldo Glielmo, Claudio Zeni, Ádám Fekete, Alessandro De Vita. Building nonparametric n-body force fields using Gaussian process regression. In: *Ma-*

chine Learning for Quantum Simulations of Molecules and Materials, 1st ed. (accepted manuscript), 2019.

- Aldo Glielmo, Claudio Zeni, and Alessandro De Vita. Efficient nonparametric *n*-body force fields from machine learning. *Physical Review B*, 97(18):1 12, 2018.

Abstract

The recent years have seen a surge in the development of machine learning algorithms in different areas of scientific research. In the field of simulation of materials, the development of machine learning force fields to carry out fast and accurate molecular dynamics simulations has been attracting a lot of interest ever since the early manuscripts of Blank et al. in 1995, Brown et al. in 1996, and the pioneering work of Behler and Parrinello in 2007. Machine learning force fields are trained using reference data coming from expensive *ab initio* simulations and try to approximate these accurate methods without recurring to any *ad hoc* fitting procedure in a computationally efficient way. In this thesis, we present the work done on the development of algorithms that employ Gaussian process regression to build machine learning force fields. We specifically design Gaussian process force fields that use explicitly 2-body, 3-body, and simplified many-body descriptors of local atomic environments. Furthermore, we develop an algorithm to map such Gaussian process force fields into nonparametric classical force fields. This rather general "mapping" procedure removes the inefficient computational scaling of Gaussian process regression methods and yields, without meaningful accuracy losses, force fields that are as fast as classical parametric force fields. All the algorithms and numerical procedures discussed in this thesis are available as a Python package, named "MFF", which I have coauthored. This package is freely available at `https://github.com/kcl-tscm/mff`, and fully documented. To benchmark the speed and accuracy of the MFF package, we test it on bulk metals (Fe, Ni) and semiconductors (C, Si). We also address the problem of developing machine learning force fields for metallic nanoparticles such as Ni, Au and AgAu. Nanoparticles display very complex energetic landscapes, and accurate force fields that are not fitted on bulk properties are highly desirable to predict structural transitions and phase changes. We build force fields for a set of five isomers of Ni_{19}, and carry out classical molecular dynamics simulations for a total of $\sim$ 200 ns, a time scale not reachable via *ab initio* methods, but indeed easily accessible using our mapped machine learning force fields. Subsequently, we discuss the development of machine learning force fields that are accurate for nanoparticles with varying numbers of atoms and analyse small Ni nanoparticles containing 13 to 20 atoms, and larger Au nanoparticles containing 147, 309 and 561 atoms. For the smaller Ni nanoparticles, machine learning force fields are not transferable between particle sizes, reinforcing the belief that "every atom counts" in small nanoparticles. For larger Au nanoparticles, force fields trained on Au_{147} data well predict forces in the two bigger Au nanoparticles; this result paves the way towards the development of machine learning force fields which are accurate for nanoparticles that contain too many atoms to be effectively simulated using quantum methods.

Acknowledgements

I would like to dedicate this thesis work to Alessandro “Sandro” De Vita, the person that most influenced my scientific development in these past years. Sandro, or professor De Vita as I had met him seven years ago, was first a teacher, then a thesis advisor, finally a Ph.D. supervisor, and always a role model to me. His materials science lectures were the reason I moved progressively from engineering towards condensed matter physics. The master’s thesis I wrote with him was the reason I started a Ph.D. in Physics. His brilliance and positive attitude heavily contributed to this dissertation thesis.
I owe him much of my scientific research, and the work contained in this work would not have been possible without the guidance and support he offered me until the end. He was a great scientist and a great man, and I hope that the legacy he left behind, in the form of community, collaborations, and friendships, will endure for a long time.

I thank all the people that supported me during these years, which, even if not without troubles, have been very fun, extremely interesting, and full of new friends. Special acknowledgements go to the people in the Physics department of King's College London, and to the people behind the center for doctoral training Cross disciplinary Approaches to Non Equilibrium Systems (CANES), which created an inspiring working environment during these years. Specifically, I would like to thank Francesca Baletto for accepting the hard task of becoming my supervisor midway through my Ph.D., for all the resulting support, and the good work we have done together. Thanks also to Nicola Bonini, for the support shown and for discussions that were always pleasant and very interesting. Special kudos go to Aldo Glielmo, a friend and brilliant scientist, and my closest collaborator these past years. In the same vein, I would also like to give a tip of my (imaginary) hat to Kevin Rossi and Ádám Fekete, for all the meaningful discussions, the friendship, and the fruitful collaborations. I am extremely grateful to Luca Ghiringhelli and Chris Pickard for taking the time to read my thesis thoroughly, for their precious advice on how to improve it, and for the interesting and pleasant discussion we had during the defence. I moreover thank (again) Francesca Baletto, Kevin Rossi, Henry Lambert, and Theodoros Pavloudis, for providing the essential reference quantum datasets employed in this thesis work. I acknowledge the financial support I have received during the doctorate years, specifically from the Engineering and Physical Sciences Research Council (EPSRC) (Grant No. EP/L015854/1) and the Office of Naval Research Global (Award No. N62909-15-1-N079). Now that I am done thanking the people that I worked with, I will acknowledge the people I definitely did not do any work with.

First and foremost, thanks to Roberta, for the endless love and the continuous encouragement, which distance made harder to give and for this reason more precious. I would also like to thank my mother Antonella and my father Maurizio, for teaching me the value of hard work, for supporting my every decision in life, and for accepting that they will never really know what the devil I have been doing these past three years.

Thanks, Alessandro C., for the corroborating coffee breaks and his obsessive use of bash scripting, which helped me love Python even more. And I will not forget to acknowledge Jason, for really too many reasons; the use of tinny and woody words just to name one. Thanks also to Carla, Evan, Francesco M., Giulia, Irene, Stefano and Yurema for making these years spent in London so much better with your presence. I am also grateful to Alberto, David, Matej, Simone, Boris, Francesco L., Furquaan, Jack, and Michela for their friendship and the fun times spent together sharing a common passion. A thank you also to Gioia, Vito and Yanick, who had to daily withstand my annoying physical presence.

Contents

Abbreviations

FF	Force field
GP	Gaussian process
GPR	Gaussian process regression
MD	Molecular dynamics
ML	Machine learning
AIMD	Ab initio molecular dynamics
CMD	Classical molecular dynamics
DFT	Density functional theory
ANN	Artificial neural network
P-FF	Parametrized force field
ML-FF	Machine learning force field
ANN-FF	Artificial neural network force field
GP-FF	Gaussian process force field
M-FF	Mapped force field
MFF	“MFF” Python package
EAM	Embedded atoms method
MAE	Mean absolute error
PDF	Pair-distance distribution function
BADF	Bond angle distribution function
RMBF	Eoot mean bond fluctuation
NP	Nanoparticle
MNP	Metallic nanoparticle
PBE	Perdew–Burke–Ernzerhof

Symbols

a	Scalar
$\mathbf{a}$	Vector
$\mathbf{A}$	Matrix or high-dimensional tensor
S	System of atoms
$\mathbf{R}$	Cartesian coordinates of all atoms in a system
$\mathbf{r}$	Cartesian coordinates of an atom
$\mathbf{z}$	Atomic numbers of all atoms in a system
z	Atomic number of an atom
t	Time
T	Temperature
E	Total energy
ε	Local energy contribution
$\mathbf{f}$	Force vector
$\mathscr{D}$	Database
D	Size of database
N	Number of atoms in a system
ρ	Local atomic environment
$\mathbf{q}$	Local atomic descriptor
$\mathbf{P}$	Set of local atomic environments
$\mathbf{Q}$	Set of local atomic environment descriptors
M	Number of atoms in a local atomic environment
r_c	Cut-off radius
r_{ij}	Distance between atoms i and j

k	Local energy-local energy kernel function
$\mathbb{K}$	Gram matrix
σ	Lengthscale hyperparameter
λ	Noise hyperparameter
r_0	Distance hyperparameter

List of Figures

List of Tables

Chapter 1

Introduction

Numerical simulations have become an essential tool for modern sciences. In the context of materials science, they can be used to model materials' behaviour and predict thermal, mechanical, diffusive, catalytic, chemo-mechanical properties and many more, without resorting to expensive and complex experiments [1, 2]. The temporal and spatial scale such simulations can cover varies greatly, going from quantum simulations comprising a few atoms on the ps time scale, to finite element simulations of structures on the meter scale across several hours [3]. Computer simulation of matter via the explicit modelling of the dynamical behaviour of the atoms that compose it takes the name of molecular dynamics (MD), a technique that is nowadays widespread and sees applications in the world of biology, atmospheric science, chemistry and materials science.

In Sec. 1.1, we discuss the two most common ways to compute atomic forces in MD simulations: classical parametrized force fields and *ab initio* quantum methods. Subsequently, in Sec. 1.2, we present the case for the use of machine learning (ML) algorithms to fit FFs on reference quantum data, to bridge the existing speed and accuracy gap between the two methods presented in Sec. 1.1. Finally, in Sec. 1.3, we discuss the topic of MD simulations for metallic nanoparticles (MNPs), and remark the importance and challenge of developing reliable and computationally inexpensive FFs for these systems.

1.1 MD Simulations

Since the first MD simulations, carried out using early computers by Gibson et al. [4] and Rahman [5], the technique has grown in popularity and effectiveness, and is without doubt one of the main tools used by researchers in materials science, chemistry, and biology. In MD simulations, the temporal evolution of the system is modelled via numerical integration of Newton's second law, and relies on an accurate model to describe the forces acting on the atoms, which are calculated "on the fly" at every discrete time step. Ideally, forces and energies are computed from first principles by solving the electronic structure for a certain configuration of the nuclei. These first principles methods often rely on density functional theory (DFT) [6, 7], and the application of *ab initio* MD (AIMD) has proven to be extremely effective in predicting physical properties of complex systems, ever since the pioneering work carried out by Car and Parrinello in 1985 [8]. Nowadays AIMD can be applied to systems comprised of hundreds of atoms and for timescales of several tens/hundreds of ps [9]. Nonetheless, AIMD remains computationally intensive and is not feasible for simulating larger systems and/or for longer timescales. For large non-periodic systems, multi-scale techniques such as Quantum Mechanics/-Molecular Mechanics (QM/MM) can sometimes be successfully employed [10]. This approach is only possible if the full quantum accuracy is required in a small and well-defined zone of the system, and a simpler description employing classical P-FFs suffices everywhere else. It is often the case, however, that problems require very large minimal system sizes, and times so long that the simulations must employ a higher level of approximation and rely exclusively on classical P-FFs.

Generally, a force field is a set of analytical expressions describing the dependence of the interatomic potential energy of a system on the coordinates of its atoms/molecules. The production of suitably accurate and system-transferable P-FFs is a remarkably difficult challenge. Traditionally, the task involves tuning the parameters of analytic functions trying to ensure that extended reference data coming either from experiments or from quantum calculations coincide with the P-FFs predictions [11, 12]. One of the main issues in developing P-FFs is the choice of a suitable analytic function, which requires a certain level of chemical intuition and patient effort, which is guided by trial and error and is without any certainty of success [13]. However, in the cases where the development of a P-FF is successful, the amount of work done is richly rewarded by the opportunity to use an extremely fast and reliably precise force field model [14, 15, 16, 17]. The parametric analytic functions employed encode valuable physical knowledge on the target system, allowing for a certain degree of human understanding of the processes that govern the atomic behaviour (e.g. bond strength, angular dependencies etc.). Such knowledge is, furthermore, usually applicable to similar systems as an *a priori* working hypothesis of their behaviour. Whenever *ab initio* reference data on the new system become available, this can be directly used to adjust the parameters of the already-existing analytic form of the P-FF to a new set of best-fit values that minimise the error on the *ab initio* data.

1.2 The Reason for Machine Learning Force Fields

Machine learning has recently emerged as a novel approach to the construction of *nonparametric* FFs which are directly derived from reference data and do not comply with a fixed analytic form. This idea founds its inception in the pioneering works by Skinner and Broughton [18], who first proposed the use of ML to reproduce potential energy surfaces predicted by quantum methods, and in the subsequent manuscripts by Blank et al. [19] and Brown et al. [20]. With the growth in computational and storage capabilities, the use of ML to develop FFs is becoming an increasingly attractive option, compared with traditional P-FF development via chemical intuition and hand-tuning of parameters. Indeed, implementations of frameworks to build machine learning force fields (ML-FFs) have been blooming in the last 20 years, first based on artificial neural networks (ANN) [21], then subsequently on linear regression (LR) [22] and Gaussian Process (GP) regression [23]. Current effort in the field is aimed towards increasing the computational efficiency and accuracy of such algorithms [10, 24, 25, 26, 22, 27, 28, 29, 30, 31, 32, 33]. Al-

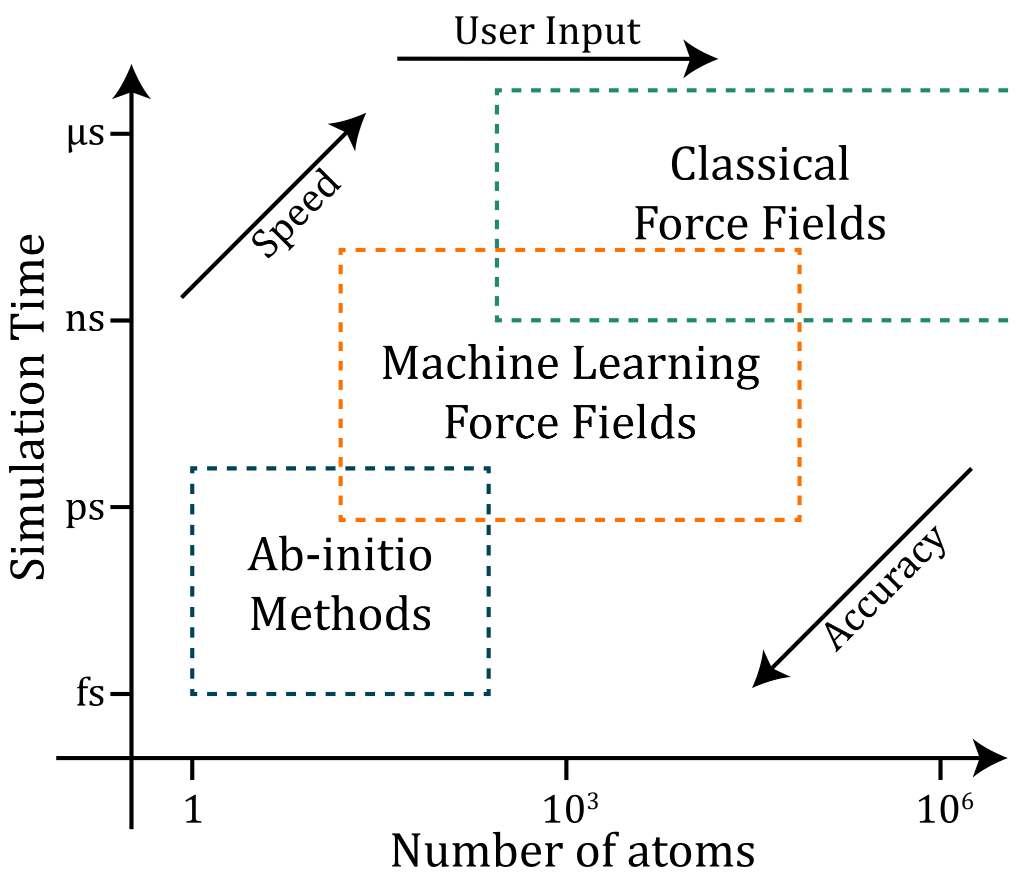


Figure 1.1: A representation of the spatial and temporal scales available to MD simulations carried on using *ab initio* quantum methods (blue), classical P-FFs (green) and novel ML-FFs (orange). Arrows indicate increasing directions for the algorithms' increasing speed and accuracy, and for the amount of user input needed to tune the algorithm.

though the development of ML-FFs ideally requires little user input, and the resulting models have been shown to be capable of high accuracies, ML-FFs have not yet substituted classical P-FFs in many of the applications where it would be expected. This is mainly because *standard* P-FFs are still orders of magnitude faster than their ML counterparts [34]; this is schematically depicted in Fig. 1.1. Furthermore, while P-FFs can be usually written in compact and humanly interpretable analytic forms which are easy to visualize, ML-FFs typically involve complex mathematical expressions, which are not easily readable.

In Chapter 2, we present an ML framework based on GP regression (GPR) that tries to address the aforementioned problems by constructing Gaussian process force fields (GP-FFs) which can be decomposed into explicit n-body terms and are, therefore, easily interpretable. Such decomposition is then used to transform these GP-FFs into tabulated nonparametric classical FFs, where the energies associated to certain arrangements of atoms are stored and then used as lookup reference values for interpolation. The resulting FFs retain the accuracy reachable by ML methods but display a computational speed that is on-par with classical P-FFs. This approach is then incorporated into a Python package, named “MFF” [35]. We describe both its usage and its structure, along with the mapping technique that allows one to transform GP-FFs into classical FFs, in Chapter 3. Speed and accuracy tests on quantum reference data for various materials are also carried out in the same chapter, to display the potentialities and limits of the approach.

1.3 Force Fields for Metallic Nanoparticles

Metallic nanoparticles (MNPs) are objectes with all their three dimensions in the nanoscale regime (less than 100 nm). They can be either monometallic, i.e. containing only one chemical species, or nanoalloys, i.e. made up of two or more metals. MNPs are used in a wide array of fields, from bio-engineering to catalysis to optoelectronics, due to their unique chemical and physical properties which are different both from their bulk and single-atom counterparts, and depend strongly on the MNPs’ architecture, defined as size, shape, chemical composition, and chemical ordering [36]. MNPs often display a complex and diverse configurational space with many local minima, often very close in energy [36, 37, 38]. Given the large dimensionality of MNPs’ configurational space (roughly $\exp(N)$, where N is the number of atoms in the MNP), the estimate of finite-temperature population distribution, and of zero-temperature minima structure of MNPs, requires significant computational effort.

In the past three decades, various methods have been developed to address the task of sampling the energy landscape of MNPs [39], such as Monte Carlo procedures [40], basin hopping [41, 42, 43, 44, 45], hyperspatial optimization [46], and evolutionary algorithms [47, 48, 49, 50, 51]. These methods attempt to solve the problem of finding the (putative) global minimum (i.e. the reference thermodynamically stable geometry) of a structure in a computationally efficient way. Nonetheless, at finite temperature and as a function of the external environment, nanoparticles might undergo various structural rearrangements on a relatively short time scale. This thermodynamical behaviour is not trivial to model and requires a precise estimate of the isomer-population as a function of the system's temperature and pressure. Computational methods like MD, path sampling [52, 53], metadynamics [54, 55, 56, 57, 58], and finite temperature partition function evaluation methods [59, 60, 61, 62, 63] allow for an estimate of such population distributions. The evaluation of converged results, however, requires thousands if not millions of energy and force calculations [57, 61, 64]. This makes the process computationally demanding, as *ab initio* methods (i.e. DFT) are often required to perform the energy and force evaluations due to the lack of, or inadequacy of, classical P-FFs. Machine learning, therefore, emerges as an ideal candidate for the development of accurate and fast FFs to employ in such demanding calculations. In Chapter 4, we address the challenge of developing ML-FFs based on GPR for Ni_{19}. We then transform these ML-FFs into nonparametric classical FFs (called here mapped force fields, M-FF) and employ them to run fast classical MD simulations of the system with an almost-*ab initio* accuracy at temperatures ranging from 300 K to 1200 K, to investigate the thermal behaviour of Ni_{19}. We simulate a variety of Ni_{19} isomers for a total of 180 ns, using a M-FF that incurs low errors on force predictions. The long time scales reached in these MD simulations allow us to predict the presence of a "slush" intermediate state between nanosolid and nanoliquid in the 700-900 K temperature range.

Given the vast amount of energetically relevant MNPs' sizes and compositions, the creation of multiple classical P-FFs, each tuned to be accurate on a restricted subset of sizes, geometries, and chemical compositions, is an extremely resource-intensive task, even for a small amount of chemical species. The opposite approach of adjusting a single existing classical P-FF so that it is accurate for multiple size ranges and morphologies also poses unique challenges, as the manual parameter tuning typically employed in P-FF construction becomes increasingly more complex as the number of reference data and properties to fit increases. Quantum methods circumvent the above-mentioned issues by the virtue of being mostly system-agnostic. Of course, the computational cost of *ab initio* methods is such that only small nanoparticles containing up to a thousand atoms can be modelled. This indeed poses a severe limit to the number of experimentally relevant systems that can be accurately modelled *in silico*, given that NPs can contain up to millions of atoms. In Chapter 5 we investigate the creation of ML-FFs that are accurate across varying ranges of MNPs' sizes and geometries. We first use a database comprised of AIMD simulations for a set of small Ni NPs of sizes 13 to 20 presenting varying geometrical arrangements, and discover that the creation of a 3-body FF that is suitably accurate for most of the nanoparticles' (NPs) sizes is only possible if NPs of different sizes are included in the training set. This result reinforces the belief that at small sizes "every atom counts", and that it is complex to construct FFs that retain their accuracy while working in an extrapolative *cross-size* regime for small NPs. We then scale up the problem and look at Au nanoparticles containing 147, 309, and 561 atoms. In this case, the 3-body GP-FFs are not accurate enough; we, therefore, employ a many-body correction term to the GP-FF, based on a descriptor of our design that is inspired by the embedded atom method (EAM) potentials. This reduces the error incurred by the GP-FF w.r.t. reference calculations, both when the GP-FFs are tested on nanoparticles of the same size as those they were trained on, and when they are instead tested on bigger nanoparticles. We, therefore, show that force predictions on MNPs of a certain size can be quite accurate when employing ML-FFs that have been trained on significantly smaller NPs. This encouraging result shows promise for the creation of ML-FFs for MNPs which contain too many atoms to be computed using quantum methods and, therefore, for which *ab initio* reference data to train an ML-FF directly is not available.

Chapter 2

Machine Learning Force Fields

In this first chapter, we thoroughly discuss the issues at the core of the development of machine learning force fields. Firstly, in Sec. 2.1, we show how this problem can be formally stated as a supervised learning problem. Then, in Sec. 2.2, we deeply discuss the properties of local atomic environment descriptors, including n-body [65] and novel “EAM-like” ones. In Sec. 2.3 we present two supervised ML algorithms: linear regression (LR) and artificial neural networks (ANNs), and briefly review their uses in the field of ML-FFs creation [66]. The final and main section, Sec. 2.4, is dedicated to Gaussian process regression, the supervised learning framework used throughout this work. The algorithm is presented in the context of building ML-FFs; a particular focus is put on the choice of the kernel function, and novel EAM-like kernels are here introduced.

2.1 Framing the Problem

2.1.1 Local Energy Approximation

The generation of a machine learning force field approximating quantum forces and energies can be framed as a supervised learning problem. The objective function that needs to be learnt is here the total energy E, working in the Born-Oppenheimer ground state [67]. Under such assumptions, the energy of a system of N atoms depends simultaneously on their positions $\mathbf{r}_i$, atomic numbers z_i:

$$E(\mathbf{R}) = E(\mathbf{r}_1, \ldots, \mathbf{r}_N, z_1, \ldots, z_N), \tag{2.1}$$

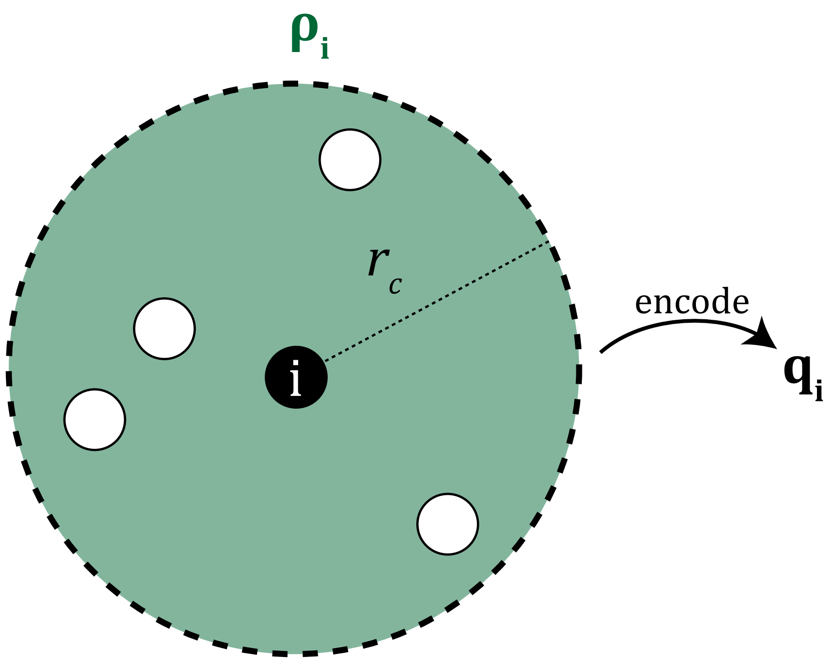


Figure 2.1: Schematic representation of the local atomic environment ρ_i, constructed by considering all atoms less distant than r_c from the central atom i. The information about ρ_i is encoded in a descriptor $\mathbf{q}_i$.

where $\mathbf{R}$ indicates the coordinates and species of all atoms in the system. The total energy can, in theory, be approximated directly, without any further assumption, with machine learning potentials. This approach has been tried in the past [68, 19, 69], but it is computationally inefficient. If the total energy is a function of the atomic coordinates of the entire system, the force acting on an atom is itself dependant on the whole system. Therefore, force evaluations in an FF built this way have a computational cost that scales with the system size (linearly or more than linearly), which is not desirable.

To then obtain an FF where force predictions' computational cost is independent of system size, one possibility is to decompose the total energy into local atomic energy contributions [21]:

$$E(\mathbf{R}) = \sum_{i=1}^{N} \varepsilon(\mathbf{q}_i), \tag{2.2}$$

where ε_i indicates the local energy term associated with atom i and $\mathbf{q}_i$ is a vector encoding information about the local atomic environment ρ_i: the set of atoms j such that $r_{ij} \leq r_c$; this is represented in Fig. 2.1. This "local energy approximation" introducted in Eq. (2.2) is justified by the *near-sightedness* principle of quantum mechanics [70]. The computational cost of force predictions is reduced by the shift from total energy prediction to local energy prediction, as the local energy of atom i depends only on the position and species of atoms within a cutoff r_c from i. Therefore, the force acting on atom i:

$$\mathbf{f}(\mathbf{q}_i) = -\frac{\partial E(\mathbf{R})}{\partial \mathbf{r}_i}, \tag{2.3}$$

can be computed using the local energy of atom i and the local energies of atoms j contained in the local atomic environment ρ_i:

$$\mathbf{f}(\mathbf{q}_i) = -\frac{\partial \varepsilon(\mathbf{q}_i)}{\partial \mathbf{r}_i} - \sum_{j \in \rho_i} \frac{\partial \varepsilon(\mathbf{q}_j)}{\partial \mathbf{r}_i}. \tag{2.4}$$

This simplification stands because the local energy of atoms outside ρ_i does not depend on $\mathbf{r}_i$ under the assumption made in Eq. (2.2). In the cases where the local energy is further decomposed into energies associated to unordered groups of atoms (e.g. pairs, triplets), Eq. (2.4) can be further simplified because of symmetry [65], to obtain:

$$\mathbf{f}(\mathbf{q}_i) = -n\frac{\partial \varepsilon(\mathbf{q}_i)}{\partial \mathbf{r}_i} \tag{2.5}$$

where n is the number of atoms contained in the aforementioned unordered groups, e.g. $n = 2$ for a pair potential.

2.1.2 Long-Range Effects

Due to the inherent locality of the descriptor $\mathbf{q}_i$, long-range effects, such as electrostatic repulsion, are traditionally treated separately from ML-FFs. The incorporation of electrostatic effects in the total energy calculations can be achieved by baselining the ML model with computationally inexpensive methods that include electrostatic interactions [71, 72], by training the ML model on data where the Ewald-like electrostatic contributions were removed [23], by employing a separate ML model that learns the partial charges of the system [73, 74, 75, 76, 77, 78], or by incorporating them in the ML-FF [79]. From now onwards we, therefore, suppose either that the long-range electrostatic interactions are negligible in our target systems, or that they have been treated separately, and the forces and energies used for learning and testing have had their electrostatic components removed.

2.1.3 Creating a Database

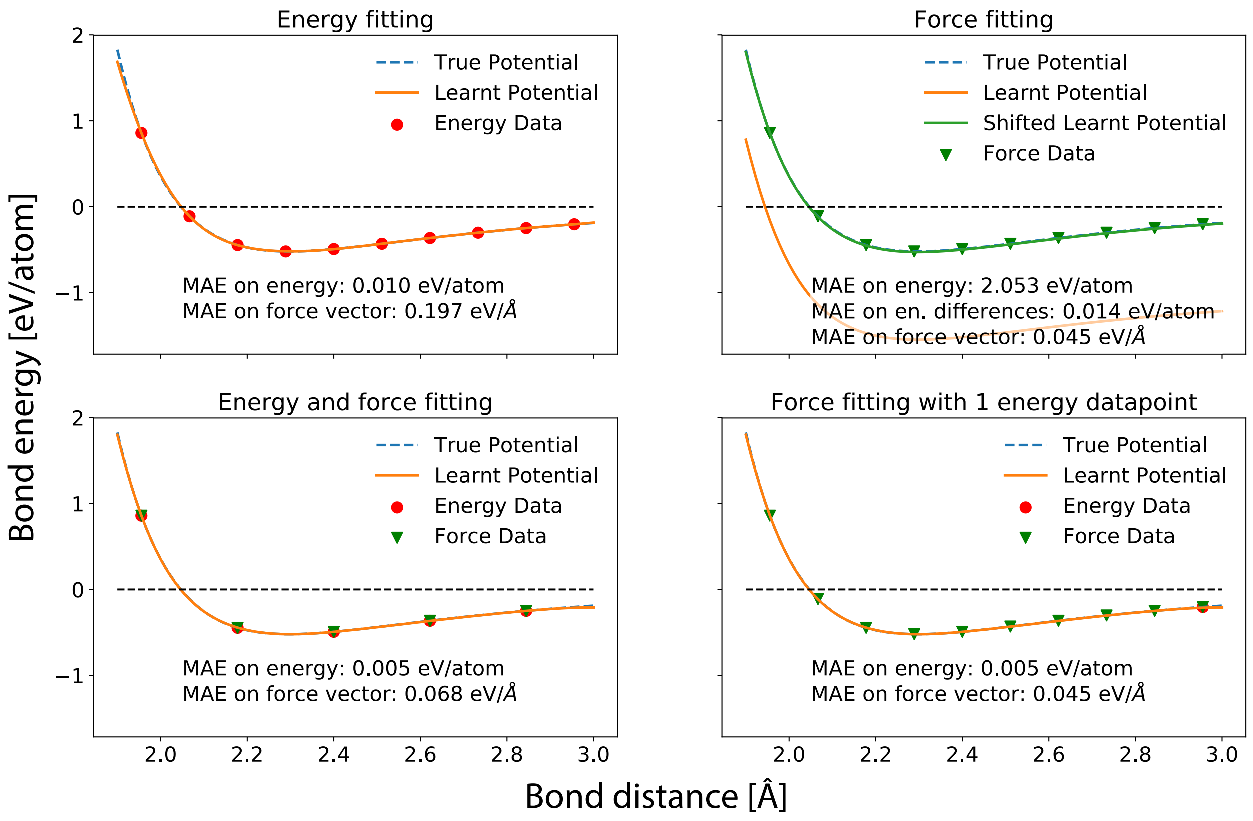


Figure 2.2: Learning of an interatomic Lennard-Jones (LJ) pair potential for Cu via Gaussian regression using 10 data points. The four panels picture the LJ potential to be learnt in blue dashed lines, and the ML potential in orange lines. The ML algorithm was trained on energy data only (top left), force data only (top right), an even split of energy and force data (bottom left), and force data with one energy data point (bottom right). For the fitting on force data only, an adjusted predicted energy curve is also shown in green, where the orange curve was vertically shifted by the error incurred in the energy prediction of a single randomly-chosen value. The training points used are highlighted as green triangles (force data) and red circles (energy data). The mean absolute error (MAE) on the force vector and the bond energy are represented and have been calculated for 100 distance values.

To learn a function, a ML model requires a training database containing reference input-output pairs. In the case of ML-FFs these input-output data points would ideally be local atomic environment descriptor - local energy pairs. However, since local energies are not physical quantities that can be obtained in *ab initio* simulations, a training database for ML-FFs instead typically contains total energies and forces. These quantities can be then used to train local energy functions as demonstrated in Section 2.4. The training database $\mathscr{D}$ containing D training points consists of two separable parts: a total energy dataset $\mathscr{D}_E$ containing D_E points and a local force dataset $\mathscr{D}_f$ containing D_E points, with $D = D_E + D_f$. The total energy dataset contains pairs composed of sets of local atomic environment descriptors $\{\mathbf{q}_i\}_d$ of each atom i in the system d, which is denoted as $\mathbf{Q}_d$, and total energy values E_d:

$$\mathscr{D}_E = \{\mathbf{Q}_d, E_d\}, \;\; d = 1, \ldots, D_E. \tag{2.6}$$

The force dataset instead contains pairs of local atomic environment descriptors - forces acting on the central atom of the local atomic environment:

$$\mathscr{D}_f = \{\mathbf{q}_d, \mathbf{f}_d\}, \;\; d = 1, \ldots, D_f. \tag{2.7}$$

In order to obtain a conservative FF, the cutoff used for the force dataset should be equal to the one used to generate local atomic environments in the energy dataset only if Eq. (2.5) holds validity, otherwise the cutoff for the force dataset should be doubled, as per Eq. (2.4).

It is possible to build a force field starting from energy data only [21, 80], using gradient information to derive the forces [81, 82], or using force data directly [10, 83, 24, 84]. Nonetheless, it could be beneficial to train the algorithm using a number D_f of force training points and a smaller number D_E of total energy training points to incur overall lower errors while predicting forces and total energies. This approach is investigated in Fig. 2.2, where four GP-FFs employing the 2-body kernel discussed in Section 2.4.3 are trained on force and/or total energy data generated using a Lennard-Jones Cu pair potential. The mean absolute error (MAE) for force and energy predictions is reported in Fig. 2.2; this error measure is defined for a scalar or vectorial quantity a as:

$$\mathrm{MAE} = \frac{1}{D_{test}} \sum_{d=1}^{D_{test}} ||a_d - \hat{a}_d||_2, \tag{2.8}$$

where a_d and $\hat{a}_d$ are, respectively, the reference and predicted values (i.e. the total energies E or local forces $\mathbf{f}$), and $|| \bullet ||_2$ denotes the Euclidean norm. For vector quantities, the MAE is therefore the average norm of the difference vector between reference and predicted vectors. It is evident from Fig. 2.2 that including mostly force data and a small number (i.e. one) of energy reference points in the training set yields the lowest mean absolute error (MAE) on energy, without meaningfully affecting the force prediction error w.r.t. sole force training. In the cases where the accurate prediction of quantum forces is most relevant (e.g. when computing dynamical properties), training only on forces results in a more computationally efficient choice w.r.t. training on force and energy data. In particular, we can notice how the energy prediction for the potential trained on force data only can be almost as accurate as the one yielded by an energy-trained potential, once the predictions are shifted by a constant factor. This factor can be easily calculated, for example as the error incurred by the force-fitted potential when evaluating the energy of a single data point.

The training database can be created using any reference method that is deemed accurate for the system, but a dataset must be consistent: all of the data contained has to be generated using the same exact methodology (e.g. the same DFT functional and cutoff energy). This consistency is required since, otherwise, the resulting ML-FF would be trained as an average of the FFs defined by the *ab initio* methods employed, weighted by their relative presence in the training database. In literature, DFT is the most commonly used *ab initio* method to generate reference data, either via AIMD simulations or via sets of single-point energy and force evaluations. In this manuscript, all of the databases used were extracted from AIMD simulations employing DFT, the specifics of the functional and methodology used will be given for each dataset.

2.2 Local Atomic Environment Descriptors

Before diving into the details of the machine learning algorithms used to learn FFs, it is useful to discuss the design and properties of the inputs these algorithms take. These inputs are named local atomic environment descriptors, and their design and formulation directly affect the properties of the learnt FFs. Let us here focus on a system of N atoms of the same chemical species. Generalization to multi-chemical species is, in the case of GP-FFs, straightforward, and it will be formally discussed in Sec 2.4.5.

2.2.1 Descriptor's Properties

An appropriate representation of the local environment ρ_i surrounding an atom i is encoded in a descriptor $\mathbf{q}_i$, which has to possess the following key properties. First, it is best if descriptors are invariant to rigid translations, rotations, reflections, and permutations of same-species atoms. This is required to strictly impose these fundamental physical invariances upon the learnt FF. Invariance properties could be learned automatically by any sufficiently flexible algorithm when provided with enough data. Nonetheless, their strict imposition through an invariant representation is found to be extremely beneficial both for the transferability and to the learning speed of the FF [83, 25, 24, 10]. Secondly, the descriptor must be differentiable w.r.t. the atomic coordinates of the central atom, as this is required for an analytic computation of the atomic forces after the interpolation of an energy function or, equivalently, to consistently fit an FF from a force and energy database. Thirdly, a descriptor should be informative, e.g. should be able to discern different local atomic environments, while capturing the relevant physics of the system. For example, a smoothed function of the number of atoms present in a local atomic environment ρ_i is a descriptor that respects all invariance and differentiation properties, but on its own can not provide enough information to build an accurate model. Moreover, the descriptor should not only be informative but be so while capturing some of the essential features of the sought energy or force function (e.g. pairwise atomic interaction), as this makes the learning process faster and the resulting FF potentially more accurate. Finally, a descriptor should be computationally cheap w.r.t. the reference method used to build the database (e.g. DFT). As an extreme example, the *ab initio* force predicted on the central atom i in ρ_i meets all the above conditions, while being a useless descriptor because of its computational cost.

The above requirements allow for many degrees of freedom in the choice of the exact descriptor's form. For this reason, many local atomic environment descriptors have been designed and applied to build ML-FFs in the past years. Notable examples are atomic symmetry functions [21], spherical harmonics power spectrum [85, 86], Coloumb matrix [87], many-body tensor representation [88], moment tensors [89], crystal graphs [90], and fragment descriptors [91]. In some instances, such descriptors are optimised by a ML algorithm [30], or outsourced to online competitions [92]. In the next section, the explicit n-body descriptors and the novel "EAM"-like descriptors, used throughout the manuscript and in the "MFF" Python package, are presented [65, 84, 93, 66, 35]. Such descriptors respect all of the properties listed above and offer an intuitive and fast-to-compute way to represent local atomic environments.

2.2.2 Explicit n-body Descriptors

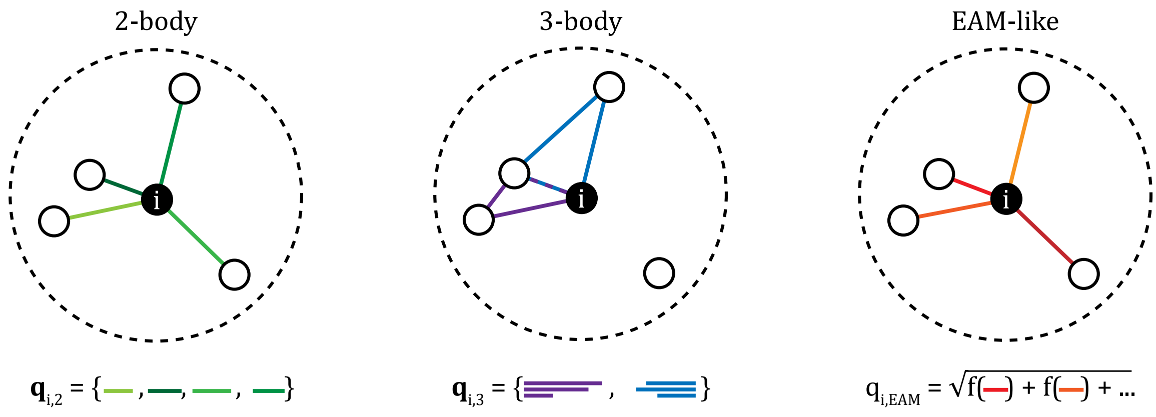


Figure 2.3: Visual representation of the 2-body (left), 3-body (center) and EAM-like descriptors (right) for an atom i. Transcription of the visual representation is also given.

In P-FFs, forces and energies are often computed as parametric closed-form functions of descriptors such as interatomic distances (e.g. Lennard-Jones potential [94]), angles (e.g. Tersoff [12] potential), or dihedral angles. These descriptors rely on the use of invariant degrees of freedom of groups of n atoms and are built so to respect the descriptor's requirements listed in Sec. 2.2.1. Since the versatility of an FF is directly influenced by the complexity of the employed descriptor, it is convenient to formally define a measure of such complexity.

We present a definition of a descriptor's order, first introduced as a definition of a kernel's order in Ref. [65]. The descriptor order identifies the number of simultaneously interacting particles that affect the descriptor's value. We formally define the order n for a descriptor $\mathbf{q}(\rho_i)$ of a local atomic environment ρ_i as the smallest integer n for which the following is true:

$$\frac{\partial^n \mathbf{q}_i}{\partial \mathbf{r}_1, \dots, \partial \mathbf{r}_n} = 0 \,\forall\, \mathbf{r}_1 \neq \cdots \neq \mathbf{r}_n \,\in\, \rho_i, \tag{2.9}$$

where $\mathbf{r}_1, \dots, \mathbf{r}_n$ are the positions of any choice of different atoms in the local atomic environment ρ_i. Following the definition above, it is easy to observe how descriptors based solely on pairwise distances are 2-body, descriptors which encode angular information are 3-body, descriptors presenting a dihedral term are 4-body, and so on. Finally, a descriptor $\mathbf{q}_i(\rho_i)$ whose value depends simultaneously on any number of atoms in ρ_i is called many-body.

We consider ML-FFs to be a natural extension of P-FFs, where the parametric equations governing the force and energy evaluations are replaced by nonparametric, data-driven functional forms yielded by the ML algorithm. For this reason, when designing the inputs to the ML algorithm, symmetry-invariant descriptors on the relevant degrees of freedom of groups of 2-, 3-, n-atoms, emerge as perfect candidates, given their similarity to classical P-FFs descriptors. With this notion in mind, we can construct n-body descriptors starting from the explicit degrees of freedom of n-plets of atoms. For a practical example, a 2-body descriptor can be built using the unordered set of distances of atoms j contained in the local atomic environment ρ_i:

$$\mathbf{q}_{i,2} = \{r_{ij}\}_{j\in\rho_i}. \tag{2.10}$$

It is then rather trivial to construct a 3-body descriptor $\mathbf{q}_{i,3}$, defined as the unordered set containing triplets of distances between atom i and other two atoms j and l which are each in the local atomic environment of the other two:

$$\mathbf{q}_{i,3} = \{(r_{ij}, r_{ik}, r_{jl})\}_{j,k\in\rho_i,i,l\in\rho_j,i,j\in\rho_l}. \tag{2.11}$$

The above descriptors, represented in Fig, 2.3, contain by construction the full 2- and 3-body information about the local atomic environment. They are computationally efficient, simple to interpret, and do not require any choice of parameters or truncation approximation. These advantages come at a cost. Indeed, the requirement that the above sets are unordered is strictly needed to preserve permutational invariance, and such a condition must be imposed also to the ML algorithm that uses these descriptors as inputs. This can be enforced rather straightforwardly when using a Gaussian process regression model (see Section 2.4). Explicit n-body features as the one provided above have been extensively used in this framework [95, 65, 96], also for building force fields for nanoclusters [84].

We note that, while 2- and 3-body descriptors emerge somehow trivially from the relevant degrees of freedom of pairs and triplets of atoms, this is not true for 4-body descriptors. The relative positions of four atoms cannot be uniquely described using a list of six interatomic distances [85]. Apart from this uniqueness problem, the dimensionality of an n-body descriptor grows exponentially with n, as shown in Tab. 2.1. It would, therefore, be convenient to restrict the descriptors used in ML-FFs to low orders of n to reduce the overall computational burden. Indeed, the two most used descriptors in literature, namely, atomic symmetry functions [21] and spherical harmonics power spectrum [23], are constructed using solely 2 and/or 3-body features. Low order descriptors capture well the ionic and covalent nature

n	Dimensionality of $\mathbf{q}_n$
1	$\mathscr{O}(1)$
2	$\mathscr{O}(M)$
3	$\mathscr{O}(M^2)$
4	$\mathscr{O}(M^3)$
$\vdots$	$\vdots$
n	$\mathscr{O}(M^{n-1})$

Table 2.1: The dimensionality of an n-body descriptor $\mathbf{q}_n$ based on the invariant degrees of freedom of n-plets of atoms, for a local atomic environment containing M atoms.

of chemical bonds and are therefore a rather natural choice for most systems where many-body interactions do not contribute significantly to the energy of the system.

It is important to note that the absence of any angular information in a 2-body descriptor always prevents a full characterization of higher-order interactions. This is not true for 3-body descriptors, and a nonlinear function of a 3-body descriptor can in principle capture also higher-order interactions [21, 23]. Nonetheless, it could be desirable to use a ML approach that preserves the order n of the descriptor, yielding an ML-FF that is also n-body. This is, for example, true for mapped force fields (M-FFs) [84], where a tabulation process can be used to transform ML-FFs into classical nonparametric FFs, to obtain a very large computational speed-up without accuracy loss, as presented in Sec. 3.1. If such an approach is indeed sought, higher-order interactions must be incorporated using a many-body descriptor that is low-dimensional (or scalar).

2.2.3 EAM-like Descriptors

For some systems, FFs based on 2+3-body descriptors might not be accurate enough w.r.t. the *ab initio* reference data, as displayed, for example, in Section 5.2. In such systems, a many-body descriptor can help reduce the error incurred by the FFs on the *ab initio* forces and energies; this approach is not uncommon for ML-FFs [23, 95, 80, 97]. Unfortunately, as discussed later in Sec. 3.1, most many-body FFs cannot be transformed into nonparametric tabulated FFs because of the high dimensionality of their input space. This problem can be circumvented by using a force field where the energy is a function of a low-dimensional input, which is still able to capture many-body atomic interactions. A prime example of such a descriptor can be found in the Embedded Atom Method (EAM) FFs, where the model's many-body energy $\hat{\varepsilon}_{MB}(\mathbf{q}_i)$ is modelled by:

$$\hat{\varepsilon}_{MB} = \phi(q_{MB,i}), \tag{2.12}$$

with:

$$q_{MB,i} = \sum_{j \in \rho_i} g(r_{ij}), \tag{2.13}$$

where g and ϕ are scalar functions, and q_{MB} is a many-body scalar descriptor [98]. The function ϕ can then be inferred by an ML algorithm, given an appropriate descriptor choice. The mappable many-body descriptor used in this thesis, and in the "MFF" package, q_{EAM} is inspired by the local energy equation found in second-moment approximation to the tight binding [98], and reads:

$$q_{\mathrm{EAM},i} = -\sqrt{\sum_{j \in \rho_i} e^{-2\left(r_{ij}/r_0 - 1\right)}}, \tag{2.14}$$

where r_0 is a distance hyperparameter of the descriptor. Such descriptor (Eq. 2.14) is many-body by the definition given in Eq. 2.9, and possesses all of the key properties listed in Sec. 2.2.1. Since the descriptor is a scalar, an appropriate ML method that uses it as an input can be easily mapped into a tabulated FF. This will be discussed in Sec. 3.1.2.

2.3 Linear Regression and Artificial Neural Networks

Linear regression and artificial neural networks are, together with Gaussian process regression (or, similarly, kernel ridge regression), the most widely used ML frameworks to construct ML-FFs. While the work contained in this thesis deals only with Gaussian process regression methods, here we present the state-of-the-art of the other two algorithms for ML-FF generation so that the reasoning behind the use of

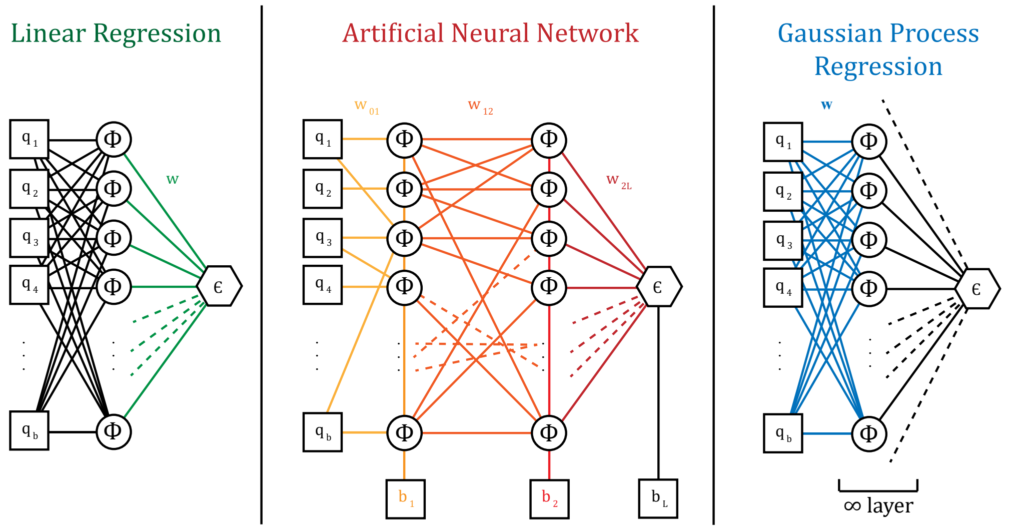


Figure 2.4: Schematic representation of linear regression, 2-layer artificial neural network, and Gaussian process regression. The figure is structured so to highlight differences and similarities across the three algorithms. The symbols in the figure mirror the ones used in the main text. The equivalency between a Gaussian process and a single-layer neural network with infinite nodes was proven in [81].

GPR in this work can be introduced more clearly. Fig. 2.4 shows a schematic comparison of the structure of LR, ANN and GPR algorithms for supervised learning.

2.3.1 Linear Regression

The local energy function ε_i can be expressed as a linear combination of functions of the local atomic environment descriptor as:

$$\hat{\varepsilon}(\mathbf{q}_i) = \mathbf{W}^T \phi(\mathbf{q}_i), \tag{2.15}$$

where $\mathbf{W}$ indicates the weights, $\hat{\varepsilon}(\mathbf{q}_i)$ is the modelled energy function and ϕ is a multivariate function. The choice of the exact form of ϕ is on the user, and its expression directly influences the accuracy of the regression algorithm. The weights are usually optimised so that the squared error loss L_2 on the total energy E_d is minimised:

$$L_2 = \sum_{d=1}^{D_E} ||\hat{E}_d - E_d||^2. \tag{2.16}$$

The weights that minimise L_2 can be easily found analytically via ordinary

least squares. In the linear regression case, the solution to the learning problem is therefore fast to compute, and the predictions the model yields are computationally cheap [22]. Despite the simplicity of the learning model, a variation of this method proved to yield interesting results, for example in aiding the prediction of adsorption energies of NO on RhAu nanoparticles [99, 100].
Nonetheless, in most applications, linear regression models lack the complexity needed to construct an accurate force field. The arbitrary choice of the function ϕ indeed restricts the representative power of FFs generated using linear regression, much like the choice of a parametric functional form in P-FFs constricts their flexibility.

2.3.2 Artificial Neural Networks

ANNs are a framework for supervised learning; they are composed of nodes organised in layers that connect the input $\mathbf{q}_i$ to an output $\varepsilon(\mathbf{q}_i)$. In ANNs, connections have weights $\mathbf{W}$ which multiply the connection's input value, and biases $\mathbf{B}$ which are instead added to a node's value. These weights and biases are optimised simultaneously during training and are usually initialized at random values. For a two-hidden-layer ANN, the equation for the prediction of the local energy $\hat{\varepsilon}(\mathbf{q}_n)$ reads:

$$\hat{\varepsilon}(\mathbf{q}_n) = \phi \left\{ b_3 + \sum_{j_2} w_{23}^{j_2} \phi \left[b_2^{j_2} + \sum_{j_1} w_{12}^{j_1 j_2} \phi \left(b_1^{j_1} + \sum_i w_{01}^{i j_1} (\mathbf{q}_n)_i \right) \right] \right\}, \tag{2.17}$$

where ϕ is a scalar activation function (tangent, hyperbolic sigmoid, etc.), the j_l indices run over the nodes of layer l (here $l = 1, 2$), $w_{l,l'}$ are the weights connecting layer l nodes to layer l' nodes, and $\mathbf{b}_l$ are the biases added to nodes of layer l. The structure of a 2-layers ANN can perhaps be more easily understood by looking at Fig. 2.4; it is then straightforward to imagine how Eq. 2.17 extends to networks with more than two hidden layers.

Much like in linear regression, the training of ANNs consists in the search of weights $\mathbf{W}$ and biases $\mathbf{B}$ which minimise a loss function L on a training set containing N_{tr} points. The squared error loss (Eq. 2.16) is usually chosen for regression problems such as energy and/or force fitting, but other loss functions can also be used (e.g. absolute error loss). Given how the number of parameters $\{\mathbf{W}, \mathbf{B}\}$ can range from thousands to millions, the optimization task is not trivial and the loss function is highly non-convex. For this reason, deterministic gradient descent on the loss landscape induced by the loss function computed on the full training set would be sub-optimal, as the optimization would stop at the first local minima of the loss function. Therefore, batch training is typically used to introduce stochasticity in the optimization process: a subset of training points is chosen and many gradient descent steps are taken; this process is then iterated. The training process is then stopped once the error on the validation set starts increasing, indicating that the ANN is starting to over-fit on the training data.

ANNs are universal approximators as, in the limit of an infinite number of nodes, they can approximate any continuous function to arbitrary precision [101, 102]. This is one of the main appeals to the method since, given enough time, computational power, and training data, one could construct a very large ANN and in principle approximate any function, no matter how complex. ANNs have been used to develop FFs for many systems in condensed matter physics, such as bulk metals and semiconductors [103, 104, 105], and metallic nanoparticles [106, 107, 108, 109, 106, 110, 111, 112, 113, 114, 115, 105]. The price to pay for the flexibility and expressive power of ANNs is that they are very data-hungry, and typically require orders of magnitude more training points than linear regression or GPR methods to reach the desired accuracy [116]. In the field of ML-FFs, the need for large training sets can impose a severe obstacle to the development of ANN-FFs, as only widely-studied systems usually possess databases large enough to train an ANN-FF. Moreover, given ANNs' complex structure, formed by a large number of layers connected via non-linear activation functions, ANN-FFs are often difficult to interpret in a human sense.

2.4 Gaussian Process Regression

We present the ML algorithm used to infer ML-FFs throughout this work and in the "MFF" Python package: Gaussian process regression. GPR algorithms present several advantages over linear regression and ANNs when it comes to the generation of ML-FFs. Namely, they require orders of magnitude less data than ANN to converge to a certain accuracy, which is beneficial when working on systems where a large database of consistent *ab initio* calculations is not already present and has, therefore, to be generated. Moreover, GPR being a fully Bayesian approach means that it is possible to automatically assign an uncertainty to the predictions yielded by GP-FFs. This feature allows for on-the-fly training of the ML-FF, which is trained on new *ab initio* whenever the system is deemed to be inaccurate for the task at hand [96]. Finally, GP-FFs are straightforward to interpret, i.e. the underlying properties of the learnt FFs, e.g. the order of atomic interaction they can describe, are directly dictated by the chosen form of the kernel function. They have been employed for the creation of ML-FFs for both bulk systems [95, 117, 97] and nanoparticles [84, 118].

The main drawbacks of GPR are two: the care needed in the choice of the kernel function, as this directly enforces the properties of the learnt FF, and, most importantly, the computational cost of predictions, which scales linearly with the number of training points and is generally higher than the one of comparable ANNs [116]. In the following sections, we present the formalism behind GPR for ML-FFs, and analyse thoroughly the generation of kernel functions that respect the physical invariances associated to force fields, and that allow for a "mapping" of the GP-FFs. This "mapping" method, discussed in detail in Sec. 3.1, allows circumventing the poor computational scaling of GP-FFs in prediction tasks, therefore removing the biggest drawback associated with the use of GP-FFs.

2.4.1 Formalism

A GP regression is a Bayesian framework that learns from a database $\mathscr{D}$ of input-output pairs how to predict outputs given new inputs [81]. We assume that a database of local atomic environment descriptor - local energy pairs exists:

$\mathscr{D}_\varepsilon = \{\mathbf{q}_i, \varepsilon_i^r\}$. Although the atomic energies are eventually not available from, for example, *ab initio* calculations, it is more convenient to introduce the formalism behind the GP regression using a scalar quantity, such as the local atomic energy and then to extend it to local atomic forces and total energies, as explained in Sec. 2.4.2.

We assume that the reference local energies ε^r in the training database can be written as [81, 23, 93]:

$$\varepsilon_i^r = \varepsilon(\mathbf{q}_i) + \xi_i, \tag{2.18}$$

where the noises ξ_i are independent zero-mean Gaussian random variables with standard deviation λ:

$$\xi_i \sim \mathcal{N}(0, \lambda^2). \tag{2.19}$$

This noise in the data can be thought of as a combination of the uncertainty associated with the method used to generate the data, and of the uncertainty coming from approximations and assumptions made when predicting the local energy. For example, an important source of uncertainty is the *locality error* resulting from the assumption of a finite cutoff radius, outside of which we consider atoms not to interact. Another source of uncertainty could be given by the choice of a non-fully-descriptive local atomic environment descriptor $\mathbf{q}_i$ (e.g. 2-body descriptor).

The main advantage of GP regression over linear regression or parametric approaches is that $\varepsilon(\mathbf{q})$ is not constrained to a given parametric functional form. It is instead assumed that the GP predictions are distributed as a Gaussian stochastic process, usually with zero mean:

$$\varepsilon(\mathbf{q}_i) \sim \mathtt{GP}(0, k(\mathbf{q}_i, \mathbf{q}_i)), \tag{2.20}$$

where k is the *kernel* or *covariance* function of the GP. The notation of Eq. 2.20 indicates that for any finite set of inputs $\{\mathbf{q}_1, \dots, \mathbf{q}_{D_\varepsilon}\}$, the corresponding set of outputs $\boldsymbol{\varepsilon} = (\varepsilon(\mathbf{q}_1), \dots, \varepsilon(\mathbf{q}_{D_\varepsilon}))^T$ is distributed according to a multivariate Gaussian distribution. The covariance matrix of such distribution is constructed using the kernel function:

$$p(\varepsilon) = \mathcal{N}(0, \mathbb{K}), \tag{2.21}$$

where the elements of the Gram matrix $\mathbb{K}$ are calculated as:

$$[\mathbb{K}]_{ij} = k(\mathbf{q}_i, \mathbf{q}_j). \tag{2.22}$$

Since both ξ and $\varepsilon(\mathbf{q})$ follow a Gaussian distribution, and since the sum of two Gaussian variables is also a Gaussian variable, we can express the distribution of the reference energies ε^r of Eq. (2.18) as another Gaussian distribution where the mean and the covariance matrix are the sums of the original two:

$$p(\varepsilon^r) = \mathcal{N}(0, \mathbb{C}), \tag{2.23}$$

with:

$$\mathbb{C} = \mathbb{K} + \mathbb{I}\lambda^2, \tag{2.24}$$

where $\mathbb{I}$ is the identity matrix.

We can now write the probability distribution for the local energy ε_i associated to an unseen input configuration $\mathbf{q}_i$, given the knowledge of a training dataset $\mathscr{D}_\varepsilon = \{\mathbf{q}_d, \varepsilon_d^r\}$ [81, 119]:

$$p(\varepsilon_i \mid \mathbf{q}_i, \mathscr{D}_\varepsilon) = \mathcal{N}(\bar{\varepsilon}(\mathbf{q}_i), \mathrm{VAR}(\mathbf{q}_i)), \tag{2.25}$$

where:

$$\bar{\varepsilon}(\mathbf{q}_i) = \mathbf{k}^T \mathbb{C}^{-1} \varepsilon^r, \tag{2.26}$$

and:

$$\mathrm{VAR}(\mathbf{q}_i) = k(\mathbf{q}_i, \mathbf{q}_i) + \lambda^2 - \mathbf{k}^T \mathbb{C}^{-1} \mathbf{k}. \tag{2.27}$$

In Eqs. (2.26), (2.27) we define the vector of kernel function evaluations $\mathbf{k} = (k(\mathbf{q}_i, \mathbf{q}_1), \ldots, k(\mathbf{q}_i, \mathbf{q}_{D_\varepsilon}))^T$. The variance of ε_i furthermore provides us with an estimate of the uncertainty associated with the prediction. This uncertainty is typically expressed as the standard deviation $\sqrt{\mathrm{VAR}}(\mathbf{q}_i)$ found in Eq. (2.27). The mean $\bar{\varepsilon}(\mathbf{q})$ of the predictive distribution is now the best guess for the prediction of the local energy function, under the assumption of a squared loss function on the local energy prediction [81, 23, 93]. This mean function is therefore used to predict output values given inputs; we can then rewrite it as an explicit summation over the training dataset of the kernel function:

$$\bar{\varepsilon}(\mathbf{q}_i) = \sum_{d=1}^{D_\varepsilon} k(\mathbf{q}_i, \mathbf{q}_d)\alpha_d, \tag{2.28}$$

where the coefficients α_d are obtained as $\alpha_d = (\mathbb{C}^{-1}\varepsilon^r)_d$. The kernel function k can be thought of as a "similarity" function between pairs of inputs and in this sense Eq. (2.28) helps to visualise the GPR prediction as a "weighted average" of the similarity between a target input descriptor $\mathbf{q}_i$ and each other input descriptor in the training set.

2.4.2 Learning from Energy and Forces

Training a GP-FF usually requires a database of total energies and/or a database of local forces, as these are the quantities usually available in the quantum reference data (coming e.g. from AIMD simulations). In order to build kernel functions that respect the energy conservation principle (Eq. (2.3)), and the local energy approximation (Eq. (2.2)),we start from a local energy-local energy kernel function k and derive [86]:

$$\begin{aligned}
k^{\varepsilon E}(\mathbf{q}_i,\mathbf{Q}_b) = &\quad \textstyle\sum_{j\in b} k(\mathbf{q}_i,\mathbf{q}_j), \\
k^{EE}(\mathbf{Q}_a,\mathbf{Q}_b) = &\quad \textstyle\sum_{i\in a}\sum_{j\in b} k(\mathbf{q}_i,\mathbf{q}_j), \\
\mathbf{k}^{\varepsilon f}(\mathbf{q}_i,\mathbf{q}_j) = &\quad -\frac{\partial k(\mathbf{q}_i,\mathbf{q}_j)}{\partial \mathbf{r}_j}, \\
\mathbf{k}^{fE}(\mathbf{q}_i,\mathbf{Q}_b) = &\quad -\textstyle\sum_{j\in b}\frac{\partial k(\mathbf{q}_i,\mathbf{q}_j)}{\partial \mathbf{r}_i}, \\
\mathbf{K}^{ff}(\mathbf{q}_i,\mathbf{q}_j) = &\quad \frac{\partial^2 k(\mathbf{q}_i,\mathbf{q}_j)}{\partial \mathbf{r}_i \partial \mathbf{r}_j},
\end{aligned} \tag{2.29}$$

where $k^{\varepsilon E}$ is the local energy-total energy kernel function, k^{EE} the total energy-total energy kernel function, $k^{\varepsilon f}$ the local energy-force kernel function, $\mathbf{k}^{fE}$ the force-total energy kernel function, and $\mathbf{K}^{ff}$ the force-force kernel function. This array of kernel functions is required when training on total energy and force data, and when predicting forces and/or energies. The equations for the energy and force prediction in fact read, respectively:

$$\hat{\varepsilon}_i(\mathbf{q}_i) = \sum_{d_E=1}^{D_E} k^{\varepsilon E}(\mathbf{q}_i,\mathbf{Q}_{d_E})\alpha^E_{d_E} + \sum_{d_f=1}^{D_f} \mathbf{k}^{\varepsilon f}(\mathbf{q}_i,\mathbf{q}_{d_f})^T \alpha^f_{d_f}. \tag{2.30}$$

and:

$$\hat{\mathbf{f}}_i(\mathbf{q}_i) = \sum_{d_E=1}^{D_E} \mathbf{k}^{fE}(\mathbf{q}_i,\mathbf{Q}_{d_E})\alpha^E_{d_E} + \sum_{d_f=1}^{D_f} \mathbf{K}^{ff}(\mathbf{q}_i,\mathbf{q}_{d_f})\alpha^f_{d_f}. \tag{2.31}$$

In Eqs. (2.30), (2.31), $\alpha^E_{d_E}$ is the total energy weight associated with the data point d_E, and $\alpha^f_{d_f}$ is the 3-D vector of force weights associated with the data point d_f. The total energy α^E and force weights α^f are obtained during training as:

$$\alpha = [\alpha^E, \alpha^f] = (\mathbb{K} + \lambda \mathbb{I})^{-1} \cdot [\mathbf{E},\mathbf{F}], \tag{2.32}$$

where $[\bullet,\bullet]$ indicates the column stacking of vectors, $\mathbb{I}$ is the identity matrix, and λ the hyperparameter associated to noise in the training data, as seen in Eq. (2.19). The Gram matrix $\mathbb{K}$ is a block matrix computed during training, with the following structure:

$$\mathbb{K} = \left[\begin{array}{cc} \mathbb{K}^{EE} & \mathbb{K}^{Ef} \\ \mathbb{K}^{Ef^T} & \mathbb{K}^{ff} \end{array} \right], \tag{2.33}$$

and where each block is a matrix containing kernel functions evaluated between each pair of inputs in each dataset. More specifically:

$$\begin{aligned} \mathbb{K}^{EE}_{ij} &= k^{EE}(\mathbf{Q}_i, \mathbf{Q}_j) \quad \text{with } i, j \in \mathscr{D}_e, \\ \mathbb{K}^{Ef}_{il} &= \mathbf{k}^{Ef}(\mathbf{Q}_i, \mathbf{q}_l) \quad \text{with } i \in \mathscr{D}_e, \ l \in \mathscr{D}_f, \\ \mathbb{K}^{ff}_{lm} &= \mathbf{K}^{ff}(\mathbf{q}_l, \mathbf{q}_m) \quad \text{with } l, m \in \mathscr{D}_f. \end{aligned} \tag{2.34}$$

Now that the interactions between force, local energy, and total energy data is clarified, we focus on the construction of the local energy-local energy kernel function $k^{\varepsilon,\varepsilon}$, from which all other kernel functions can be derived according to Eq. (2.29).

2.4.3 *n*-body Kernels

Starting from the n-body descriptors defined in Sec. 2.2.2, we present a way to systematically build n-body kernels that respect all the invariance properties of Sec. 2.2.1, and that posses the same interaction order of the descriptor they take as input [65]. As already mentioned, n-body descriptors such as the ones defined in Eqs. (2.10), (2.11) are invariant w.r.t. translation, rotation and permutation of atoms. We build invariant n-body kernels, k_n, by explicitly summing a non-symmetric kernel function $\tilde{k}_n$ over the components of input n-body feature vectors $\mathbf{q}_n$, i.e. on the symmetry-invariant degrees of freedom of n-plets of atoms:

$$k_n(\mathbf{q}_{i,n}, \mathbf{q}_{j,n}) = \sum_{\mathbf{c}_n \in \mathbf{q}_{i,n}} \sum_{\mathbf{c}'_n \in \mathbf{q}_{j,n}} \tilde{k}_n(\mathbf{c}_n, \mathbf{c}'_n), \tag{2.35}$$

where $\mathbf{c}_n$ and $\mathbf{c}'_n$ encode the information relative to each n-plet of atoms in ρ_i and ρ_j, respectively. The non-symmetric n-body kernel $\tilde{k}_n$ can be any differentiable kernel. We opt for the radial basis function kernel because of its simplicity, smoothness, and computational efficiency:

$$\tilde{k}_n(\mathbf{c}_n, \mathbf{c}'_n) = \exp\left(-\frac{||\mathbf{c}_n - \mathbf{c}'_n||^2}{2\sigma^2}\right), \tag{2.36}$$

where σ is the kernel's lengthscale hyperparameter, which has the same units of measure as the descriptor $\mathbf{c}_n$, and controls the spatial variability of the learnt GP-FF. Applying the procedure described above to the invariant descriptors of Eqs. (2.10), (2.11), the 2-body local energy-local energy symmetric kernel reads:

$$k_2(\mathbf{q}_{i,2}, \mathbf{q}_{j,2}) = \sum_{l \in \rho_i} \sum_{m \in \rho_j} \exp\left(-\frac{||r_{il} - r_{jm}||^2}{2\sigma^2}\right), \tag{2.37}$$

where r_{il} indicates the distance between atoms i and l. Similarly, the 3-body local energy-local energy symmetric kernel is:

$$k_3(\mathbf{q}_{i,3}, \mathbf{q}_{j,3}) = \sum_{\mathbf{c}_3 \in \mathbf{q}_{i,3}} \sum_{\mathbf{c}'_3 \in \mathbf{q}_{j,3}} \sum_{\mathbf{P} \in \mathscr{P}_3} \exp\left(-\frac{||\mathbf{c}_3 - \mathbf{P}\mathbf{c}'_3||^2}{2\sigma^2}\right), \tag{2.38}$$

where $\mathbf{q}_{i,3}$ and $\mathbf{q}_{j,3}$ are defined as per in Eq. (2.11), and $\mathbf{P}$ is an element of the set of cyclical permutations of 3 objects $\mathscr{P}_3$.

Kernels such as the one in Eq. (2.35) are computationally demanding for $n > 3$: the computational cost of evaluating an n-body kernel is $\mathscr{O}(M^{2(n-1)})$, where M is the average number of atoms in the local atomic environment. This scaling can be found by observing that, as displayed in Tab 2.1, the dimensionality of $\mathbf{q}_n$ grows as M^{n-1}, and that to compute a kernel function we must sum over the dimensions of the descriptors of both inputs.

To obtain high order kernels which have better computational scaling than the ones presented above, it is possible to augment the order of interaction of an already symmetric kernel from n to $n' = (n-1)\zeta + 1$ by raising it to a power ζ [23, 85, 65]:

$$k^{\#}_{n'}(\rho_i, \rho_j) = \left(k_n(\rho_i, \rho_j)\right)^{\zeta}. \tag{2.39}$$

This operation does not increase the computational cost of evaluating the kernel, but the *uniqueness* of the underlying descriptor is lost; this is represented by the $\not{u}$ superscript in Eq. (2.39). An n-body descriptor $\mathbf{q}_n$ possesses the *uniqueness* property if any n-plet of atoms is mapped uniquely to a descriptor's value. This can easily be understood through an example, illustrated in Fig. 2.5, where in the first case the triplet of atoms is mapped onto a unique descriptor containing three interatomic distances (r_{il}, r_{lm}, r_{im}), while in the second case the non-unique descriptor contains only two distances (r_{il}, r_{im}). Both descriptors in Fig. 2.5 are 3-body as

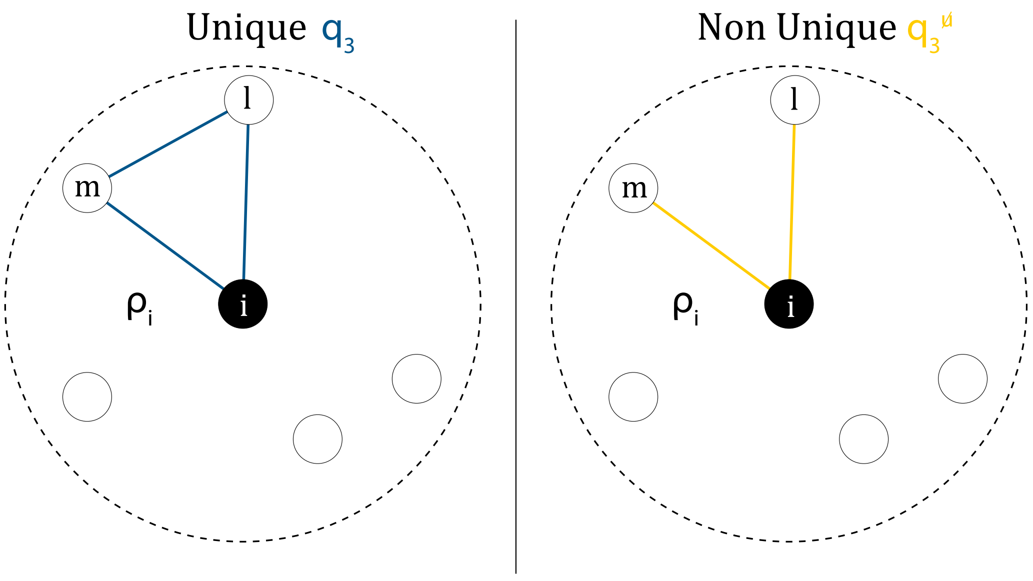


Figure 2.5: Schematic representation of a 3-body unique descriptor $\mathbf{q}_3$ (left, blue) and a 3-body non-unique descriptor $\mathbf{q}_3^{\not{u}}$ (right, yellow) for a local atomic environment ρ_i.

per Eq. (2.9), but the non-unique 3-body descriptor is *not* able to resolve angular information, as pairs of atoms l, m with distances (r_{il}, r_{im}) from the central atom i forming different angles $\angle lim$ would be encoded onto the same descriptor (r_{il}, r_{im}).

A non-unique many-body kernel can be obtained by exponentiating a symmetric n-body kernel:

$$k_{MB}(\rho_i, \rho_j) = \exp\left(\frac{k_n(\rho_i, \rho_j)}{\gamma^2}\right), \tag{2.40}$$

where γ is a hyperparameter that governs the relative importance of low-body and high-body interactions. This kernel is impossible to map onto a tabulated FF because it is equivalent to a sum of n body kernels with $n \to \infty$; this can be easily checked by looking at the Taylor series' expansion of Eq. (2.40) [65].

To enforce energy conservation whenever atoms leave or enter a local atomic environment during an MD simulation, it is necessary to add a smooth cutoff function to the local energy-local energy kernel functions. This additional term to the local energy-local energy kernel function propagates to all mixed force/energy kernels as per Eq. (2.29). The choice of a smooth function of the interatomic distance which has value and first derivative 0 at $r_{ij} = r_C$ leaves a lot of freedom w.r.t. its specific expression. In this work, and in the MFF package presented in the next chapter, we adopted the following, widely employed, cutoff function on interatomic distances:

$$f_C(r_{ij}) = \frac{1}{2}\left(1 + \cos\left(\pi \frac{r_{ij}}{r_C}\right)\right). \tag{2.41}$$

When the cutoff function of Eq. 2.41 is added to the 2-body local energy-local energy kernel function, the resulting kernel reads:

$$k_{2,C}(\mathbf{q}_{i,2}, \mathbf{q}_{j,2}) = \sum_{k \in \rho_i} \sum_{l \in \rho_j} \exp\left(-\frac{||r_{ik} - r_{jl}||^2}{2\sigma^2}\right) \cdot f_C(r_{ik}) \cdot f_C(r_{jl}). \tag{2.42}$$

For notational simplicity, the cutoff function is excluded from all equations in this manuscript.

2.4.4 EAM-like Kernels

To construct a GP-FF which is many-body and can be tabulated into a mapped force field, it is necessary to use a many-body descriptor that is low-dimensional. Moreover, the resulting local energy-local energy kernel must be differentiable w.r.t. the position of the central atoms in the local atomic environments, so that derivative kernels can be obtained as per Eq. (2.29).

We here introduce a way to construct such a kernel by employing a radial basis function kernel on the EAM-like descriptor introduced in Eq. (2.14), to obtain:

$$k_{\text{EAM}}(q_{\text{EAM},i}, q_{\text{EAM},i}) = \exp\left(-\frac{||q_{\text{EAM},i} - q_{\text{EAM},i}||^2}{2\sigma^2}\right). \tag{2.43}$$

This kernel is many-body, like the descriptor it originates from, and is non-unique: its value does not depend on all the degrees of freedom of the atoms in the local atomic environment, as the EAM-like descriptor does not resolve angular information. Nonetheless, the addition of the EAM-kernel to GP-FFs employing 2- and 3-body kernels can reduce the error incurred w.r.t. reference data; this is showed e.g. in Sec. 5.2. Moreover, the computational cost of evaluating an EAM kernel is, despite it being a many-body kernel, lower than any of the n-body kernels presented so far. In fact, given that the EAM descriptor is scalar, the calculation of the kernel function is $\mathscr{O}(1)$ and, since the cost of computing each descriptor is $\mathscr{O}(M)$, the total computational complexity of a kernel evaluation results to be $\mathscr{O}(M)$. We note that the procedure here introduced to generate many-body kernels that are mappable can be extended to create an array of diverse many-body, low-dimensional (e.g. scalar) kernels.

2.4.5 Kernels for Multiple Elements

The kernels presented up to now assume that all the atoms in the system are of the same chemical species. However, this formalism can be extended to the case of multi-chemical species, i.e. binary and ternary alloys, via the addition of an atomic-number discerning term to previously presented kernels. The n-body descriptors introduced in Sec. 2.2.2 can, therefore, be expanded to account for the chemical species of atoms; specifically, we can write the element-discerning 2-body descriptor as:

$$\mathbf{q}^m_{i,2} = \left[\mathbf{q}_{i,2}, \{(z_i, z_j)\}_{j\in\rho_i}\right], \tag{2.44}$$

where z_i is the atomic number of atom i. Similarly, the element-discerning 3-body descriptor reads:

$$\mathbf{q}^m_{i,3} = \left[\mathbf{q}_{i,3}, \{(z_i, z_j, z_l)\}_{j,l\in\rho_i,i,l\in\rho_j,i,j\in\rho_l}\right]. \tag{2.45}$$

It is then trivial to write the element-discerning 2-body kernel as:

$$k_2(\rho_i,\rho_j) = \sum_{l\in\rho_i}\sum_{m\in\rho_j} \exp\left(-\frac{||r_{il}-r_{jm}||^2}{2\sigma^2}\right)\left(\delta(z_i,z_j)\delta(z_l,z_m)+\delta(z_i,z_m)\delta(z_l,z_j)\right), \tag{2.46}$$

where $\delta(z_l,z_m)$ indicates the Kronecker delta function. The element-discerning 3-body kernel can be derived in a similar fashion, its explicit equation is not shown here because of the cumbersome notation required.

Chapter 3

MFF: a Python Package

In this chapter, the MFF Python package, developed by the author in collaboration with Dr. Aldo Glielmo and Dr. Ádám Fekete, is introduced. The MFF python package offers a tool for researchers to automatically build *ad hoc* nonparametric classical force fields - named here mapped force fields (M-FFs) - and employs Gaussian process regression to train the FFs starting from a database of reference calculations. The package is publicly available at `https://github.com/kcl-tscm/mff` and contains documentation and tutorials [35]. A diagram representing the transformation of raw reference data into a classical nonparametric FF within the MFF package is depicted in Figure 3.1.

After the initial submission of this thesis manuscript, a collaboration started between the author, Aldo Glielmo, and the authors of the Fast Learning of Atomic Rare Events (FLARE) Python package [96], which resulted in a merging of the two packages. The majority of the functionalities included in MFF, most prominently GP-FF mapping and EAM-like kernels, are therefore available in the FLARE package, which can be found at `https://github.com/mir-group/flare`. While the algorithmic details may differ between the two codes, the underlying theory is the same, and is discussed in detail in this chapter.

In Sec. 3.1, we introduce the method that is the core of the MFF package and that allows transforming GP-FFs into nonparametric classical FFs: the “mapping”

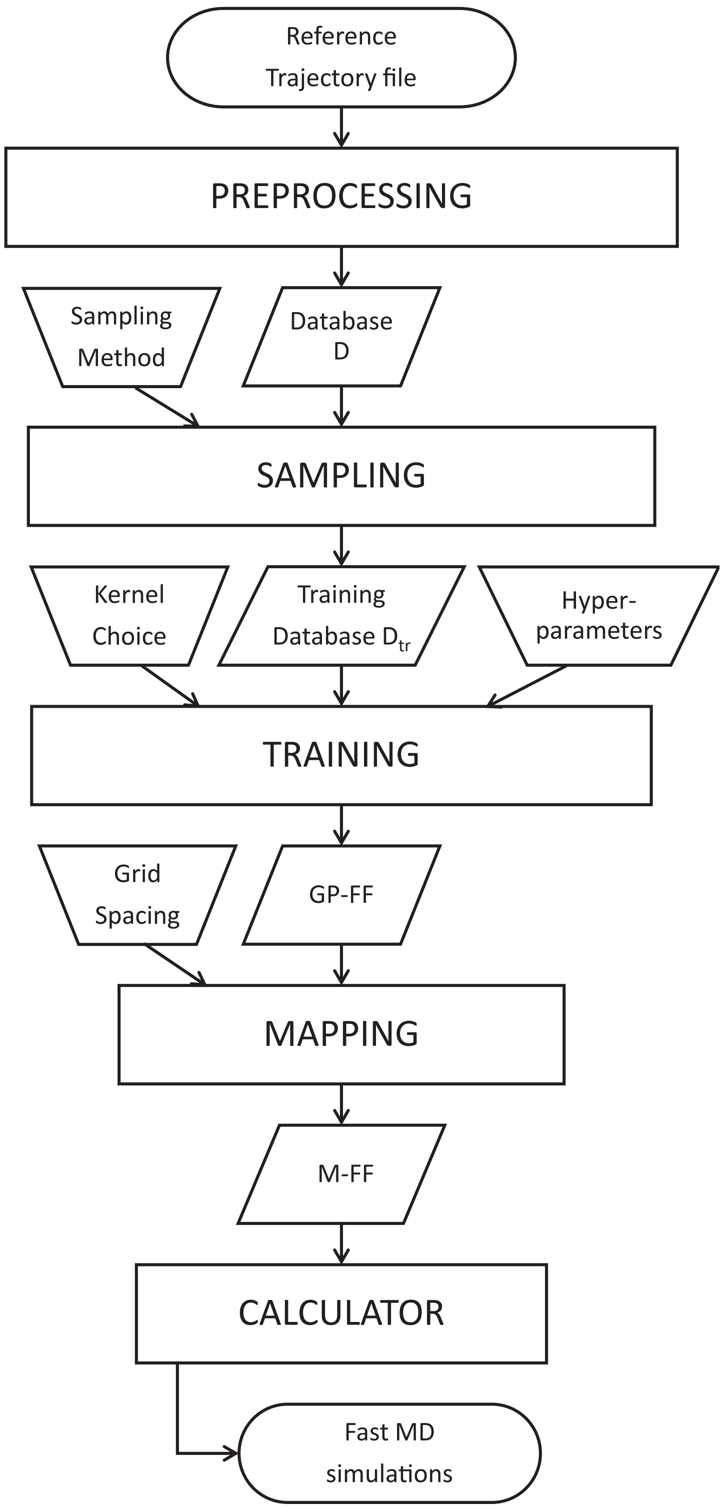


Figure 3.1: Flowchart illustrating the typical usage of the MFF package. User inputs or choices are displayed in trapezoidal blocks, method calls in rectangles, data in parallelepipeds and endpoints in rounded rectangles.

methodology. This rather general procedure is used to remove the computational burden of GP algorithms associated with having to compute the kernel function between a new input and the whole training set for every output prediction. Thanks to the speed-up offered by mapping the GP-FF, the resulting *n*-body M-FF is as fast as classical parametrized FFs of the same order *n*, while maintaining the accuracy w.r.t. reference data of the original GP-FF. In Sec. 3.2 we present the modules contained in MFF and explain the overall workflow for the production and usage of a mapped force field. Finally, in Sec. 3.3 we showcase the accuracy and computational speed reachable by FFs generated with the MFF package for a set of bulk materials and metallic nanoparticles w.r.t. reference *ab initio* calculations.

3.1 Mapping Gaussian Process Force Fields

3.1.1 Mapping *n*-body GP-FFs

The computational cost of evaluating a GP-FF scales linearly with the number of training points, as the kernel function must be evaluated between a target input $\mathbf{q}_i$ and every input in the training set $\mathbf{q}_d$ (see Eq. (2.28)). We observe that once a GP is trained, and the training set is not modified, then the underlying FF is also fixed. We, therefore, might expect that the computational cost of an *n*-body FF depends only on the complexity of the input descriptor $\mathbf{q}_n$ (i.e. on its order *n* for unique *n*-body kernels), and is not dependant on the amount of data used in training. To formally prove this concept, we look at the equation for the local energy prediction in the case of a unique 3-body force field, which we obtain by combining Eqs. (2.28) and (2.38):

$$\hat{\varepsilon}(\mathbf{q}_{i,3}) = \sum_{d=1}^{D_\varepsilon} \sum_{\mathbf{c}_3 \in \mathbf{q}_{i,3}} \sum_{\mathbf{c}'_3 \in \mathbf{q}_{d,3}} \sum_{\mathbf{P} \in \mathscr{P}_3} \exp\left(- \frac{||\mathbf{c}_3 - \mathbf{P}\mathbf{c}'_3||^2}{2\sigma^2} \right). \tag{3.1}$$

We can then rearrange the above sum and bring the summation over $\mathbf{c}_3 \in \mathbf{q}_{i,3}$ outside to obtain:

$$\hat{\varepsilon}(\mathbf{q}_{i,3}) = \sum_{\mathbf{c}_3 \in \mathbf{q}_{i,3}} h(\mathbf{c}_3), \tag{3.2}$$

where $h(\mathbf{c}_3)$ can be thought of as the energy associated with the triplet of atoms described by $\mathbf{c}_3$:

$$h(\mathbf{c}_3) = \sum_{d=1}^{D_\varepsilon} \sum_{\mathbf{c}_3' \in \mathbf{q}_{d,3}} \sum_{\mathbf{P} \in \mathscr{P}_3} \exp\left(-\frac{||\mathbf{c}_3 - \mathbf{P}\mathbf{c}_3'||^2}{2\sigma^2}\right). \tag{3.3}$$

Since the dimensionality of the input space for the triplet energy function $h(\mathbf{c}_3)$ is low (i.e. 3-dimensional, as a triplet of atoms has 3 degrees of freedom), the function can easily be computed for a 3-D grid of $\mathbf{c}_3$ values and then stored as a table. This procedure effectively transforms the 3-body GP-FF into a tabulated 3-body M-FF, where the local energy is computed as a sum over triplets of atoms of a 3-body energy term. This 3-body energy term can be evaluated for every triplet of atoms via tricubic spline interpolation on the 3D stored grid of values. Its derivative, the 3-body force contribution, can be computed as the derivative of such spline w.r.t. the atomic position of the atom the force is acting on, $\mathbf{r}_i$:

$$\begin{aligned}
\hat{\mathbf{f}}(\mathbf{q}_{i,3}) &= -\sum_{\mathbf{c}_3 \in \mathbf{q}_{i,3}} \frac{\partial h(\mathbf{c}_3)}{\partial \mathbf{r}_i} \\
&= -\sum_{j,k \in \rho_i} \frac{\partial h(r_{ij}, r_{ik}, r_{jk})}{\partial \mathbf{r}_i} \\
&= -\sum_{j,k \in \rho_i} \frac{\partial h(r_{ij}, r_{ik}, r_{jk})}{\partial r_{ij}} \frac{\partial r_{ij}}{\partial \mathbf{r}_i} + \frac{\partial h(r_{ij}, r_{ik}, r_{jk})}{\partial r_{ik}} \frac{\partial r_{ik}}{\partial \mathbf{r}_i}.
\end{aligned} \tag{3.4}$$

Fig. 3.2 shows the convergence of the M-FF force error for a 3-body kernel w.r.t. the GP forces as a function of the number of grid points used to tabulate the function $h(\mathbf{c}_3)$. From Fig. 3.2 we can see how, once an appropriate number of grid points is used, the error introduced by the mapping procedure ($\sim 10^{-2}$) is one order of magnitude smaller than the error typically incurred by GP-FFs w.r.t. reference force data ($\sim 10^{-1}$ eV/Å). The cost of computing $h(\mathbf{c}_3)$ as per Eq. (3.3) is $\mathscr{O}(DM^2)$, while evaluating a tabulated function is $\mathscr{O}(1)$; this speed-up reduces the cost of a 3-body FF local energy calculation from $\mathscr{O}(DM^4)$ for a GP-FF to $\mathscr{O}(M^2)$ for an M-FF. Since a local atomic environment contains typically between 10 and 40 atoms, and the number of training points is typically $10^2 - 10^3$, the computational speed-

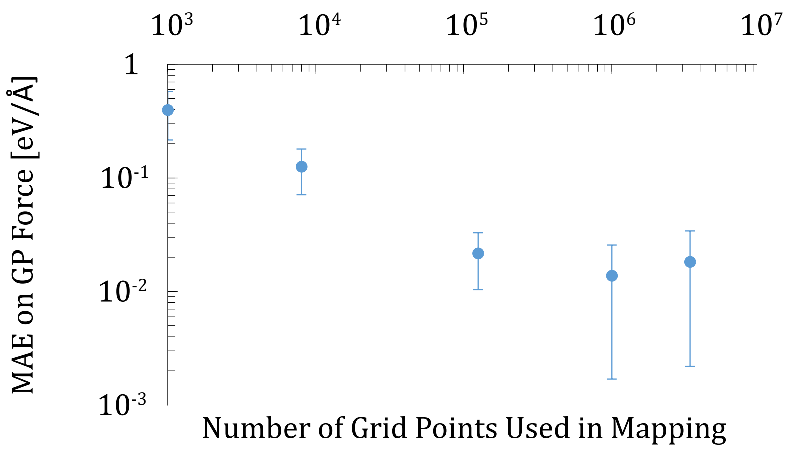


Figure 3.2: Log-log plot showing the error incurred by the 3-body M-FF force evaluation w.r.t. the GP-FF used to build such M-FF, as a function of the number of points used to build the tabulation grid, for a Ni_{13} nanoparticle. The standard deviation of the mean absolute error on force vector is displayed for each point.

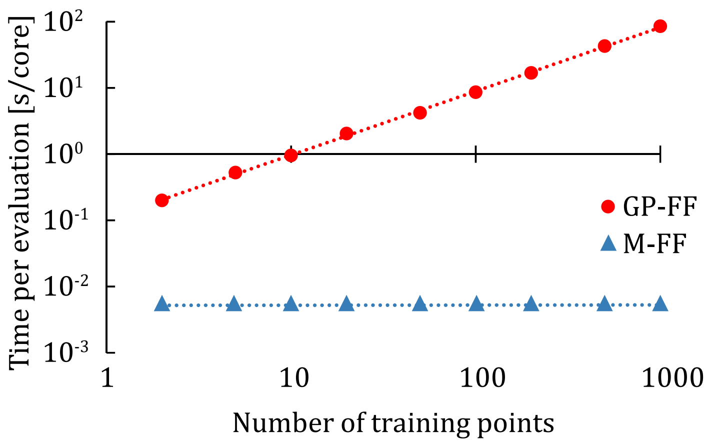


Figure 3.3: Plot showing the computational time needed for the force prediction on an atom in a Ni_{19} nanoparticle as a function of the number of training points for a 3-body GP-FF (red dots) and the M-FF built from the same 3-body GP-FF (blue dots). Original figure first published in [93].

up for mapping a 3-body GP-FF is of the order of $10^4 - 10^5$. This is represented in Fig. 3.3, where the computational cost of a 3-body GP-FF and relative M-FF are compared.

The mapping technique is very general, and it can be shown via the same procedure as the one displayed in Eqs. (3.1), (3.2), (3.3) that any *n*-body GP-FF (with *n* finite) can be transformed into a M-FF by explicitly tabulating the energy function associated to the n-plets of atoms. Mapping is available in the MFF package for GP-FFs of order $n = 2, 3$, as higher-order GP-FFs are computationally costly to tabulate. The dimensionality of the tabulation grid for n-body GP-FFs is equal to the number of invariant degrees of freedom of an n-plet of atoms: $3n-6$. For this reason, the grid becomes too large to properly sample whenever $n \geq 4$. GP-FFs of order above 4 are therefore not easily mappable, and unique many-body kernels are impossible to map, as their descriptor is infinite-dimensional, or very large-dimensional whenever a truncated expansion is used as a many-body descriptor, as is the case for the SOAP descriptor [23, 85].

3.1.2 Mapping EAM-like GP-FFs

While it is impossible to map unique many-body FFs, non-unique many-body GP-FFs can be transformed into M-FFs provided that the descriptor they employ is low-dimensional. This is indeed the case for the EAM-like descriptor introduced in Eq. (2.14), which is a scalar. An EAM-like GP-FF can, therefore, be mapped into an M-FF which can then be considered the nonparametric parallel to the parametric EAM potentials fitted using the force-matching technique [120]. The expression for the local energy of the mapped EAM-like FF is:

$$\hat{\varepsilon}(\mathbf{q}_{i,\mathrm{EAM}}) = h(\mathbf{q}_{i,\mathrm{EAM}}), \tag{3.5}$$

where h is the scalar mapped function, as the EAM-like descriptor q_{EAM} is a scalar representation of the whole local atomic environment ρ. The expression for the force reads instead:

$$\begin{aligned}\hat{\mathbf{f}}(\mathbf{q}_{i,\mathrm{EAM}}) &= -\frac{\partial h(\mathbf{q}_{i,\mathrm{EAM}})}{\partial \mathbf{r}_i} \\ &= -\frac{\partial h(\mathbf{q}_{i,\mathrm{EAM}})}{\partial \mathbf{q}_{i,\mathrm{EAM}}} \sum_{j \in \rho_i} \frac{e^{-2\left(r_{ij}/r_0 - 1\right)}}{r_0 \mathbf{q}_{i,\mathrm{EAM}}} \frac{\partial r_{ij}}{\partial \mathbf{r}_i}.\end{aligned} \tag{3.6}$$

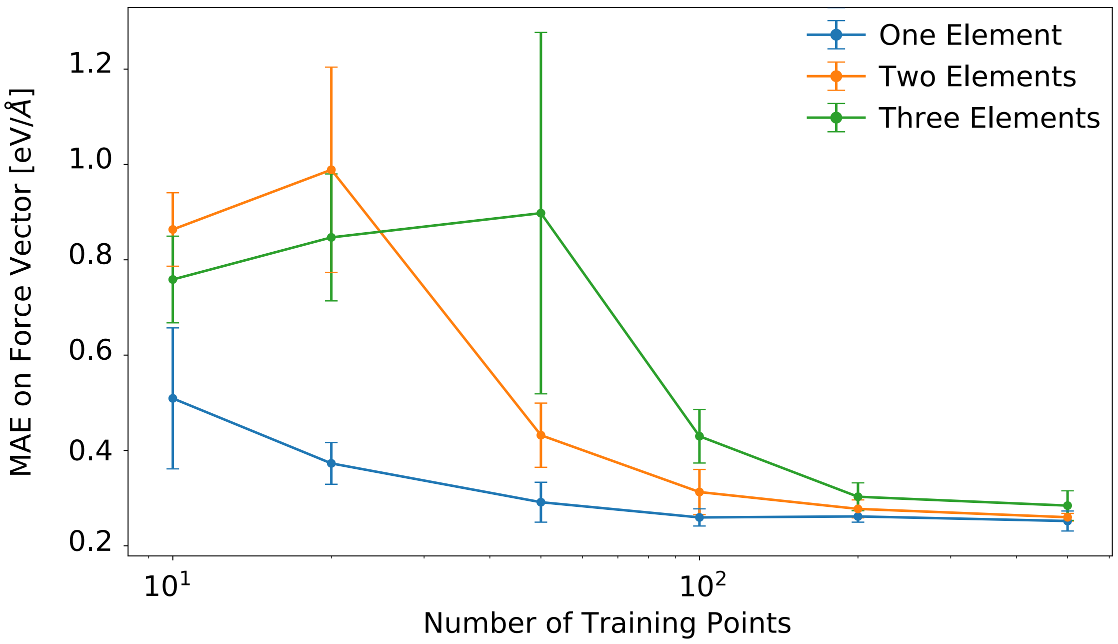


Figure 3.4: Mean absolute error on the force vector as a function of training points for a 2-body GP trained on data combing from an AIMD of Ni_{19} at different temperatures. The three lines indicate the MAE incurred by the GP when trained on the same data where the species of the atoms are randomized so that there is only one unique chemical species (blue line), two (orange line) and three (green line). The error bars represent the standard deviation of the MAE after 5 measurements each on 200 test configurations.

3.1.3 Mapping GP-FFs for Many Elements

A straightforward extension of the mapping procedure to multi-element systems exists and is implemented in the MFF package. To map n-body force fields of multiple chemical species, we map every n-body interaction between unique combinations of the chemical species present in the system. Eq. (2.46) shows that a 2-body GP-FF for a system containing 2 chemical species is equivalent to 3 independent 2-body FFs: one for each of the two homogeneous pair interactions, and one for the heterogeneous pair interaction. In general, for n-body FFs of systems with s chemical species, the number of element-specific FFs contained in a many-element FF is $\binom{n+s-1}{n}$, i.e. the number of combinations with replacements of n objects of s types. These FFs can be independently tabulated into MFFs in the same way that

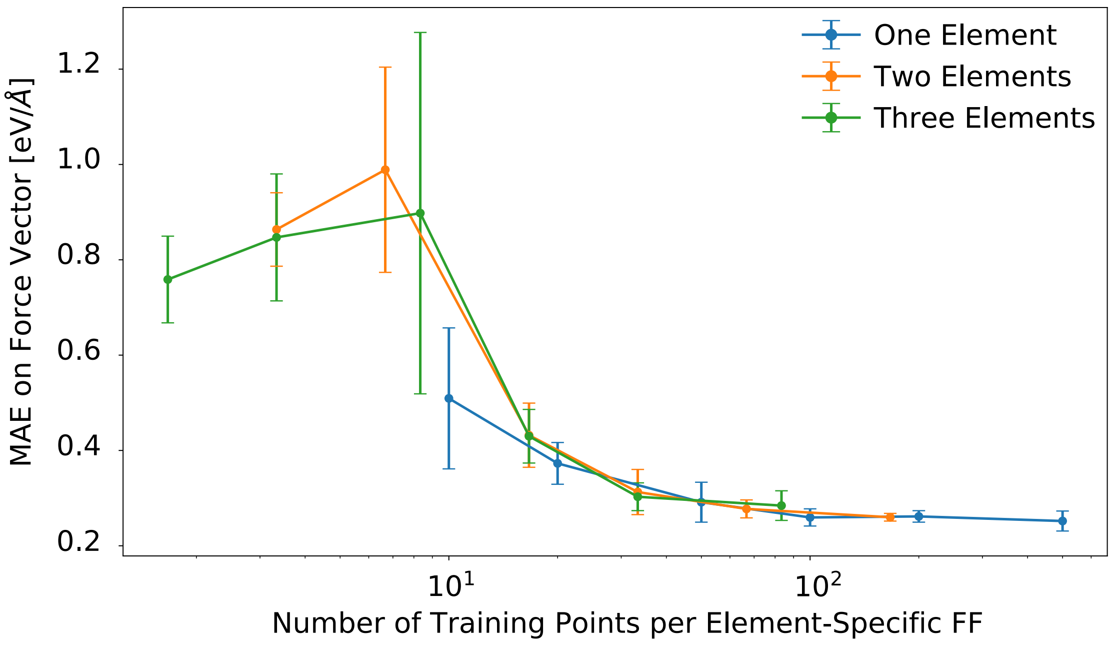


Figure 3.5: The same graph as in Figure 3.4, but the x-axis is normalized so that the average number of training points per element-specific FF is shown. This normalization is obtained by dividing the number of training points of the data displayed in Figure 3.4 by 1 for the one-element tests (blue line), 3 for the two-elements tests (orange line) and 6 for the three-elements test (green line).

single-element FFs are.

The fact that a multi-species GP-FF is equivalent to many element-specific GP-FF has repercussions on the prediction accuracy's convergence rate. A GP-FF is effectively trained for each of the $\binom{n+s-1}{n}$ interactions; therefore the number of training points required to reach a certain prediction accuracy is directly proportional to the number of interactions that need to be modelled. An n-body GP-FF for a system containing s chemical species does require, on average, $\binom{n+s-1}{n}$ times the number of training points to reach a certain accuracy w.r.t. an n-body GP-FF trained on a similar single-element system. Figure 3.4 helps visualize this behaviour for a 2-body kernel trained on *ab initio* data coming from an MD simulation of a Ni_{19} nanoparticle. In Figure 3.4, the atomic species of atoms in the descriptors $\mathbf{q}_2$ were artificially modified, so that the system would contain Ni atoms only (blue line), Ni and Ga atoms (orange line), and Ni, Ga and Mo atoms (green line). It can be seen that, for the same underlying force training set, the presence of multiple elements increases the number of FFs that have to be learnt to incur given MAE, and therefore shifts the learning curve by a factor $\binom{n+s-1}{n}$ w.r.t. a single-element system. This shift can be more easily observed in Figure 3.5, where the number of training points per element-specific FF is shown on the x-axis.

3.2 Algorithm Structure

The transformation of raw reference data into M-FFs which are then used within the ASE Python package [121] to perform MD simulations happens in five distinctive steps within the MFF module. These steps are represented as rectangles in Fig. 3.1 and handle, in order: preprocessing of reference data, sampling of the training set, training of the GP, mapping of the GP-FF, and usage of the mapped FF as a force and energy calculator in ASE MD simulations. In this section, we briefly describe each of the five main steps together with example code calls to illustrate the practical use of MFF. The MFF package incorporates functions from the following Python packages: Numpy [122], Scipy [123], ASE [121], Asap, Theano [124] [1] .

3.2.1 Preprocessing

The reference data used by the MFF package must be imported as a list of snapshots containing the positions, species, energies, and forces acting on atoms during an MD simulation. These snapshots are not required to be ordered, to belong to the same simulation, or even to describe the same system. After a cutoff radius has been chosen by the user, the first transformation applied to this raw data is the extraction of local atomic environments ρ_i and associated forces $\mathbf{f}_i$ for every atom i in every snapshot S_j contained in the input trajectory file; this data is stored in a force dataset $\mathscr{D}_f = \{\rho_i, \mathbf{f}_i\}$. Also, the total energies E_j relative to each snapshot S_j

[1]The Numpy package is used in MFF to speed-up operations on vectors, while Scipy is used for matrix inversion, ASE to handle MD simulations and atomic environments, Asap to quickly compute atomic neighbourhoods, and Theano is employed for its automatic differentiation so to compute derivative kernels.

are stored, alongside the list of local atomic environments $\mathbf{P}_j = \{\rho_i\} \; \forall \; i \in S_j$, in an energy dataset $\mathscr{D}_E = \{E_j, \mathbf{P}_j\}$. Both datasets are saved into a single file containing additional information about the chemical species present and the cutoff used to generate the local atomic environments.

All of this is called when using the MFF Python package as:

```
from mff import configurations
data = configurations.generate_and_save(filename, cutoff)
elements, lae, forces, energies, gae =
    configurations.unpack(data)
```

where `filename` is a path to the file containing the reference calculations, `cutoff` $= r_c$, in Å, `lae` $= \{\rho_i\}$, `forces` $= \{\mathbf{f}_i\}$, `gae` $= \{\mathbf{P}_j\}$, `energies` $= \{E_j\}$, and `elements` contains the list of which atomic species are present in the dataset.

3.2.2 Sampling

Once the database has been saved, a training set must be sub-sampled to train a GP-FF. This can be done at random or using a sampling technique to try to reduce the error incurred by the GP-FF on force and energy predictions via an appropriate choice of the data points to incorporate in the training set. In the current version of the MFF Python package, four sampling techniques are supported: random, 2-body descriptor sampling, 3-body descriptor sampling, and common neighbour analysis (CNA) sampling. For details on the sampling methods available in MFF, the interested reader is referred to Appendix A.1.

The sampling of training points from a database is called in the MFF package as:

```
from mff import utility
lae_tr, forces_tr = utility.get_training_set(lae, forces,
    elements, ntr, method, cutoff, cna_cutoff)
```

where lae_tr and forces_tr indicate, respectively, the local atomic environments, and the forces to be used for training, ntr $= D_{tr}$, method is a string indicating which method to use for sampling, and cna_cutoff is an optional argument specifying r_{CNA}.

3.2.3 Training

Within the MFF package, the training, fitting and tabulation of a GP-FF are handled by the models module, which calls the gp and kernels modules when appropriate. The user must first choose which kernel to employ: 2-body, 3-body, 2+3-body, EAM, 2+3-body + EAM, or many-body, and to set the appropriate hyperparameters, such as kernel length scale σ and noise associated to the data λ. To fit the model, in this example a 2-body kernel, on a force dataset, the syntax is then:

```
from mff import models
gp = models.TwoBodySingleSpeciesModel(elements, cutoff,
    sigma, noise)
gp.fit(lae_tr, forces_tr, ncores)
```

where sigma $= \sigma$, noise $= \lambda$, and cores indicates the number of cores to use for the training process. The GP-FF can also be trained using energy and force data using the following command:

```
from mff import models
gp = models.TwoBodySingleSpeciesModel(elements, cutoff,
    sigma, theta, noise)
gp.fit_force_and_energy(lae_tr, forces_tr, gae_tr,
    energies_tr, ncores)
```

Predictions on unseen local and global atomic environments can then be computed with the following commands:

```
from mff import models
new_forces = gp.predict(new_lae, cores)
new_energies = gp.predict_energy(new_gae, cores)
```

3.2.4 Mapping

Once a GP-FF that showcases an accuracy on force and/or energy prediction that is deemed satisfactory has been trained, the mapping procedure is applied to improve the computational speed of predictions. The user must specify the grid density for the FF tabulation and the smallest interatomic distance that has to be tabulated; the upper limit to the tabulated interatomic distance is the cutoff radius of the local atomic environment descriptor. The command used to map GP-FFs is the following:

```
gp.build_grid(grid_start, npoints_grid, cores)
```

where `grid_start` is the minimum interatomic distance to consider, in Å, and `npoints_grid` indicates the number of grid points to use for each dimension of the tabulation grid. In the case of EAM-like GP-FFs, the FF is tabulated for descriptor values that range from three times the lowest descriptor value found in the training set to 0, since $q_{i,\mathrm{EAM}} \leq 0 \ \forall i$. This seemingly arbitrary choice of mapping grids' lower bound is necessary as q_{EAM}'s values are non-trivially dependant on the choice of r_0.

3.2.5 Calculator

The calculator is necessary to run MD simulations within the ASE package using M-FFs. This module is used to calculate the total energy and the forces acting on a system during MD simulations, at a computational cost that is $\mathcal{O}(NM)$ for 2-body and EAM-like M-FFs, and $\mathcal{O}(NM^2)$ for 3-body M-FFs; where N is the total number of atoms in the system and M the average number of atoms in a local atomic environment. We use the following commands to assign a 2+3-body many-species M-FF calculator to a system of ASE atoms:

```
from mff import calculator
calculator = calculators.CombinedManySpecies(cutoff,
   elements, gp.grid_2b, gp.grid_3b)
atoms.set_calculator(calculator)
```

where `grid_2b` and `grid_3b` are, respectively, the 2- and 3-body interpolation mapped potential spline functions.

3.3 Algorithm Performance

We now test the accuracy w.r.t. reference methods and the computational efficiency of GP-FFs and M-FFs on a variety of single-element and multi-element systems. In this and the following chapters, we denote as "3-body" an FF that contains both a strictly 2-body FF trained on reference data and a strictly 3-body corrective FF trained on the difference between the 2-body FF's predictions and the reference data. The final predictions yielded by such FF are then obtained as the sum of the strictly 2-body predictions with the strictly 3-body predictions. Such a GP-FF is denominated "combined" in the MFF package; we choose to call it 3-body in this work for simplicity's sake.

3.3.1 Accuracy Tests

We focus on the error incurred by GP-FFs and, therefore, M-FFs on the prediction of atomic forces for a variety of reference systems. The analysis of the training curves yielded by different kernels does also provide useful information regarding the nature of interatomic interactions present in the reference dataset. The following results on Ni bulk, Fe bulk, C and amorphous Si refer to GP-FFs trained on force data only, we, therefore, use the mean absolute error (MAE) on the force vector as an appropriate error metric. The inclusion of a small number of total energy data would reduce the error on the predicted energy while affecting only minimally the MAEs incurred on the force vectors, as showcased in Fig. 2.2. The many-body GP-FF which performance is displayed in Figs. 3.6, 3.7, 3.8, 3.9 employs a unique many-body kernel (and is therefore not mappable); it is here used to benchmark the performance of other, mappable, GP-FFs. Such many-body kernel is indeed a universal approximator, i.e. in the limit of infinite training points it can approximate the quantum forces to arbitrary accuracy; its specific formulation can be found in A.2.

Fig. 3.6 reports the errors incurred by 2-body, 3-body and many-body kernels on a Ni FCC bulk system where the training data is coming from an AIMD simulation carried out at 500 K. In this case, the 2-, 3- and many-body kernels all incur

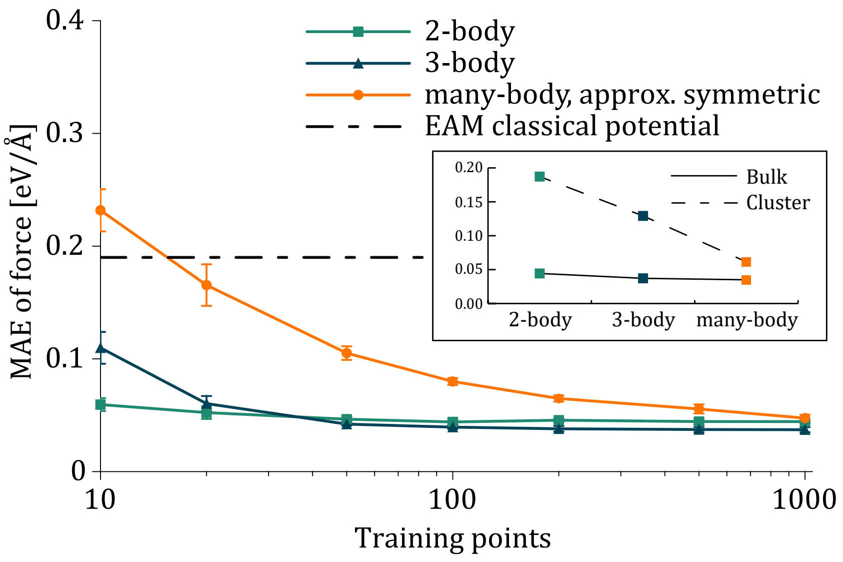


Figure 3.6: (Main) MAE incurred by GP-FFs w.r.t. reference force vectors as a function of the number of training points used to train the GP-FF. The system is FCC Ni coming from AIMD simulations carried out at 500 K using DFT. Different colors indicate different kernels and descriptors used in the GP-FFs. (Inset) comparison of the convergence MAE on forces for 2-, 3- and many-body GP-FFs in bulk Ni and a Ni_{19} nanoparticle. Original figure first published in [65]

MAEs on force vectors which are below 0.10 eV/Å, a threshold value below which we consider the GP-FF to be accurate for bulk reference data. The 2-body GP-FF is, therefore, the best choice to model the atomic interactions in this particular system, as it is the simplest ML-FF that satisfies our requirements (i.e. MAE on force vectors < 0.10 eV/Å).

The 2-body GP-FF is, however, often not complex enough to well reproduce the reference data. This can be seen in Fig. 3.7, where the 2-body kernel is not able to capture the angular features of the interactions in BCC Fe containing a vacancy, and therefore incurs MAE of ~ 0.15 eV/Å at convergence. In this particular system, the 3-body GP-FF instead incurs MAE of ~ 0.08 eV/Å at convergence and is our choice to model the system as it is the GP-FF of lowest order that is still able to reach the target accuracy. The low error incurred by the 3-body GP-FF yields chemical insight regarding the nature of atomic interactions in this particular system: bond and angular interactions are responsible for most of the contributions

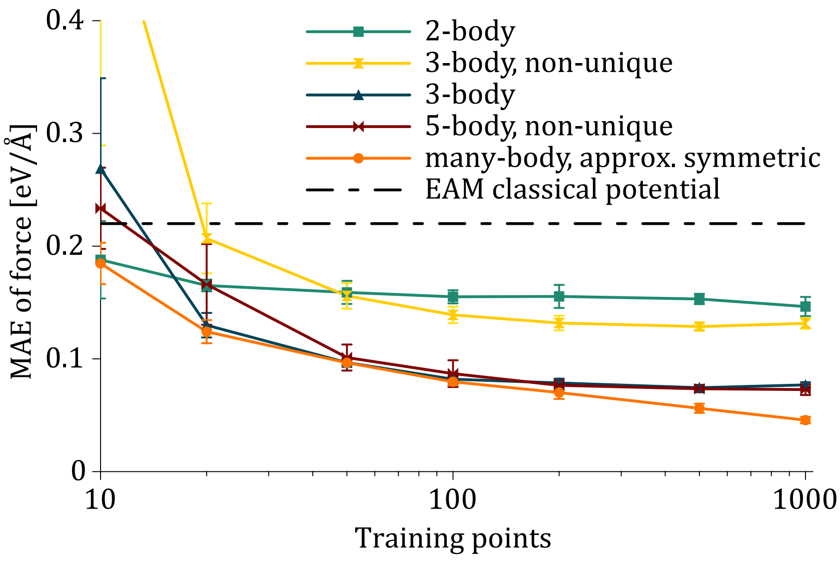


Figure 3.7: MAE incurred by GP-FFs w.r.t. reference force vectors as a function of the number of training points used to train the GP-FF. The system is BCC Fe containing a vacancy coming from AIMD simulations carried out at 500 K using DFT. Different colors indicate different kernels and descriptors used in the GP-FFs. Original figure first published in [65].

to quantum forces in the *ab initio* reference data. Fig. 3.7 also reports the training curves for non-unique 3- and 5-body kernels, obtained as per Eq. (2.39);. These non-unique n'-body kernels incur higher MAEs than their unique counterparts. In particular, the non-unique 3-body kernel in Fig. 3.7 displays a behaviour that is half-way between the one of the corresponding unique 3-body kernel, and the 2-body kernel that has been used to generate it following Eq. (2.39).

Fig. 3.8 displays the training curves of GP-FFs for a system containing different phases of C: graphite, bulk diamond structure and amorphous. In this case, no kernel can reach MAEs below 0.10 eV/Å with the number of training points investigated, and the best-performing GP-FF is the 3-body, which incurs convergence MAE of 0.13 eV/Å at 500 training points. Therefore, the 3-body is the kernel of choice in this system; if more data points would be available and usable, then a higher-order GP-FF would probably incur lower convergence error [93].

Fig. 3.9 reports the training curves for a system containing amorphous Si.

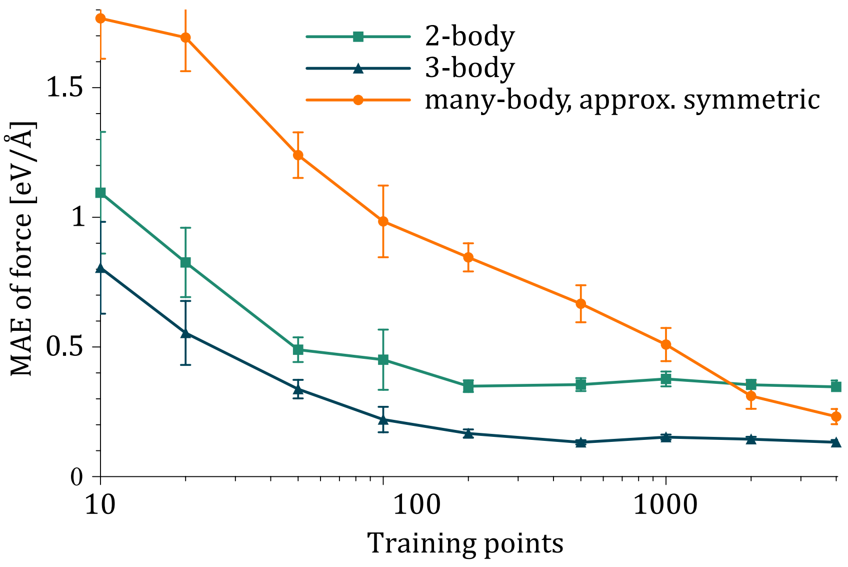


Figure 3.8: MAE incurred by a GP-FF w.r.t. reference force vectors as a function of the number of training points used to train the GP-FFs. The system contains amorphous C, graphite and bulk diamond coming from AIMD simulations carried out using DFT at various temperatures and from single-point force evaluations. Different colors indicate different kernels and descriptors used in the GP-FFs. Original figure first published in [65].

Here, 2- and 3-body GP-FFs fall far from the accuracy target, with MAEs at convergence of 0.35 eV/Å and 0.23 eV/Å, respectively. Only the many-body GP-FF comes close to the target, incurring in a MAE of 0.13 eV/Å when trained on 4000 training points. Therefore, in amorphous Si, bond and angular information is not sufficient to capture the relevant interactions, and 4- and higher-body terms are required to well capture the behaviour of quantum forces.

3.3.2 Speed-Accuracy Comparisons

As showcased in the previous paragraph, M-FFs trained on *ab initio* reference calculations (such as DFT), incur non-zero errors on the training set, but are much more computationally efficient than the reference method. They, therefore, offer an alternative to classic P-FFs for the simulation of systems where explicit quantum calculations are prohibitive. It is therefore imperative to compare the accuracy w.r.t. reference data and the computational cost of such GP-FFs, M-FFs and P-FFs. In Fig. 3.10 we report a scatter plot where the MAE on forces w.r.t. reference data is

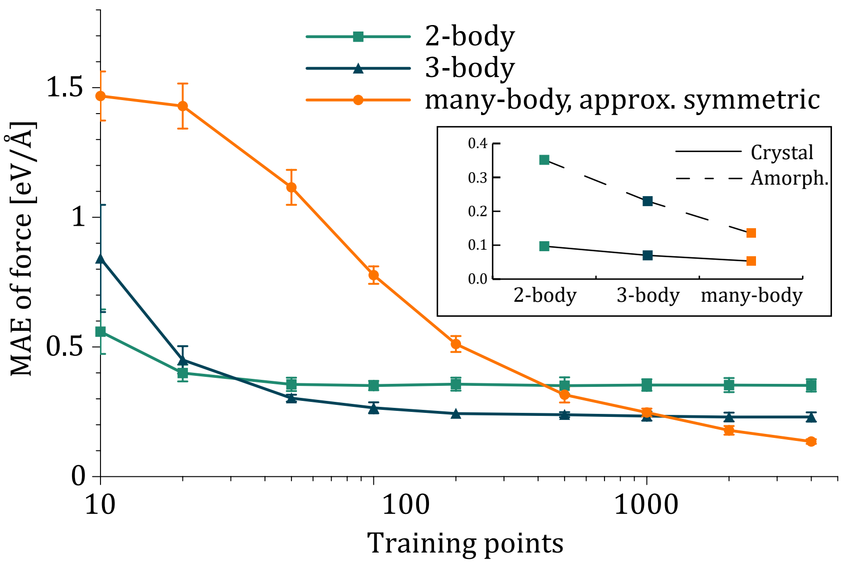


Figure 3.9: (Main) MAE incurred by a GP-FF w.r.t. reference force vectors as a function of the number of training points used to train the GP-FFs. The system contains amorphous Si extracted from AIMD simulations carried out at 650 K using DFT. Different colors indicate different kernels and descriptors used in the GP-FFs. (Inset) comparison of the convergence error for 2-, 3- and many-body GP-FFs in amorphous and crystalline Si. Original figure first published in [65]

plotted against the computational time required to compute the forces on a system of 566 atoms. We do so for the reference method (which obviously reports zero error on itself), a 2+3-body+EAM GP-FF, the M-FF built from such GP-FF, and six EAM classical potentials for Fe taken, in order, from [128, 129, 130, 131, 132, 133]. The reference data [2] comes from DFT single-point calculations, and comprises Fe with a vacancy, bulk Fe containing a symmetric tilt grain boundary with a [111] orientation axis and a [211] boundary plane, and bulk Fe containing a symmetric tilt grain boundary with a [100] orientation axis and a [120] boundary plan and a misorientation angle of 53.13° [134]. It is evident from Fig. 3.10 that the M-FF is as fast as classical EAM potentials while incurring in a MAE w.r.t. reference DFT data that is almost half of the one incurred by the best-performing EAM potential.

[2]We thank Dr. Henry Lambert, Physics Department, King's College London, for providing this reference quantum dataset.

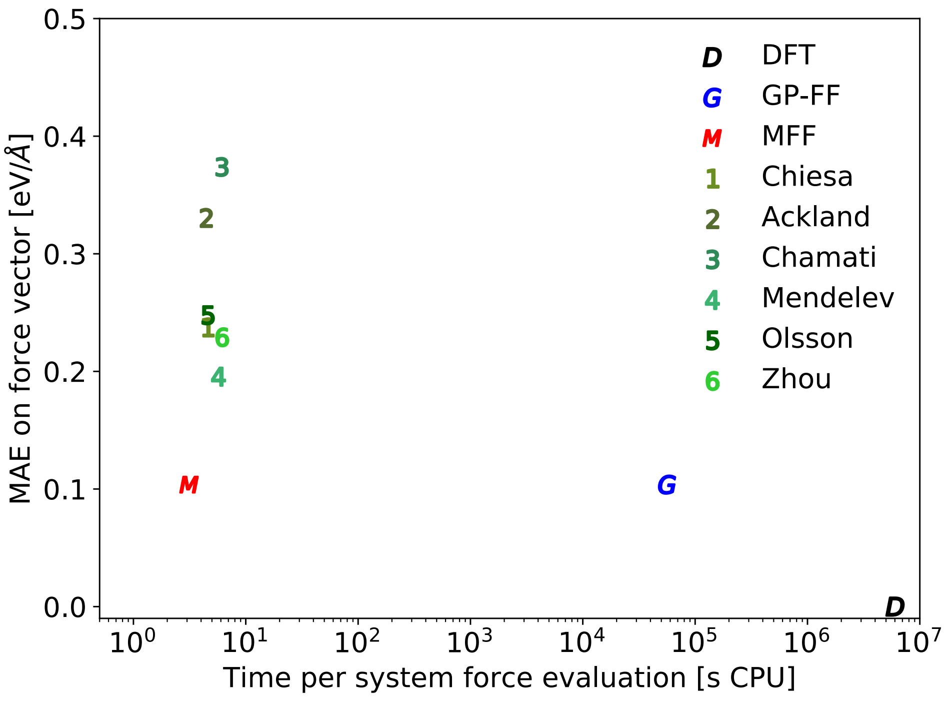


Figure 3.10: Comparison of the computational cost and MAE incurred in force prediction w.r.t. DFT calculations for a Fe system of 566 atoms containing a grain boundary. The GP-FF shown is a 2+3+EAM GP-FF, trained using 500 local atomic environments and forces. The MFF is the mapped version of the above mentioned GP-FF, where 10^6 grid points were used for the mapping procedure. The six EAM potentials displayed were taken, in order, from [128, 129, 130, 131, 132, 133].

Chapter 4

Machine Learning Force Fields for Nanoparticles

ML-FFs have emerged in recent years as a way to reduce the computational requirements of force and energy evaluations for MNPs. The speed-up offered by ML algorithms enables thorough searches of the configurational space of systems and extends the time scale accessible to accurate simulations from the order of ps (i.e. using DFT) to possibly μs using M-FFs. The state-of-the-art for applications of ML-FFs to MNPs is briefly reviewed in Sec. 4.1, to better introduce the topic to the reader. Subsequently, in Sec. 4.2, we present the application of GPR to develop FFs that are accurate for different morphologies of Ni_{19} NPs. The training database contains five isomers, and we find a positive correlation between the geometrical complexity of an isomer and the amount of information its inclusion in the training set provides. We furthermore test the accuracy of 2-, 3- and many-body GP-FFs trained on different subsets of the available data and highlight that the best-performing GP-FFs are 3-body ones that include a varied subset of morphologies extracted from AIMD simulations carried out at different temperatures. Afterward, in Sec. 4.3, we build two M-FFs using respectively data coming from low-T and high-T *ab-initio* MD simulations and study the phase changes in Ni_{19}. Employing M-FFs allows us to efficiently run simulations for a total time of 60 ns, a timescale not accessible to methods such as DFT, and to observe the formation of slush states in the temperature range between $\sim$700 and $\sim$900 K.

4.1 State-of-the-art of ML-FFs for MNPs

The applications of ML-FFs to MNPs found in the recent literature can be broadly divided into three categories. The first comprises cases where ML-FFs are used to supplement or substitute *ab initio* calculations for energy evaluations such as global minima structure search [21, 135, 136, 111, 137, 118, 112]. It is in these cases fundamental that the error incurred by ML-FFs on energy predictions is very low, for obvious reasons. The same principle applies to the second class of applications of ML-FFs to MNPs found in literature: the estimation and mapping of activation energies for catalytic property predictions [138, 99, 100]. The third and final category comprises cases where ML-FFs are used to carry out NVE or NVT simulations over time scales not easily accessible to *ab-initio* methods. These works aimed to estimate phase diagrams and/or thermal properties of metallic nanoparticles. For this reason, the error incurred by the ML-FFs on reference quantum forces has to be low, as the correct prediction of dynamical pathways and vibrational behaviour is mandatory to study the dynamical properties of a system. ML-FFs for dynamical applications are therefore trained using either reference force data or reference force and energy data. Most of the studies found in recent literature regarding ML-FFs to run fast MD simulations employed ANNs as the ML framework of choice and used atomic symmetry functions as an input 2+3-body descriptor. These studies trained ML-FFs to simulate Cu nanoparticles supported on zinc oxide [139, 113], AuCu MNPs [114], Au MNPs [106], AgCu MNPs [140] and PtNiCu nanocatalysts [115]. To the author's knowledge, there is only one example in literature where a descriptor of order higher than 3 was used to train ML-FFs to run MD simulations of MNP; in this case, a many-body descriptor (SOAP) was used to train an ANN on total energy data, which was then used to search for global minima of Au_{147} nanoclusters using the basin-hopping method [107].

In all of the above cases, the accuracy and speed of the ANN-FFs allowed the researchers to simulate the systems for timescales inaccessible to DFT methods (i.e. several ns) and/or to thoroughly explore large configurational spaces. This is not by chance, as the computational cost of predicting forces and energies with an ANN-FF is independent of the training set size D and, while not being comparable to classical P-FFs, is still much cheaper than *ab initio* reference methods such as DFT. The mapping of GP-FFs, therefore, presents a new frontier of ML-FFs for MD simulations of MNPs, thanks to the combined advantages of a low computational cost, comparable to that of classical P-FFs, and the ability to train ML-FFs for systems where a small dataset is available.

4.2 GP-FFs for Ni_{19} Nanoparticles

As the first-ever application of M-FFs for MNPs, we report our original study on Ni nanoclusters of 19 atoms. We aim to develop a single M-FF that incurs low prediction errors w.r.t. the reference forces so that fast, classical MD simulations can be carried out as a surrogate for *ab initio* MD.

4.2.1 Dataset Construction

Among the possible isomers available for a Ni_{19}, we focus on five of them, depicted in Fig. 4.1. Three of them, namely the double icosahedron (DIH), the three-layer HCP (3HCP), and the bipyramid with a square base (BIP) have been

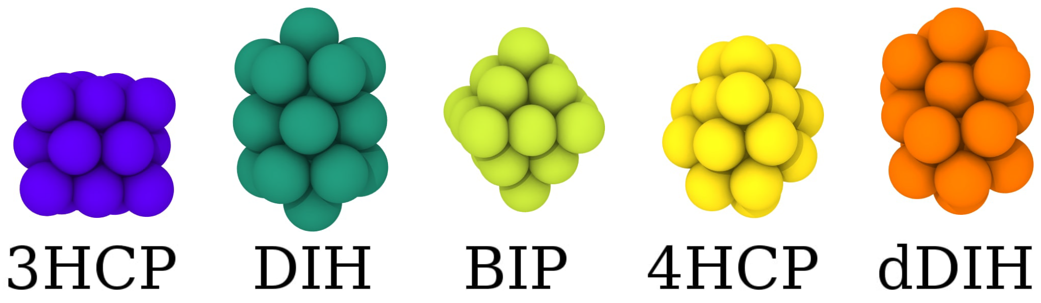


Figure 4.1: Five Ni_{19} structures: 3HCP, DIH, BIP, 4HCP, and dDIH. The structures are ordered, left to right, according to their total energy, with the leftmost structure being the lowest in energy according to VASP PBE calculations. Original figure first published in [84].

chosen according to an energetic criterion. Indeed, they are energetically the putative most favourable geometries in local spin unrestricted DFT-calculations using a PBE exchange-correlation functional. The less symmetric isomers in Fig. 4.1 labelled 4HCP (presenting four HCP layers) and dDIH (a defected double icosahedron with the subsequent formation of a hexagonal ring, reminiscent of the rosette defects found in Pt MNPs [141]) have been discovered using a metadynamics sampling scheme [56, 142], and have been included due to their low symmetry. To generate a set of reference quantum structures and forces, Born-Oppenheimer *ab initio* MD simulations were performed within the VASP package, using a plane-wave energy cutoff of 270 eV, and an energy convergence cutoff of 10^4 eV [1]. The exchange-correlation functional was calculated employing the PBE/GGA approximation [143], and spin-orbit coupling effects are accounted for. [2] The *ab initio* MD simulations employ a Nose-Hoover thermostat to control the temperature for NVT runs at 300, 600 and 900 K; the time step is set to 3 fs and the total simulation time is approximately 100 ps. We monitor the structural evolution of each isomer at during these simulations using standard geometrical quantities to check whether the initial structure is maintained and to eventually classify new isomers.

4.2.2 Validation Methodology

When considering which kernels to try for the system, we first observe that for FCC bulk Ni systems, 2-body GP-FFs are found to be incredibly accurate (See Fig. 3.6) [24]. We, therefore, choose to try the 2-body kernel of Eq. (2.37) and a unique approximately symmetric, many-body kernel (See appendix A.2 to build GP-FFs in the first place. The many-body GP-FF we here employ cannot be transformed into an M-FF but, given that it is a universal approximator, its accuracy is not constrained by the simplifying assumption that the reference forces can be decomposed into n-body contributions, with n finite. It is therefore used here as a benchmarking tool to obtain an estimate of the error on force prediction that is introduced by

[1]We thank Dr. Kevin Rossi, Laboratory of Computational Science and Modelling, École Politechnique Fédérale de Lausanne, for providing this reference quantum dataset

[2]Because the spin of the system is not fixed, the learnt GP-FF is trained on a weighted average of the potentials encountered in the spin states visited throughout the AIMD simulations.

the finite cutoff radius of local atomic environment descriptors, and the one that is caused by the noise present in the training set (see Eq. (2.18)). After some initial testing, we realise that the 2-body GP-FFs are not capable of capturing the relevant physics of the system and do not meet our target accuracy of MAE on forces, here set to 0.15 eV/Å. We, therefore, go to the next level of complexity and employ the 3-body kernel of Eq. (2.38) to build GP-FFs that reach the accuracy target and are therefore used to build M-FFs. For comparison, a state-of-the-art classical P-FF developed for Ni [144] incurs MAE on the reference forces of 0.59 ± 0.39 eV/Å.

To assess the errors incurred by the kernels while used in interpolation and (putative) extrapolation regimes, we analyse the GP-FFs force predictions on test databases containing the five morphologies displayed in Fig. 4.1. We do so for GP-FFs that have been trained on data sets containing one, two, three and all five of the geometrical motifs of Fig. 4.1. We, therefore, introduce a novel way of measuring the similarity between cluster geometries, based on the errors incurred by GP-FFs which are trained on data coming from MD simulations containing a single morphology and tested on data containing another one. We choose a test set that contains 400 randomly-selected local atomic environments for each of our five cluster morphologies, yielding a total of D_{test} = 2000 test points. Every test is repeated five times to estimate a standard deviation for the MAE on the force vector. The tests we carry can be separated into three categories: *same-isomer*, *cross-isomer*, and *mixed-isomer*, depending on which databases were used to build the training sets and which subset of the testing pool is used; this is depicted in Fig. 4.2. In the *same-isomer* tests, the data used to build the $\mathscr{D}_{tr}$ and $\mathscr{D}_{test}$ are extracted from the same morphology. In the *cross-isomer* tests, $\mathscr{D}_{tr}$ contains data coming from a single morphology, which is different from the one used to generate $\mathscr{D}_{test}$. Finally, for the *mixed-isomer* tests we build and test all possible heterogeneous training sets $\mathscr{D}_{tr}$ that contain inputs from two, three, and all five motifs. When $\mathscr{D}_{tr}$ contains data coming from all five morphologies, this will be equivalent to a *same-isomer* test. In all of the above cases, no data point present in $\mathscr{D}_{tr}$ is allowed to be in $\mathscr{D}_{test}$, and data

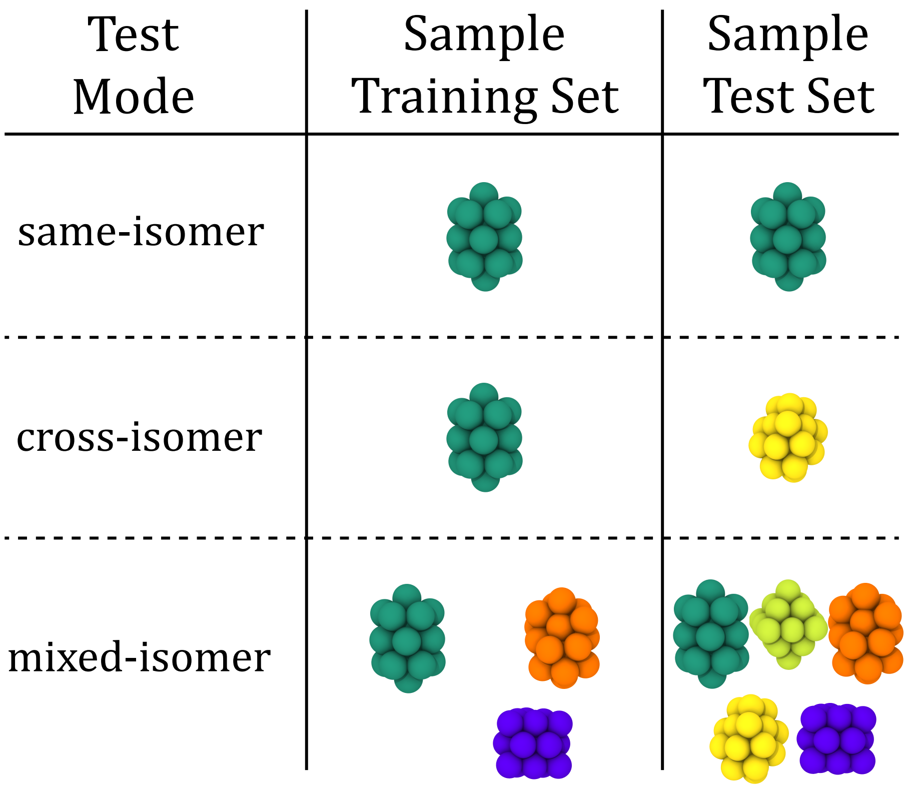


Figure 4.2: Schematic representation of the three test modes used to validate M-FFs for five Ni_{19} morphologies, coloured as in Fig. 4.1.

points are selected at random for each dataset.

4.2.3 Same- and Cross-Isomer Accuracy Tests

We first discuss the results for *same-isomer* tests; the accuracy results will provide a benchmark for the accuracies reachable by GP-FFs trained using $\mathscr{D}_{tr}$ that do not contain solely the structures they are tested on. Figure 4.3 reports a learning curve (MAE on forces against D_{tr}) for the example case of a 3HCP structure. The 2-body GP-FF achieves its maximum accuracy for any $D_{tr} > 50$; similarly, the 3-body MAE on forces decreases with D_{tr} until $D_{tr} > 100$, where an accuracy plateau is reached. The accuracy of the many-body GP-FF, on the other hand, always increases with D_{tr}; this is expected as the many-body kernel is a universal approximator [102, 65]. The learning curves for *same-isomer* tests in other structures show the same trends; this can be seen in Fig. 4.4. Figure 4.5 displays the MAE at convergence obtained on *same-isomer* tests by GP-FFs when using 2-, 3-, and many-body kernels, for the

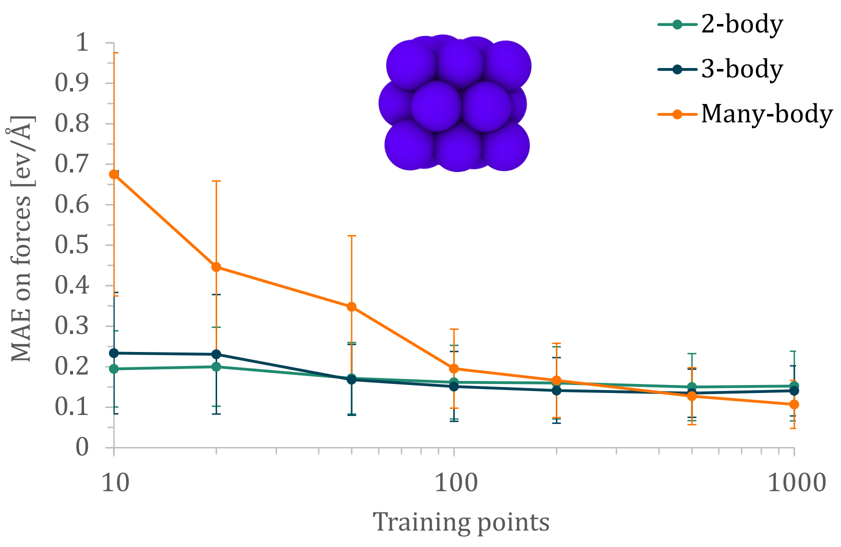


Figure 4.3: MAE on force vectors as a function of the number of training points employed for training and testing on a Ni_{19} 3HCP morphology. The GP-FFs' order is colour coded, with 2-body represented in teal, 3-body in blue, and many-body in orange.

five morphologies considered. For the many-body kernel, the error incurred at D_{tr} = 2000 was used instead of the convergence error, as the many-body GP-FF's accuracy keeps increasing with the training set size. The left-hand histogram shows that using a 2-body potential to model the atomic interactions in Ni_{19} yields an MAE larger than the target accuracy of 0.15 eV/Å for all motifs but 3HCP. Incidentally, this is the structure that most resembles bulk FCC Ni, where 2-body GP-FFs were found to be surprisingly accurate. The other two histograms in Fig. 4.5 reveal that both 3- and many-body GP-FFs achieve a force prediction for all cluster structures that satisfies the sought accuracy when $D_{tr} \geq 200$.

The accuracy of n-body kernels reveals insightful information about the relative importance of n-body contributions to the forces in the five Ni_{19} cluster morphologies. For instance, in the 3HCP morphology, the MAEs of 2-body and 3-body forces are very similar; this indicates that the angular dependence of forces is not crucial in this structure. This is indeed not true for other structures such as 4HCP and dDIH, where the 3-body MAE is significantly lower than the 2-body one; an-

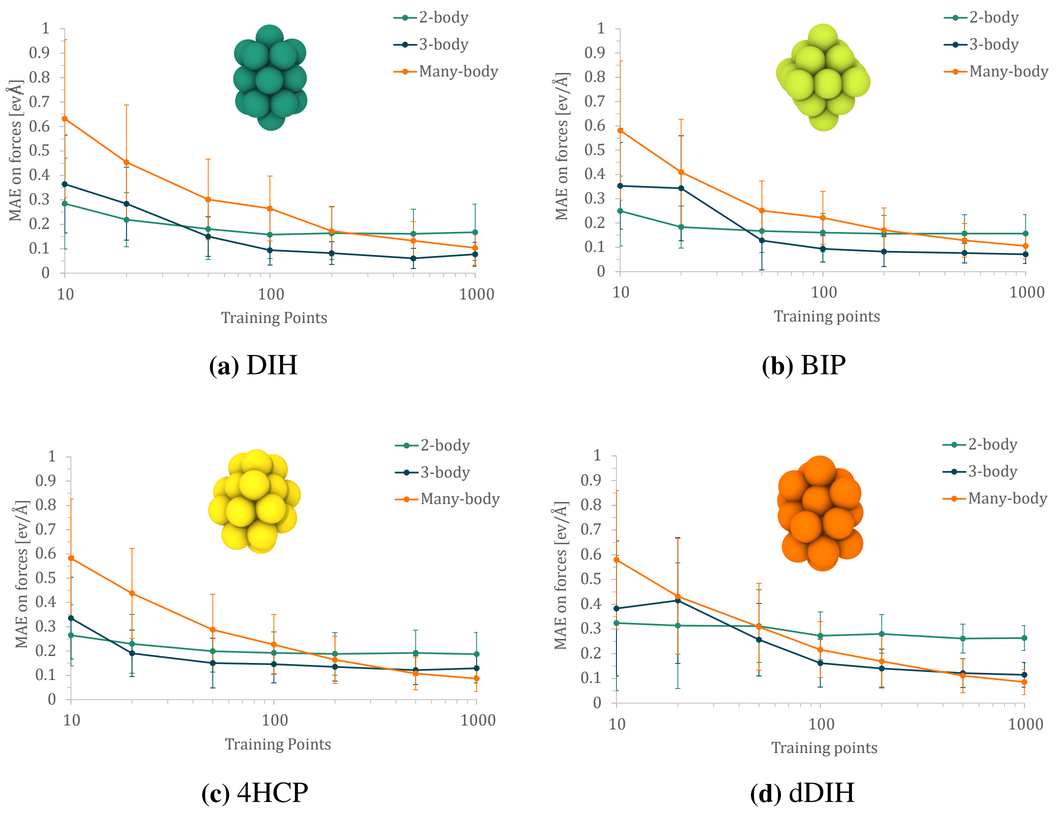


Figure 4.4: MAE on force vectors as a function of the number of training points employed for 2-, 3- and many-body GP-FFs. *Same-isomer* tests on DIH (a), BIP (b), 4HCP (c) and dDIH (d) are shown. The GP-FFs' *n*-body order is colour coded, with 2-body represented in teal, 3-body in blue, and many-body in orange.

gular interactions must be included to obtain an accurate prediction of forces in these motifs. We note that the comparison of *n*-body GP-FFs' errors could be generally used as a tool to characterize the nature of interatomic interactions in complex systems such as MNPs or grain boundaries, and to highlight and measure (dis)similarities both among these systems and relative to bulk structures.

The MAEs on force vectors obtained for *cross-isomer* tests are reported in Figure 4.6; the graphs include also the results for *same-isomer* tests. In these cases, the 2-body GP-FFs incur MAEs consistently higher than the reference error of 0.15 eV/Å. By comparing the 3- and many-body GP-FFs, we notice that the accuracy achieved by the 3-body GP-FFs strongly depends on the training morphology, while the many-body GP-FFs incur more consistent errors across different

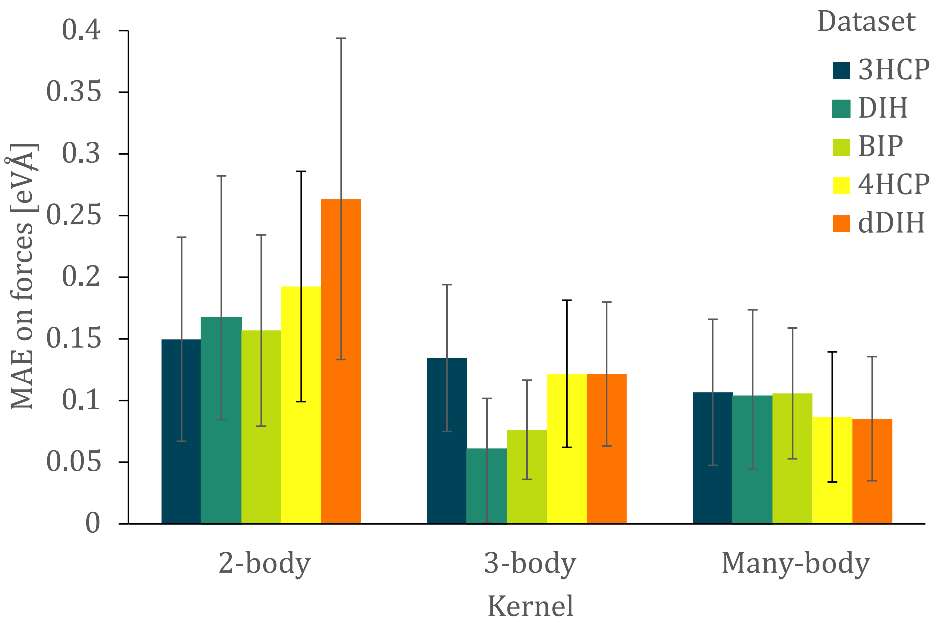


Figure 4.5: MAE on force vectors incurred by GP-FFs at convergence for *same-isomer* tests for each of the 5 morphologies in Figure 4.1. The performances of the 2-body, 3-body, and many-body GP-FFs are reported. For all tests, N_{test} = 400 and N_{tr} = 500 (2- and 3-body GP-FFs) or N_{tr} = 1000 (many-body GP-FFs).

structures. This non-trivial observation could be rationalised by arguing that a many-body kernel can learn high-n interaction terms whose contributions are effectively encountered in every data set, even e.g. in structures where their contribution to the force prediction is less important. These high-n interaction terms may help to retain a good prediction accuracy also on "partially unknown" new morphologies, where higher-order contributions have increased importance (e.g. in low-symmetry motifs such as 4HCP and dDIH). The 3-body kernel is instead restricted by design to 3-body interactions; if the reference forces include e.g. a 4-body contribution, the GP-FF will learn to "fold" this higher-dimensional interaction into the 2- and 3-body terms by over-fitting a correction to the true underlying 2+3-body contributions for the training data where a 4-body interaction is present. This incorporation of high-order interactions may achieve some success only in *same-isomer* (interpolation) tests, where the test configurations share structural similarities to the training structures the GP-FF was fitting on, but will not correctly extrapolate to new structures. This suggests that the accuracy of a 3-body GP-FF on a target

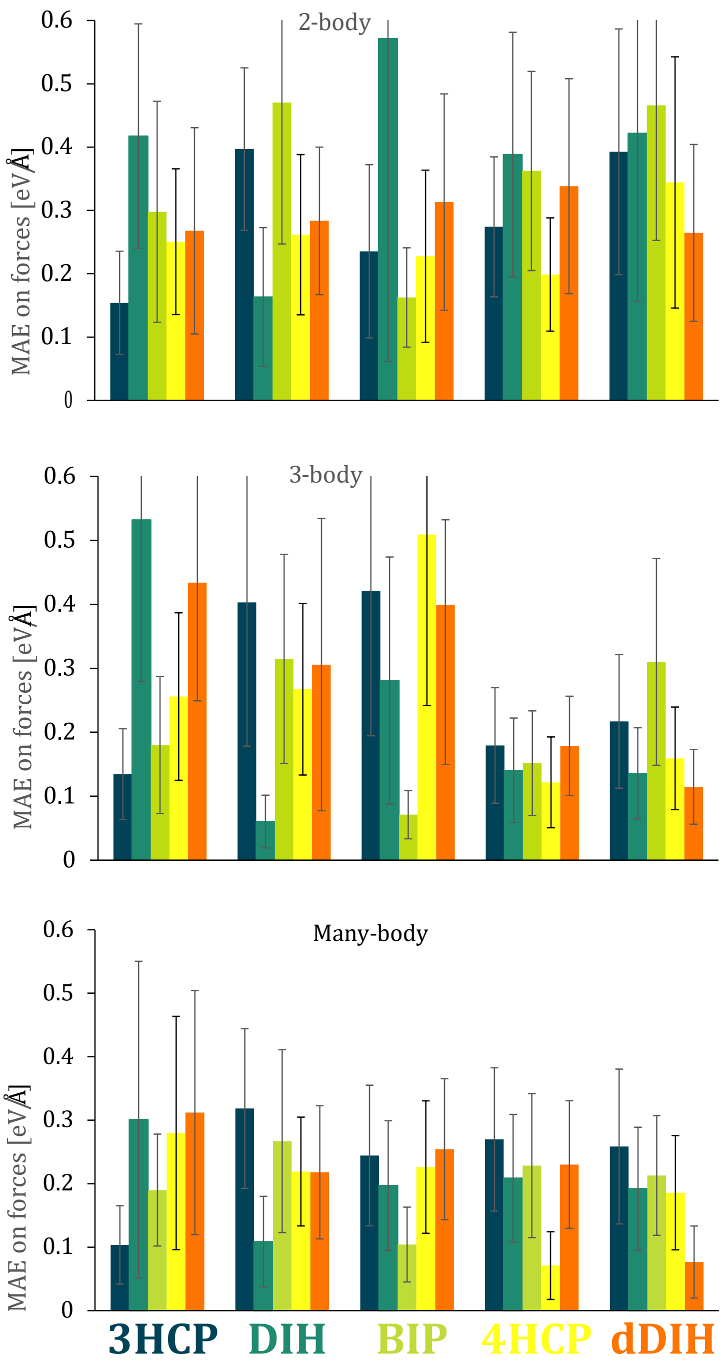


Figure 4.6: From top to bottom, the MAE on force vectors and its standard deviation for the 2- ($N_{tr} = 500$), 3- ($N_{tr} = 500$) and the many- ($N_{tr} = 1000$) body GP-FFs trained and tested on different Ni_{19} datasets at 300 K. The label color refers to the isomer contained in the training set, the bar color indicates the isomer present in the test set; they are consistent with the colors used in Fig. 4.1.

input is significantly conditional to the presence of points in the training database which are representative of that specific input. Consistently, for 3 body GP-FFs,

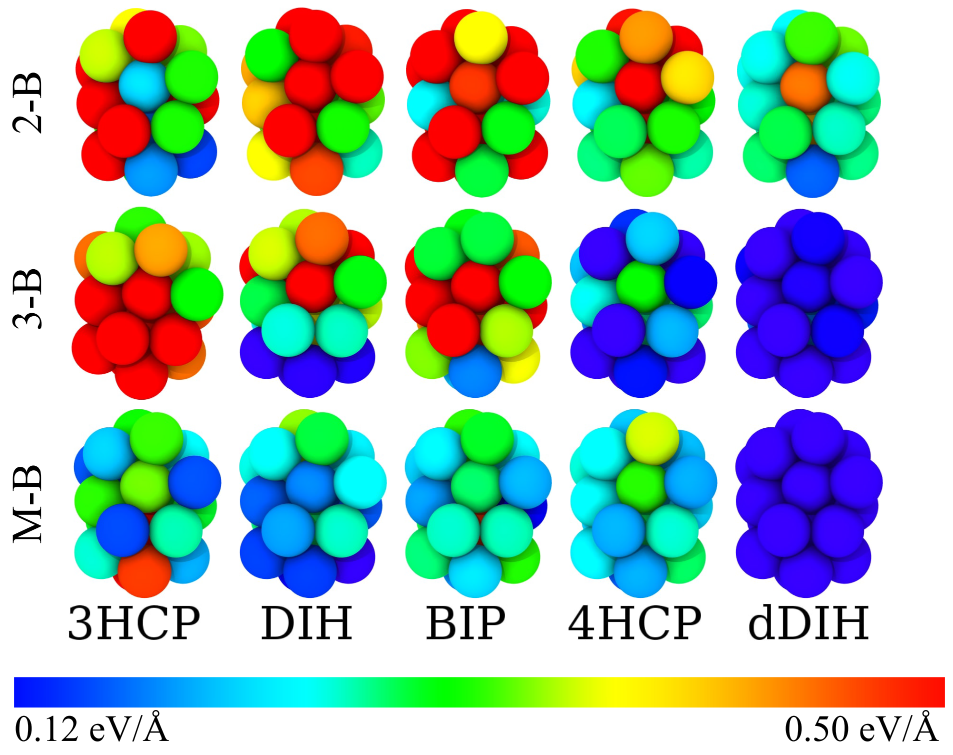


Figure 4.7: MAE on force vectors incurred by the 2-(top), 3-(central), and many-body (bottom) GP-FFs on every atom comprising dDIH when trained on (left to right): 3HCP, DIH, BIP, 4HCP, dDIH. The MAE represented via color-coding ranges from 0.12 eV/Å (blue) to 0.5 eV/Å (red). Original figure first published in [84]

the training databases of the low-symmetry geometries (4HCP and dDIH) that have the most varied local atomic environments are those which incur low MAEs in *cross-testing*. We also observe that the *cross-testing* MAE incurred by training on 4HCP and testing over its higher-symmetry counterpart 3HCP is 0.18 eV/Å, while the opposite test yields a larger MAE of 0.26 eV/Å. This behaviour is even more evident when we look at the dDIH (low symmetry) and DIH (high symmetry) pair of geometries, where the errors for the direct and reversed tests are 0.14 eV/Å and 0.31 eV/Å, respectively.

Further analysis of the GP-FFs' accuracies yields some qualitative understanding of why the use of different training databases results in larger or smaller accuracies over the testing sets. We look in detail at the case of training on each of the five morphologies and testing on dDIH for different kernels: figure 4.7 contains a visual representation of the MAEs incurred in testing the 2-, 3- and many-body GP-FFs on every atom of dDIH. As expected from Fig. 4.6, the 2-body GP-FFs incur large errors for all training sets but dDIH - which is in this example in *same-isomer* testing. For the 3-body GP-FF, training on a database composed of data coming from 4HCP offers the best overall *cross-isomer* test result (while, not surprisingly, *same-isomer* training on a dDIH database yields better accuracy). Such a GP-FF incurs a low MAE on most atoms of dDIH, falling short only around the rosette defect, a distortion present only within the dDIH structures. The accuracy displayed by GP-FFs trained on the DIH database is also good for the lowermost 5-fold cap of the dDIH structure; this is to be expected as local atomic environments containing 5-fold caps can be encountered in both datasets. *Cross-isomer* training using local atomic environments coming from the BIP and 3HCP datasets instead fails to accurately predict forces around the two icosahedral centres and the rosette defect of dDIH. These results are mirrored for the many-body kernel, where the DIH-trained GP-FF is the best performing one (excluding the GP-FF employing the dDIH dataset).

4.2.4 Heterogeneous Training and Training Set Optimisation

We now examine the *mixed-isomer* tests, to analyse how a small training dataset $\mathscr{D}_{tr}$ which still includes a sufficiently varied ensemble of 2- and 3-body local atomic environment descriptors (i.e. bond distances and angles) can be constructed. To do so, we test the accuracy of training on datasets that contain data from two, three, and all five different cluster geometries. Our results indicate that the MAE on forces incurred by the 3-body GP-FFs is rather homogeneous in all of the various *mixed-isomer* tests, presenting a standard deviation across all tests of 0.03 eV/Å, contrary to what was found for the *same-isomer* tests. This trend suggests that introducing some variety in the training dataset allows achieving a reasonably complete training of a 3-body GP-FF, avoiding over-fitting caused by the misrepresentation of

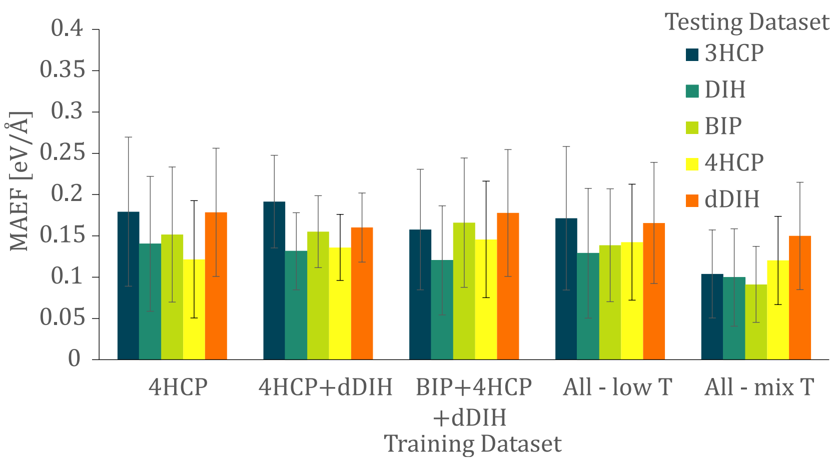


Figure 4.8: Convergence MAEs on forces for four 3-body GP-FFs trained on data coming from *ab initio* MD simulations carried out at 300 K and comprised of 500 data points representative of one, two, three and all five cluster geometries, and a "all - mix T" database containing 1000 training points that include all cluster morphologies, extracted from AIMD simulations carried out at different temperatures (300, 600 and 900 K).

higher-order interactions pertinent to a particular morphology. Consistently, the MAE on forces incurred by the many-body kernel, a kernel that is not restricted to 3-dimensional feature space and is therefore much harder to fully train, presents a slightly higher standard deviation (0.05 eV/Å) in the same scenarios. For all GP-FFs, we find in general that *mixed-isomer* tests yield errors comprised between those incurred in *same-isomer* and *cross-isomer* tests. The MAEs on forces are therefore slightly higher than those incurred in while in *same-isomer* tests but significantly smaller than those associated with *cross-isomer* ones.

Figure 4.8 depicts the error incurred by the 3-body GP-FFs trained on the "best" single-morphology database (4HCP), the best two- and three-morphologies mixed databases, the full 5-morphologies database ("all - low T") containing 500 training points, and another 5-morphologies database ("all - mix T") that includes 1000 data points extracted from AIMD simulations carried out at 300, 600 and 900 K. The inclusion of a training set that contains high-temperature data is justified by the increased amount of internal database variability present; which has been observed to be a positive contributor to a GP-FF's accuracy. The accuracies displayed meet the target of MAE on forces < 0.15 eV/Å for all training frameworks and notably, the 5-morphologies "all - low T" dataset showcases the same accuracy of all the other database choices which had to be pinpointed as the best restricted ones. The overall best accuracy is achieved when using the "all - mix T" training dataset, which displays an average MAE on forces of 0.11 eV/Å, lower than the one encountered by the second best-performing dataset, "all - low T": 0.14 eV/Å.

We now analyse the influence on a GP-FF's accuracy of the exact composition of its training set. To do so, we create 100 independent "all - low T" training databases, each containing a total of 500 local atomic environment descriptors: 100 for every morphology, randomly selected. The average MAE on forces incurred by 3-body GP-FFs trained on these "all - low T" datasets when tested on a fixed dataset containing all five morphologies is 0.14 ± 0.07 eV/Å. We find a very small difference of 0.004 eV/Å between the MAE on forces incurred by the best-performing 'all - low T" 3-body GP-FF and the average value.

The above result suggests that the accuracy gain obtainable by a careful and guided choice of the best possible training database is practically negligible in this scenario. To further investigate this issue, we perform Metropolis Monte Carlo simulations on the choice of points to include in the training set, to try to find a training set which reports lower errors. The quantity we choose to sample in this Monte Carlo procedure is a linear combination $X(\mathscr{D}_{tr})$ of the MAE on forces and the standard deviation (STD) of the absolute error on forces, both taken as functions of the training set $\mathscr{D}_{tr}$:

$$X(\mathscr{D}_{tr}) = \text{MAE}(\mathscr{D}_{tr}) + 2\ \text{STD}(\mathscr{D}_{tr}). \tag{4.1}$$

We first sample and keep fixed a test set $\mathscr{D}_{test}$, used to test the accuracy of all generated GP-FFs. We then initialize a training database of D_{tr} randomly-chosen data points and carry out Metropolis steps exchanging database entries with new ones randomly chosen from our complete dataset, and using the metric defined in Eq. (4.1). This process typically yields a MAE on forces that is lower by just 0.001 eV/Å than the one of a randomly sampled training set, when using $D_{tr} = 200$ data points to train the many-body GP-FF; this discrepancy is further reduced with increasing D_{tr}. The accuracy gain for this rather exhaustive procedure is therefore negligible.

4.2.5 Analysing Similarity Metrics: PDF and BADF Distances

We now compare the pair-distance function (PDF) and the bond angle distribution function (BADF) of the five morphologies as computed using trajectory data of the AIMD simulations carried out at 300 K, reported in Fig 4.10 in color. These distributions encode information regarding the structural differences between the geometries. For example, the PDF peak present at 3.3 Å in the 3HCP, 4HCP, and BIP structures disappears for the DIH and dDIH motifs. The BADF in the bottom panel of Fig. 4.10 displays wider peaks for 4HCP and dDIH and much sharper ones for 3HCP, DIH, and BIP.

We employ the Kullback−Leibler (KL) divergence as a quantitative measure of how well a PDF "samples" another one. For two discrete probability distributions, namely P and Q, this is defined as:

$$\mathrm{KL}(P||Q) = \sum_i P(i) \log \frac{P(i)}{Q(i)}. \tag{4.2}$$

The KL divergence can be thought of as an asymmetric measure of the information "lost" when using a function Q to approximate another function P. This measure returns 0 when the two functions are identical, i.e. $P = Q$, and a positive number that increases as P grows dissimilar from Q. Since we want to evaluate the ability of a GP-FF trained on a morphology to predict forces on another morphology, Q and P are in our example the PDFs associated with the training and test set, respectively. In Fig. 4.9 we display a scatter plot that compares the KL divergence calculated be-

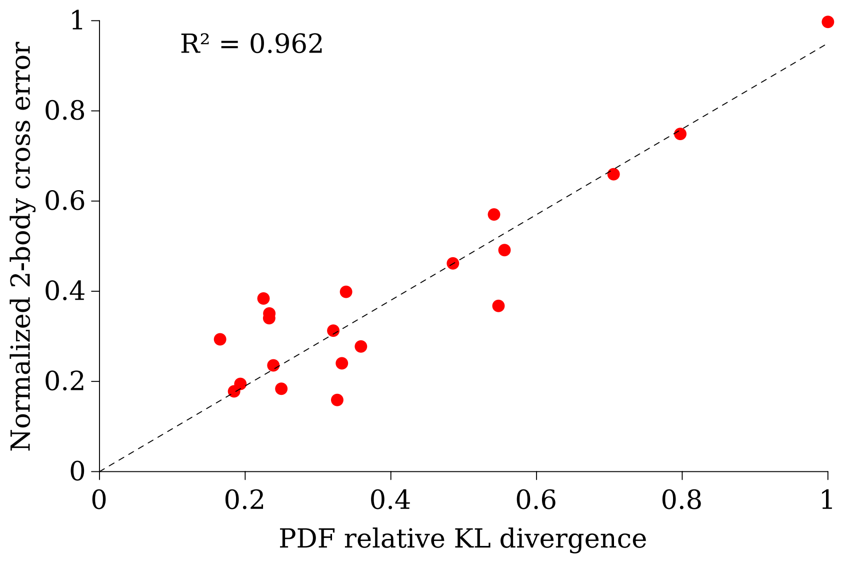


Figure 4.9: Scatter plot highlighting the correlation between the normalized KL divergence calculated on ordered pairs of binned PDFs from Fig. 4.10 and the *cross-isomer* test MAEs on force vectors incurred for the corresponding pairs of train-test morphologies. A linear fit (black dashed line) and the corresponding R^2 coefficient are displayed. Original figure first published in [84].

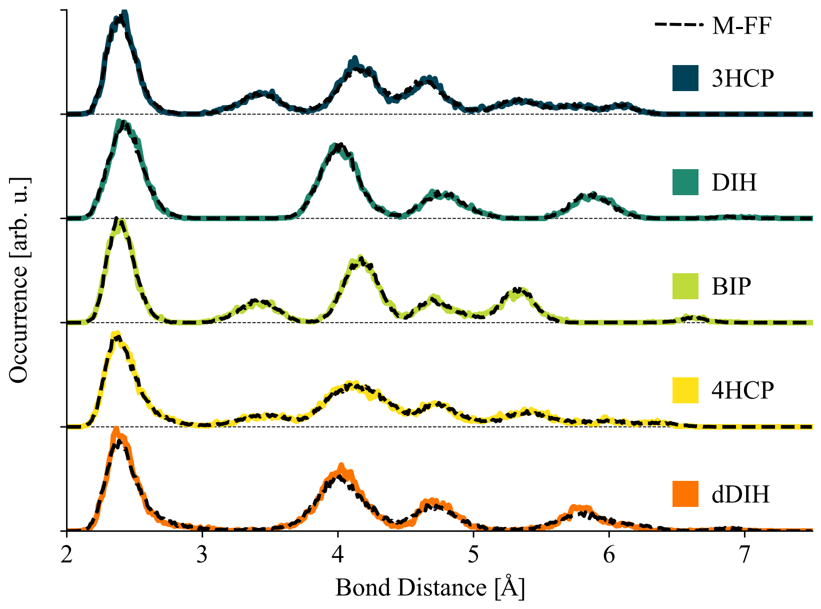
tween each ordered pair of PDFs of from Fig. 4.10 (normalized so that the maximum value is 1) with the corresponding MAE on forces incurred by the 2-body kernel for *cross-isomer* tests (also normalized so that the maximum value is 1). A correlation between the two dissimilarity measures emerges from the graph. This highlights the importance of the presence of training database entries which contain pairs of atoms at distances that are relevant for the local atomic environment descriptors found in the test set. Moreover, since the PDF can be assumed to act as a unique structural descriptor in the case of monometallic nanoparticles [145, 146, 147, 56], the presence of a correlation in Fig. 4.9 indicates that the 2-body *cross-isomer* test error is non-trivially connected to features that go beyond 2-body descriptors. For this reason, the KL divergence between PDFs could be used as an *a priori* estimate of the expected accuracy of 2-body kernels in *cross-isomer* tests. For completeness sake, we carried out similar tests for the 3-body kernel, looking at the KL divergence between the BADFs and the 3-body MAEs. These scatter plots also display positive correlation, but the R^2 index is too low to consider these results to be statistically significant. These results suggest that evaluating the KL divergence for functions other than the PDFs or BADFs could provide an array of structural dissimilarity estimators. These could be then used to guide the creation of informed, minimally sized training databases from a "general" database too large to be used *in toto* for GP regression. Indeed, this idea has been incorporated in the MFF package via the "descriptor sampling" algorithms, which use the presence of unique bond distances (or angles) to guide the inclusion of local atomic environment descriptors in the training set (See Appendix A.1 for more details). Such algorithms have been tested for Ni_{19} but no significant accuracy gain w.r.t. random sampling methods was observed for 2- and 3-body GP-FFs.

4.3 M-FFs for Ni_{19} Nanoparticles

To perform computationally cheap MD simulations with near-*ab initio* accuracy we map the two best-performing GP-FFs from Sec. 4.2 ("all - low T" and "all - mix T") onto non-parametric 3-body M-FFs, following the procedure introduced in Sec. 3.1.

We first test their accuracy on data extracted from 600 K and 900 K AIMD simulations, where the starting structures were set to be 3HCP and DIH, respectively. In both cases, the Ni_{19} nanoparticle undergoes several structural rearrangements during the course of the AIMD simulation. The computed MAE on forces for the "all - low T" M-FF is 0.26 ± 0.24 eV/Å for 3HCP at 600 K and 0.25 ± 0.17 eV/Å for DIH at 900 K. These values indicate at the "all - low T" M-FF retains an acceptable accuracy level on configurations not represented in its training dataset. We should also point out that higher MAEs should be expected for high-T samples since forces have larger moduli at higher temperatures. The MAE on forces incurred by the "all - mix T" M-FF is instead 0.25 ± 0.46 eV/Å for 3HCP at 600 K, and 0.17 ± 0.09 eV/Å for DIH at 900 K.

To check whether M-FFs trained on finite temperature data coming from dynamical simulations can capture information relevant to the minima structures of the nanoparticle, we checked their accuracy on force predictions for minimized 0 K



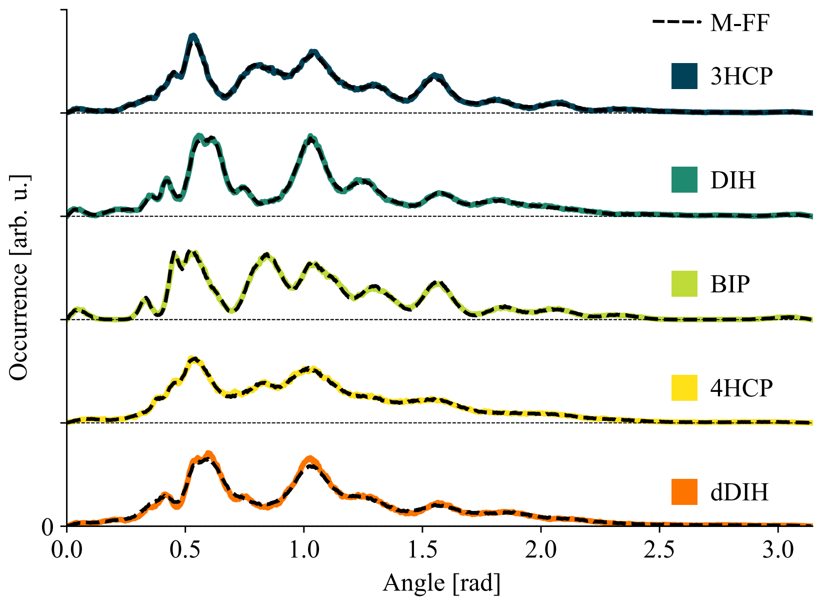


Figure 4.10: Pair-distance (top) and bond angle (bottom) distribution functions for the five morphologies, averaged over 2 ps of AIMD (colour) and M-FF (black dashed) MD simulations carried out at T=300 K. Original figure first published in [84]

structures. The two M-FFs incur MAE of 0.10 $\pm$ 0.02 eV/Å and a 0.06 $\pm$ 0.02 eV/Å for the “all - low T” and the “all - mix T”, respectively. The inclusion in the training databases of data collected during structural relaxations reduces the MAE on forces to 0.04 $\pm$ 0.02 eV/Å for both M-FFs.

To further inquire about the capability of our 3-body “all - low T” M-FF to faithfully reproduce dynamical behaviour observed in AIMD, we run three 200 ps-long 300 K MD simulations for each of the five morphologies. We then compare the observed PDFs and BADFs with the reference ones extracted from equally long AIMD simulations of the same geometries, also at 300 K. The almost perfect overlap obtained for both PDFs and BADFs, observable in Fig. 4.10, provides further validation of the ability of the M-FF to reproduce the dynamical behaviour of reference *ab initio* methods.

While the M-FFs we developed were trained with the aim of evaluating dynamical properties and therefore using force data only, it is still of interest to assess their accuracy w.r.t. reference total energy differences. For the ”all - low T” M-FF, the MAE incurred on energy predictions over the five structures at 300 K is 16 ± 10 meV/atom; this figure is reduced to 9 ± 7 meV/atom for the ”all - mix T” M-FF. The M-FFs are therefore satisfyingly accurate when predicting total energies despite not being trained on total energy data.

4.3.1 Assessing the Thermal Behaviour of Ni_{19}

To assess whether M-FFs can be used to gain novel physical insights into the system’s behaviour, we now study the kinetic evolution of Ni_{19}, exploring the extent of shape fluctuations occurring in the nanoparticle as T varies [58, 56, 61, 57]. To this end, we run NVT classical MD simulations within the ASE Python environment [121] at 50 K-spaced temperature intervals between 300 K and 1200 K using a time step of 3 fs and a Langevin thermostat, with damping set to 0.001. For both M-FFs, we perform four 480 ps-long simulations for each temperature starting from each isomer presented in Fig. 4.1. We run for a total of 4 (runs) $\cdot$ 5 (starting geometries) $\cdot$ 19 (temperature intervals) $\cdot$ 480 ps = 182 ns over 380 simulations. The use of M-FFs rather than DFT allows us to gain a computational factor of $\sim 10^6$ in these MD simulations, bringing the total time required to simulate on 24 cores from an estimated 2000 years to 4 days.

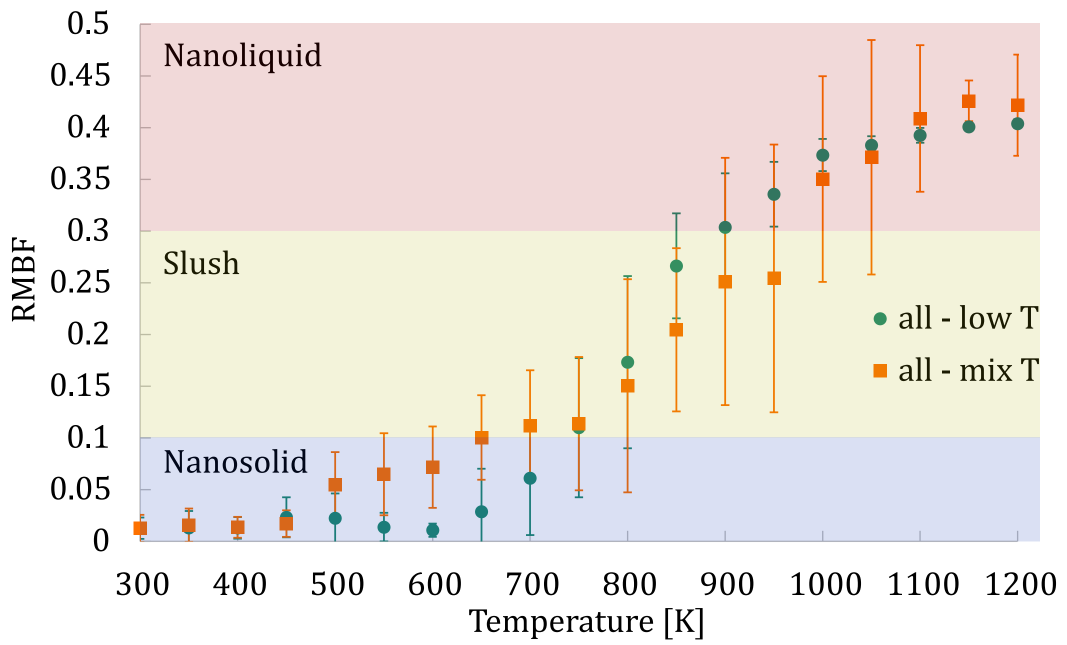


Figure 4.11: RMBF value against the nominal simulation temperature for two 3-body M-FFs; the error bars display the standard deviation over 20 simulations. Blue, yellow, and red shadowing are used to highlight the characteristic values of RMBF for nanosolids (< 0.1), slushes (0.1 - 0.3), and nanoliquids (> 0.3).

To detect when phase changes occur in such small systems is not trivial. Here we opt to calculate the root mean bond fluctuation (RMBF), which describes the average interatomic distance oscillation at a given temperature [154]. The RMBF is defined as:

$$\mathrm{RMBF} = \frac{2}{N(N-1)} \sum_{i<j} \frac{\sqrt{\langle r_{ij}^2 \rangle - \langle r_{ij} \rangle^2}}{\langle r_{ij} \rangle}, \tag{4.3}$$

where N is the number of atoms in the system and the average $\langle \bullet \rangle$ is taken over the whole simulation trajectory. The RMBF is often used to highlight phase changes in nanoscale systems [148, 149, 150, 151, 152, 153]; following the existing literature, the first few ps (here 5 ps) of every simulation here excluded from the RMBF calculation to allow for thermal equilibration of the system. Figure 4.11 displays the so-calculated RMBF values averaged over 20 (5 clusters and 4 repetitions each) simulations for each T as a function of T for MD simulations carried using the “all - low T” and the “all - mix T” M-FFs. The standard deviations of the average RMBF are also indicated in Figure 4.11 as error bars. Independently on the training set, we observe from Fig. 4.11 that the phase change is not sharp and there is a temperature window where the MNPs are in an intermediate state between solid and liquid; such window refers to the formation of slush-states [154]. Both a M-FF that was expected to operate in a largely extrapolatory regime (i.e. all - low T") and a M-FF that was instead built with a dataset that included high-T configurations (i.e. all - mix T"), more directly relevant to the MD simulations that were carried out, yield consistent RMBF trends.

In more detail, for temperatures below 650 K, all of the analysed Ni_{19} remain solid independently on the starting configuration and on the choice of the training set, as indicated by the small (< 0.1) RMBF. Within this region, the "all - mix T" M-FF displays a significantly non-zero RMBF; this indicates that small geometrical changes were present in the MD simulations for some of the starting geometries. A RMBF > 0.3, characteristic of nanoliquids [154], is observed for temperatures above 900 K in MD runs employing the "all - low T" M-FF, and in a similar manner for temperatures greater than $\sim$975 K while using the"all - mix T" M-FF. In the intermediate, $\sim$700-900 K temperature interval, the observed RMBF for the Ni_{19} nanoparticle is associated with an intermediate phase between the nanosolid and nanoliquid regimes. The prediction of a "slush" intermediate state is also consistent with the high probability of geometrical rearrangements that have been discussed thoroughly for small MNPs [154, 155].

Chapter 5

Transferability of GP-FFs for MNPs

The large dimensionality of the effective configurational space of nanoparticles, where the number of atoms, chemical composition, chemical arrangement and geometry all constitute degrees of freedom for the system's structure and behaviour, are responsible for their strong structure-property relationship. In the context of FF development, this poses a huge challenge, as developing *ad hoc* classical P-FFs that are accurate for a wide size range, different geometries and chemical compositions of MNPs is extremely resource-intensive. Gaussian process FFs are an ideal candidate to address the task, as they require small training datasets which can, therefore, be produced on demand for a large subset of system sizes and geometries, and very little user input once the kernel functions are chosen. Nonetheless, the creation of ML-FFs to model NPs for which quantum reference data cannot be generated (e.g. MNPs with more than a thousand atoms) is still an open challenge. Indeed, to the author's knowledge, no ML-FFs have been yet used to run MD simulations of NPs which size was larger than the ones present in the reference quantum training set.

In this chapter, we address the issue of the size transferability of GP-FFs for MNPs. We touch upon a parallel topic in Chapter 4, where we analyse the development of GP-FFs that are morphology-transferable, i.e. that report good accuracies across different geometrical rearrangements of the same nanoparticle. In this chapter, we take a step further and analyse the accuracy of GP-FFs across reference quantum datasets that contain structures with varying sizes and morphologies. In

Sec. 5.1 we consider small Ni nanoparticles containing 20 or fewer atoms; at these small scales, every atom counts towards the geometrical stability and electronic structure of the system. The creation of an FF that is accurate across even small variations in the number of atoms is therefore complex, but rewarding whenever a single FF can faithfully reproduce the physics of this set of systems. In Sec. 5.2 we instead consider larger size differences, taking additional steps to address the generation of GP-FFs for NPs which size is so large that reference *ab initio* data cannot be generated. We consider in particular Au NPs containing 147, 309 and 561 atoms for which quantum reference data is available, and test the accuracy of 3-body and 3-body+EAM-like GP-FFs across all three systems. Of particular interest is, in this case, the accuracy of GP-FFs which are trained on NPs of a certain size and tested on NPs of a larger size.

In Appendix A.3, we additionally report preliminary results on the application of GP-FFs across the phase diagram of nanoalloys for the case of AuAg with 32 atoms.

5.1 Transferable GP-FFs for Small Ni NPs

As a first case study for the transferability of GP-FFs across multiple MNPs' sizes, we look at a Ni system containing 13, 15, 17, 19, and 20 atoms. The data relative to Ni_{19} is the same as the one employed in the analysis carried on in Chapter 4. The remaining data is generated performing AIMD simulations within the VASP package, using a plane-wave energy cutoff of 270 eV, and an energy convergence cutoff of 10^4 eV [1]. We calculate the exchange-correlation functional employing the PBE/GGA approximation [143], and spin-orbit coupling effects are accounted for. The *ab initio* MD simulations employ a Nose-Hoover thermostat to control the temperature for NVT runs at 300 K; the time step is set to 3 fs and the total simulation time is approximately 100 ps. Figure 5.1 displays the morphologies present in the dataset, which includes the 5 Ni_{19} morphologies already discussed in

[1]We thank Dr. Kevin Rossi, Laboratory of Computational Science and Modelling, École Politechnique Fédérale de Lausanne, for providing this reference quantum dataset

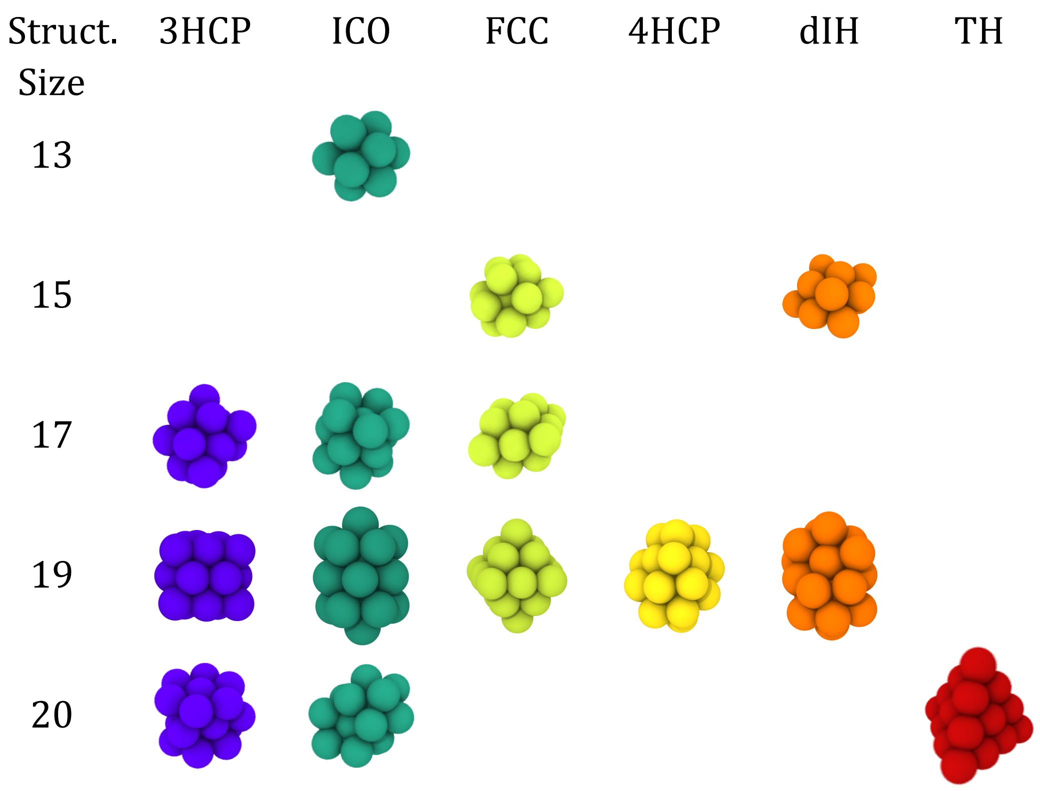
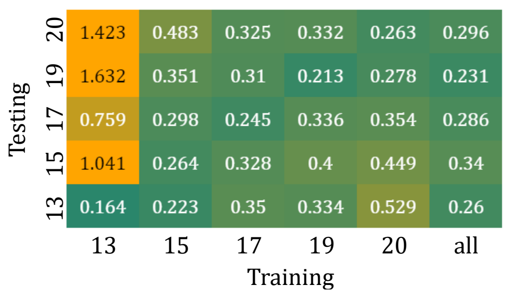


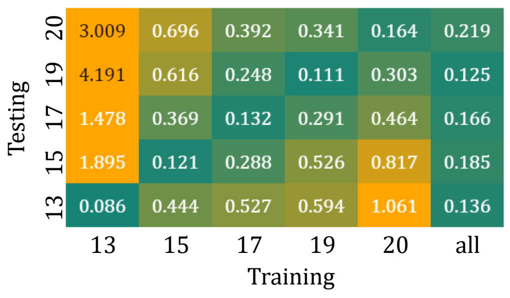
Figure 5.1: Fourteen Ni NP structures arranged vertically according to their number of atoms, and horizontally according to categorical morphology, left to right: 3-layers HCP, icoshaedron or double icosahedron, FCC-like, 4-layers HCP, defected icosahedron, tetrahedron. The color coding reflects the one used in Sec. 4.2.

Sec. 4.2. The geometries included are chosen according to an energetic criterion, and NPs containing an odd number of atoms were favoured to maintain a certain level of homogeneity w.r.t. the electronic structure and magnetic properties of NPs. NPs of size 20 are added to the database as an outlier to the above rule.

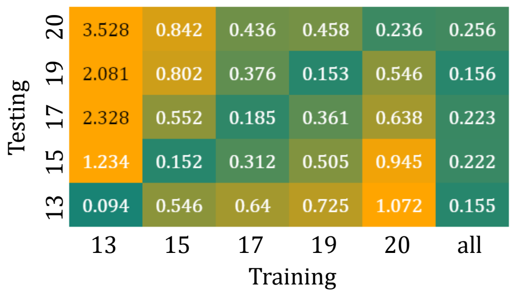
In Sec. 4.2.4 we concluded that a simple but effective recipe to build 3-body GP-FFs that incur low errors across different morphologies is to build a training database that includes a certain level of variety of such morphologies. We, therefore, construct 5 training sets, one for each NP size, that contain data coming from all of the available morphologies at that size. These 5 training sets are used to fit single-size 2-, 3- and many-body GP-FFs, which we then test both in a *same-size* and a *cross-size* regime. To do so, test sets containing 1000 data points per NP



(a) 2-body



(b) 3-body



(c) many-body

Figure 5.2: MAE on force vectors [eV/Å] for 2-body (a), 3-body (b) and many-body (c) GP-FFs trained and tested on databases containing quantum reference data relative to Ni NPs with different number of atoms, and for a database containing 1000 training points randomly sampled from all available sizes (named "all"). The diagonal entries of the matrices refer to *same-size* tests, off-diagonal entries report errors incurred for *cross-size* tests, and the last column refers to a *mixed-size* testing scenario. The standard deviation of the reported errors is not displayed for better visualization of the results, as it is consistently less than 10% of the MAE. MAE values are color-coded, with teal being MAE = 0 and orange being MAE ≥ 1.

size are built and excluded from the training sets; the training and test process is then repeated 5 times to obtain a standard deviation of the MAE on forces. The results of these tests are visible in Fig. 5.2, which reports the MAE on force vectors incurred by 2-, 3- and many-body GP-FFs when built using single-size databases containing 500 randomly-selected training points. The MAEs on forces incurred by all GP-FFs for *same-size* tests are consistently lower than the ones incurred in *cross-size* tests, as expected. We also observe that, in general, the larger the size difference between NPs, the larger the *cross-size* test error. Furthermore, GP-FFs trained on NPs of small size and tested on NPs on large size usually report higher errors than the reverse. This trend is highlighted in Fig. 5.3, where we report the

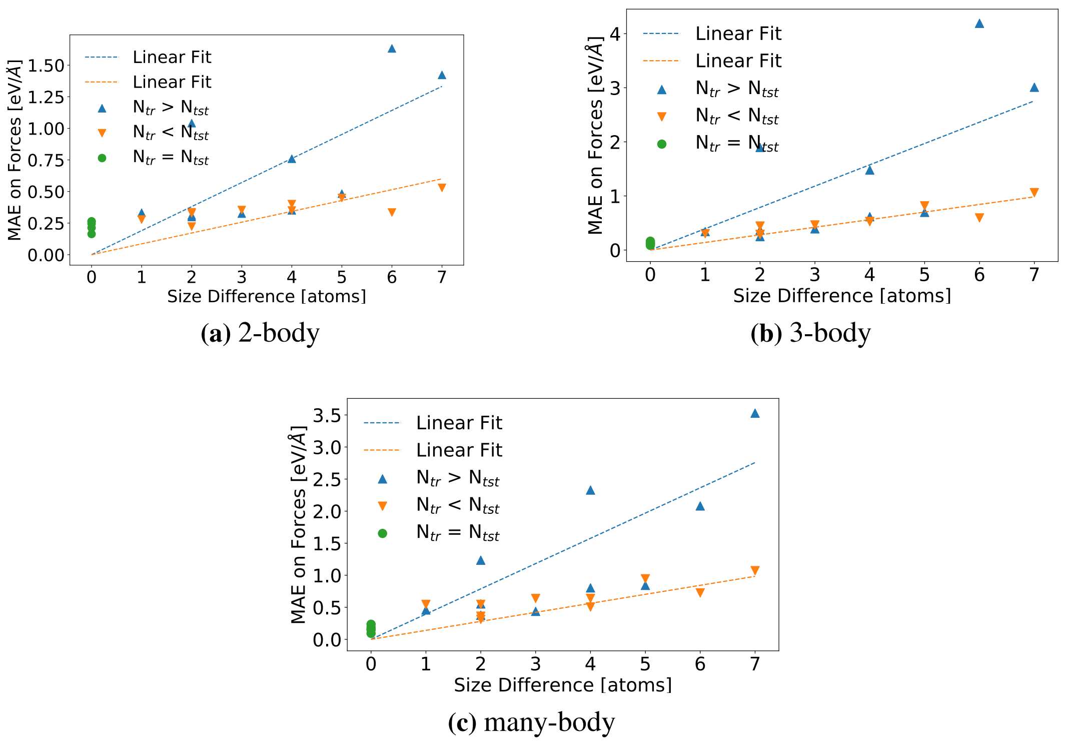


Figure 5.3: Scatter plot displaying the MAE incurred when training on a Ni NP of size N_{tr} and testing on a Ni NP of size N_{tst} as a function of the absolute size difference $|N_{tr} - N_{tst}|$ for 2- (a), 3- (b) and many-body (c) GP-FFs. *Cross-size* tests where $N_{tr} < N_{tst}$ are represented as blue upwards triangles, $N_{tr} > N_{tst}$ as orange downward triangles, and *same-size* tests as green dots. Linear fit lines are also shown in matching colors to highlight the trends; their intercept is set to 0.

MAEs displayed in Fig. 5.2 as a function of the difference in the number of atoms between the structures comprising the training and test set.

With regards to kernel performance, the 3-body GP-FF is the best one in *same-size* tests but falls off in favour of the simpler 2-body GP-FFs for *cross-size* tests. This characteristic was observed also for the case of *cross-isomer* testing in Ni_{19} but is here more prominent. The accuracy displayed by 2-body GP-FFs does not meet the target of MAE $\leq$ 0.15 eV/Å; we must, therefore, rely on 3-body GP-FFs to reach such accuracies. To train 3-body GP-FFs that are accurate for Ni NPs of different sizes we must therefore include a representative size and geometrical diversity in the training set. To avoid the over-fitting of the potential caused by the misrepresentation of size-specific many-body effects as 3-body contributions, we build and test GP-FFs that are trained using an even mix of configurations coming from Ni NPs of different sizes, mirroring the procedure followed in 4.2. The MAE on forces incurred by 2-, 3- and many-body GP-FFs trained on a database named “all”, containing 1000 configurations selected at random from the 5 NP sizes considered, are displayed in Fig. 5.2. The “all” GP-FF incurs errors that are consistently smaller than the ones displayed by GP-FFs in a *cross-size* scenario, and always slightly larger than the ones found in *same-size* testing scenarios; this is true for the 2-, 3- and many-body GP-FFs. The 3-body “all” GP-FF showcases the overall best performance, incurring in an average MAE on forces of 0.166 eV/Å across the five structures, with a maximum value of 0.219 eV/Å for tests on Ni_{20} and a minimum of 0.125 for Ni_{19}.

The above results hint at the possibility that a reliable and accurate force field for a set of MNPs presenting different sizes and morphologies can be built using a non-parametric machine learning approach. While errors on force prediction incurred by GP-FFs trained on a dataset containing MNPs of a single size on MNPs of different size are very large, much better accuracies are obtained for mixed training datasets that include all the sizes considered, especially for 3-body GP-FFs. The high errors incurred in *cross-size* test scenarios act as a confirmation that “every atom counts” in the sub-nano regime: the dynamical behaviour (i.e. forces) of an MNP are not accurately predicted by algorithms trained on MNPs containing a slightly different number of atoms. To address the size transferability of GP-FFs for NPs where the structural properties are not so strongly affected by the exact number of atoms we, therefore, turn our attention towards bigger systems, which properties are less sensitive to the addition of single atoms.

5.2 Transferable GP-FFs for Large Au NPs

To assess whether a GP-FF trained on data coming from an NP of a certain size can accurately predict forces and energies on an NP of a bigger size, we consider three Au NPs containing respectively 147, 309 and 561 atoms. The reference quantum

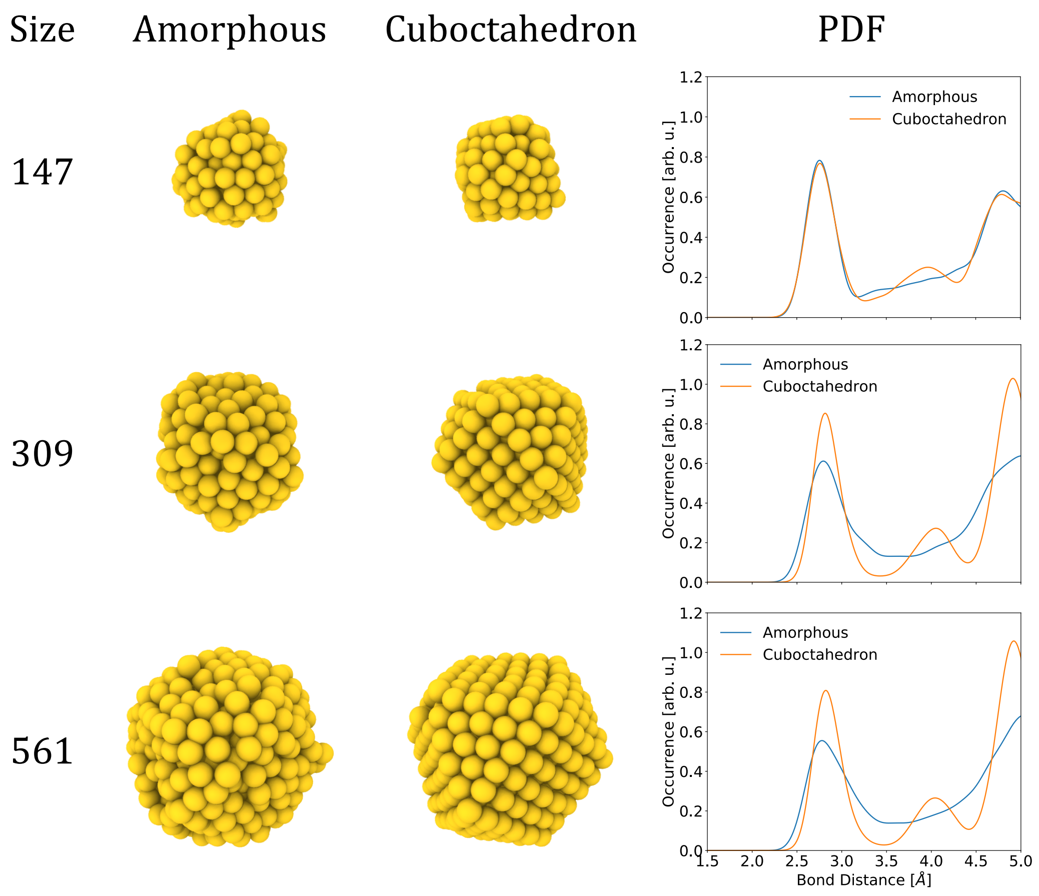


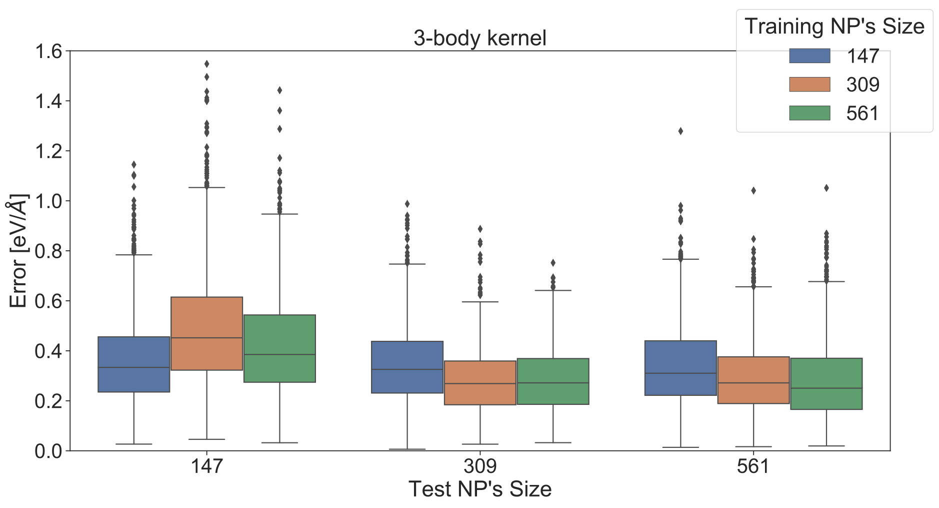
Figure 5.4: Six Au NP structures, two per NP size, representative of the morphologies encountered in the reference quantum database. Structures are vertically ordered according to their number of atoms, and horizontally according to categorical morphology: amorphous and cuboctaheral. The PDF for the depicted snapshots is also shown to the left, to highlight the structural difference between the NPs.

data, originally used in Ref. [156], is generated via AIMD simulations at temperatures ranging from 300 to 925 K using the VASP package [157], under the local density approximation [158] [2]. The AIMD simulations are carried out in an NVT ensemble, using a timestep of 2 fs, and employ a Nose-Hoover thermostat with an effective mass of 40 timesteps. The energy cutoff of the plane-wave basis set is 240 eV, with an energy convergence cutoff of 10^{-6} eV. The final dataset contains approximately 60 snapshots extracted from longer AIMD simulations for each Au MNP size considered, for a total of $\sim$ 60000 reference force data. For all three

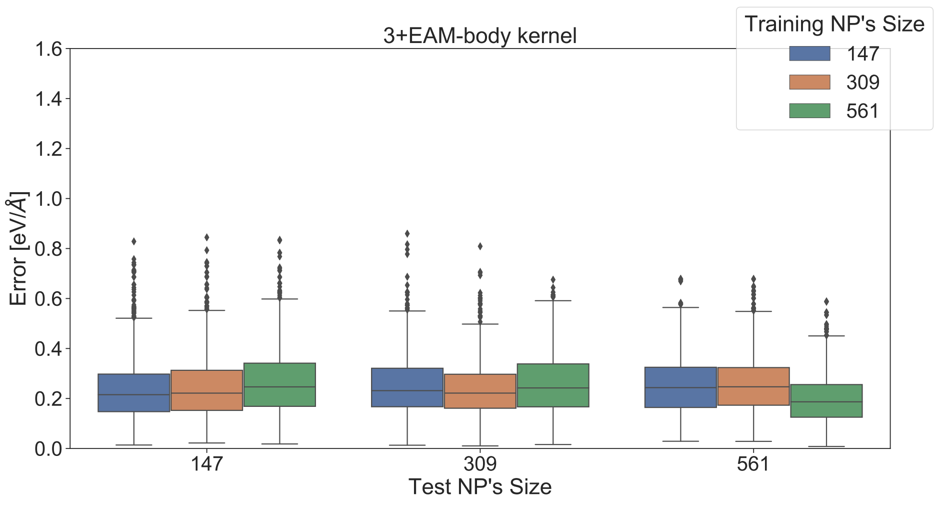
[2]We thank Dr. Theodore Pavloudis, College of Engineering, Swansea University, for providing this reference quantum dataset

NP sizes, the initial structures in the AIMD simulations are perfectly octahedral, and become increasingly more amorphous as simulation temperature increases. In Fig. 5.4 we display six snapshots of the structures present in the reference quantum datasets, two per NP size. The NPs shown in the “Cuboctahedron” column of Fig. 5.4 are the starting perfectly symmetric structures, and NPs shown in the “Amorphous” columns exemplify the low-symmetry geometries encountered in high-T AIMD simulations.

We first train 3-body GP-FFs on reference force data for each of the three NP sizes and test their accuracy both in a *same-size* and in a *cross-size* scenario. The distribution of errors on force vectors incurred by these 3-body GP-FFs, trained using 1000 $\{\mathbf{q}_3, \mathbf{f}\}$ pairs, and tested on 2000 local atomic configurations for each NP size (not included in the training dataset), are graphed in Fig. 5.5a. The MAEs incurred while *same-size* and *cross-size* testing are more homogeneous than the ones encountered in *same-size* tests for smaller Ni nanoparticles in Section 5.1, and are comprised between 0.28 and 0.48 eV/Å. Nonetheless, the 3-body GP-FFs display an overall unsatisfactory accuracy on force prediction; we, therefore, fit EAM-like GP-FFs, introduced in Sec. 2.2.3, on the difference between the 3-body GP-FFs force predictions and the reference quantum forces, to increase the GP-FFs' performance. The accuracies of the resulting GP-FFs, which do contain a 2-body, a 3-body, and a many-body term, are shown in Fig. 5.5b. A comparison of the two heat maps displayed in Fig.5.5 highlights the accuracy gain resulting from the introduction of an EAM-like correction to the 3-body GP-FF. These GP-FFs display better performance than their 3-body counterpart and incur very homogeneous errors across the nine training NP size - test NP size combinations, with an average MAE on force vectors of 0.242 $\pm$ 0.036 eV/Å. The small difference between errors encountered in *same-size* and *cross-size* tests implies that local atomic environments present in the reference data of a NP of a certain size are very similar to the ones encountered in reference data of other NP sizes for this system. Of particular interest is the good accuracy displayed by the 2+3+EAM GP-FF trained on the smallest



(a) 3-body



(b) 3-body + EAM

Figure 5.5: Box plot of errors on force vectors [eV/Å] for 3-body (a) and 3-body + EAM (b) GP-FFs trained and tested on databases containing quantum reference data relative to Au NPs with different number of atoms. The number of atoms in the NP used to train the GP-FF is color-coded, with blue being 147, orange 309 and green 561. On the x-axis we group error distributions according to the number of atoms of the NP used to test the GP-FFs. On the y-axis we report the distribution of the error on force vector incurred by the GP-FFs. Each training dataset contained ∼1000 local atomic environment - force vector pairs, and each test datasets contained ∼2000 local atomic environment - force vector pairs not included in the training dataset. The training and test datasets are kept fixed for both kernels and for all tests.

of the three NPs, Au_{147}, which incurs MAE on force vectors of 0.252 eV/Å while tested on Au_{561}, an NP more than three times larger than Au_{147}.

Chapter 6

Conclusions

The development of accurate force fields to run molecular dynamics simulations for long times and on large systems is a complex and open problem, that traditionally requires a great deal of human effort and manual tuning of parameters. In the past years, machine learning algorithms have been employed to generate non-parametric FFs starting from existing databases, commonly obtained from *ab initio* simulations. The most widespread algorithms for the creation of machine learning FFs are Gaussian process regression and artificial neural networks. The former can be trained using very restricted reference datasets, but their computational cost in prediction scales linearly with the training set size. The latter require training sets that are on average much larger than the ones used by GPR algorithms, but have the advantage that their prediction cost is independent of the number of training points used, and therefore usually lower than GPR algorithms displaying comparable accuracy. Traditionally, regardless of the specific machine learning methodology used, the resulting force fields are many-body. This means that the atomic force and energy predictions yielded by such FFs cannot be decomposed into a finite series of n-body contributions, e.g. bond distance, angle, torsion angle, but depend instead on the set of positions of all atoms contained in the local atomic environment. Many-body FFs are constructed either by employing many-body descriptors as inputs to the regression problem or by using low-order descriptors as inputs and then exploiting the nonlinearities of the learning algorithm (this is e.g. the case in artificial neural networks).

In this work, we discussed the use of GPR to build FFs that are explicitly 2- and 3-body. In Chapter 2 we introduced explicit 2- (3-) body descriptors and use them as inputs to the GPR algorithm in order to obtain Gaussian process force fields that are explicitly 2- (3-body). We then exploited the possibility to decompose such specifically designed GP-FFs into explicit n-body contributions to "map" the GP-FFs onto classical nonparametric tabulated FF. Using this approach, it is possible to circumvent the linear scaling of GPR in the training database size and to obtain a significant computational speed-up, of the order of 10^4-10^5 when compared to the traditional Gaussian process FFs. Moreover, the "mapping" technique preserves the accuracy of the original Gaussian process FF, and allows for an easy interpretation and visualization of the final FF. We also presented a many-body correction term to 2+3-body FFs, which is inspired by embedded atom method (EAM) potentials, and is a function of a scalar many-body local atomic environment descriptor. This EAM-like many-body term, unlike more complex many-body terms used in literature, can be also "mapped" onto a classical nonparametric FF, thanks to the low dimensionality of the associated descriptor. The correction introduced by this term reduces errors incurred by mapped FFs on the force prediction in metallic systems such as bulk Fe containing grain boundaries and Au NPs, without an increase in the computational cost associated with the M-FF.

In Chapter 3 we presented the MFF Python package, an open-source and documented code that encodes the algorithms discussed in this thesis work, including the mapping methodology. MFF, therefore, offers a tool for researchers to automatically construct nonparametric classical force fields starting from reference quantum data. The ability to build fast FFs from small reference databases enables the study via MD simulations of systems for which a fast, classical FF would be necessary (e.g. because of the simulation's length and/or the large number of atoms present), but is not available, or not accurate enough. We also carried out speed and accuracy tests for such package and showed that FFs trained by MFF on restricted quantum datasets containing molecular dynamics simulations of Fe grain boundaries were able to achieve better accuracy on force prediction than various iron-specific EAM potentials found in literature, while being as fast to compute. As of May 2020, following a collaboration between the author and the authors of the Fast Learning of Atomistic Rare Events (FLARE) package, the MFF Python package is no longer supported and has been merged into the FLARE Python package.

In Chapter 4, we proposed a systematic way to construct machine learning FFs for metallic nanoparticles, a traditionally challenging ensemble of systems to accurately simulate using classical parametric FFs. Chemo-physical properties of metallic NPs depend so strongly on their architecture that changes in their geometrical arrangement and chemical ordering as a function of the nanoparticles size and chemical composition, or simply due to variation of the environment's conditions, must be predicted carefully. The first system we analysed is Ni_{19}, where we considered different training datasets containing one or more isomers obtained from different AIMD at different T. We then built two mapped FFs that incurred average errors on the force vectors smaller than 0.15 eV/Å when tested on reference *ab initio* data. We used these FFs, one trained using data coming from *ab initio* simulations carried out at 300 K, the other from similar simulations carried out at 300, 600, and 900 K, to run melting MD simulations of Ni_{19}, for a total of $\sim$180 ns of simulation time for each M-FF. The speed-up obtained by "mapping" the FFs allowed us to run the simulations on 24 CPU cores in 4 days; the same process would have taken $\sim$2000 years had we employed the *ab initio* method used to generate the reference quantum data. The melting MD simulations helped us identify the presence of a "slush" state, intermediate between nanosolid and nanoliquid states, in the $\sim$700-900 K temperature range.

In Chapter 5, we discussed the issue of the transferability of machine learning force fields for metallic NPs across their sizes. We focused again on Ni nanoparticles of 13 to 20 atoms presenting various isomers. We observed that GP-FFs trained on a single NP size were not able to faithfully predict forces when tested on NPs of other sizes; this result implies that at such small particle sizes, “every atom counts” also for a ML force field. Nonetheless, by including relevant isomers at all sizes in the training set, we built a 3-body mapped FF that incurs an average error on forces of 0.15 eV/Å across the various nanoparticle sizes and geometries. We subsequently moved the first steps towards the development of mapped FFs for nanoparticles of diameter $\sim$1.5 - 2 nm. Due to the technological interest of predicting the dynamical behaviour of large NPs, it would be ideal to understand whether M-FFs trained on NPs of sizes which can be simulated at a DFT level can be accurately used for larger NPs too. We considered Au nanoparticles containing 147, 309 and 561 atoms, and tested whether GP-FFs trained on the smaller nanoparticles can then be accurately employed on bigger systems. We found that 2+3+EAM GP-FFs are able to reproduce most of the atomic interactions predicted by the quantum reference methods, incurring in an average error on forces of $\sim$0.28 eV/Å. In particular, GP-FFs trained only using data of the smallest of the available nanoparticle systems perform almost identically when tested on bigger nanoparticles to GP-FFs which were specifically trained using reference quantum calculations of the test system. These preliminary results pave the way towards the construction and usage of GP-FFs to simulate the dynamical behaviour of MNPs at larger sizes, with an accuracy reasonably close to DFT, by employing *ab initio* training datasets containing smaller MNPs.

The work presented in this thesis is by no means exhaustive and many topics of interest have not been covered here, but could be addressed by future research. The use of mapped force fields trained on both force and energy data, for example, was not fully explored in this work, although it is implemented in the MFF package. The implementation of the mappable 2+3+EAM FFs, which offers promising results, should be considered in its infancy and we forecast few possible extensions, first and foremost an improved handling of multiple chemical species in the EAM descriptor. With regards to applications of M-FFs to MD simulations, the study of large Au nanoparticles is ongoing, and the simulation of non-periodic systems containing more than 1000 atoms is a likely future direction of research. Finally, uses of the

MFF package for applications such as speeding-up the calculations of implicit water models are currently being investigated.

Appendix A

A.1 Sampling Methods in the MFF Package

The MFF package currently supports four sampling methods for the reference training dataset: random, 2-body descriptor, 3-body descriptor, and CNA sampling.

The 2-body descriptor sampling exploits the explicit dependency of the 2-body GP-FF on the inter-atomic distances to sample from the database informative local atomic environments ρ_i. To do so, an array that bins the distance values from 0 to the cutoff radius r_C is created; the number of bins is either automatically optimised to approximate the desired number of training points, or specified by the user. Then the sampling module iterates over a shuffled database and includes local atomic environments ρ_i in the training set only if at least one distance r_{ij} in ρ_i belongs to a bin of inter-atomic distances that has no elements in the training set. This way the highest number of "unique" r_{ij} values are represented in the training set, and the number of bins of the descriptor for which there is no training point are minimized. The 3-body descriptor sampling is a straightforward extension of the previous method to the case of 3-body GP-FFs. In this case, triplets of inter-atomic distances are binned in a 3-D vector, and local atomic environments containing at least a triplet of atoms in which the distance values (r_{ij}, r_{jk}, r_{ik}) are not already present in the training set are included.

Finally, CNA sampling is based upon the CNA descriptor, which was originally developed by Stukowski et. al. to characterize local atomic environments [159, 160]. The CNA descriptor works on the graph of atoms, where connections exist between nearest neighbours. Two atoms are considered nearest neighbours if their distance is less than a cutoff radius r_{CNA}, usually taken as the mean between the first and second peak of the pair distribution function in the system. The CNA descriptor for a local atomic environment is then built as an $M \times 3$ vector of integers, where each row is a CNA signature relative to the nearest neighbour j of atom i and reads:

$$\mathrm{CNA}_{ij} = (a_{ij}, b_{ij}, c_{ij}) \tag{A.1}$$

where a_{ij} is the number of nearest neighbours i and j have in common, b_{ij} the number of bonds between these common neighbours, and c_{ij} the length of the largest chain of bonds connecting common neighbours. The CNA descriptor is calculated for every atom i, and local atomic environments ρ_i are sampled from the dataset so that the distribution of CNA signatures is as uniform as possible. This way, local atomic environments that present unique atomic arrangements are prioritized in the training set.

A.2 Many-body Kernels

Throughout the manuscript, two different unique many-body kernels were used as benchmarking tools. The many-body force kernel used in Chapters 2, 3, and 4 was built starting from the following many-body energy kernel:

$$k_{MB,appr.}(\rho_i, \rho_j) = \exp\left(k_2(\rho_i, \rho_j)\right) \tag{A.2}$$

and then employing a covariant summation:

$$\mathbf{K}_{MB,appr.}(\rho_i, \rho_j) = \sum_{i=1}^{48} \mathbf{U}_i \exp\left(k_2(\rho_i, \mathbf{U}_i \rho_j)\right) \tag{A.3}$$

where $\mathbf{U}_i$ are rotation matrices belonging to the O_h crystallographic point group. This kernel, introduced in [65], is an approximation of the unique fully many-body kernel and is much faster to compute.

The many-body force kernel used in Chapter 5 instead obtained via the exponentiation of an already symmetric 3-body kernel, as per Eq. (2.40). It reads:

$$k_{MB}(\rho_i, \rho_j) = \exp\left(\frac{k_3(\rho_i, \rho_j)}{\gamma^2}\right), \qquad \text{(A.4)}$$

where $k_3(\rho_i, \rho_j)$ is the 3-body kernel of Eq. (2.38), and γ is a parameter that governs the relative importance of low-order and high-order interactions. The expression for the correspondent force and force-energy kernels was then obtained via differentiation following Eq. (2.29). The kernel in Eq. (A.4) is the many-body kernel included in the “MFF” Python package [35].

A.3 Transferability of GP-FFs in AgAu(32) NPs

We consider systems of AgAu nanoparticles comprised of 32 atoms, with varying morphology, chemical composition and chemical arrangement. Our aim is to test the accuracy of GP-FFs trained and tested on data containing heterogeneous chemical compositions and chemical arrangements. The reference quantum data was generated using the Quantum Espresso package [161], where the exchange-correlation potential was calculated employing the PBE/GGA approximation. The Rabe-Rappe-Kaxiras-Joannopoulos ultrasoft pseudo-potential has been used to model interactions between valence electrons and nuclei. The plane-wave energy cutoff was set to 544 eV, and the charge density cutoff to 4900 eV [1].

Fig. A.1 displays the seven AgAu(32) compositions contained in the reference quantum dataset; all the AIMD simulations that generated the reference data shared the same starting geometry: an elongated AgAu(32) nanoparticle presenting an HCP motif, albeit possessing different chemical compositions. We trained and tested a 3-body and a 3-body + EAM GP-FFs on the dataset; the resulting training curves for the MAE on force vectors for 1000 randomly-sampled test points are

[1]We thank Dr. Francesca Baletto, Physics Department, King’s College London, for providing this reference quantum dataset.

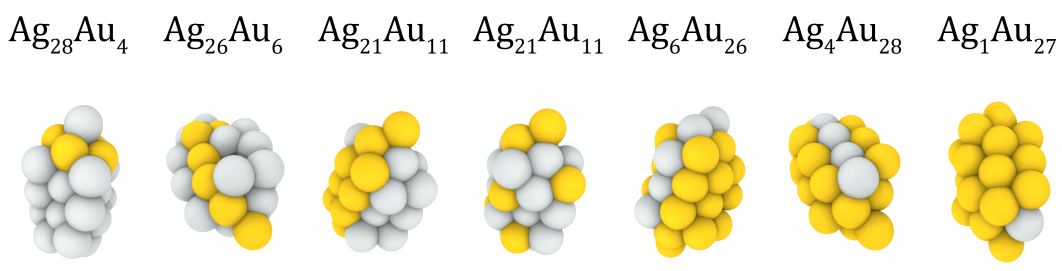


Figure A.1: Seven $AgAu_{32}$ NP structures, representative of the chemical compositions and chemical arrangements encountered in the reference quantum database. Ag atoms are represented in grey, Au in yellow.

shown in Fig. A.2, together with the relative standard deviations. The 3-body and

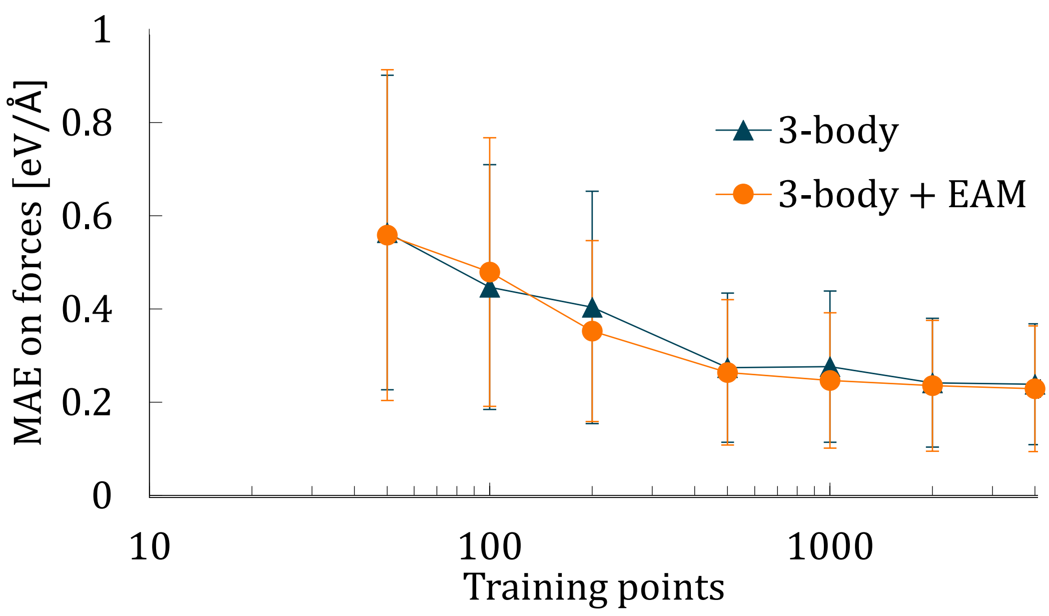


Figure A.2: MAE on reference force vectors as a function of the number of training points used for 3-body (blue triangles) and 3-body+EAM (orange dots) GP-FFs on AgAu(32) MNPs with different chemical compositions. The standard deviation of the error on force vectors is shown for a test set containing 1000 points.

the 3-body+EAM GP-FFs incur very similar errors, and both reach a plateau MAE of ∼0.23 eV/Å at 2000 training points. The negligible MAE difference between the two GP-FFs (i.e. ∼0.01 eV/Å) might be due to the way multiple atomic species are treated within the EAM kernel, which at the moment only distinguishes between different atomic species for the central atom of the local atomic descriptor, not for the surrounding ones. Improvements in the way multiple atomic species are handled by the EAM GP-FFs are under development. Nonetheless, the accuracy reached by the GP-FFs is encouraging, as it narrowly misses the target MAE of 0.15 eV/Å despite the complexity given by the presence of multiple chemical compositions and chemical arrangements in the dataset.

Bibliography

[1] M. Ferrario, G. Ciccotti, and K. Binder, *Computer simulations in condensed matter: from materials to chemical biology*, vol. 1. Springer, 2007.

[2] L. E. Murr, *Computer Simulations in Materials Science and Engineering*, pp. 1–15. Cham: Springer International Publishing, 2016.

[3] F. F. Abraham, J. Q. Broughton, N. Bernstein, and E. Kaxiras, "Spanning the length scales in dynamic simulation," *Computers in Physics*, vol. 12, no. 6, pp. 538–546, 1998.

[4] J. B. Gibson, A. N. Goland, M. Milgram, and G. H. Vineyard, "Dynamics of radiation damage," *Physical Rev.*, vol. 120, pp. 1229–1253, Nov 1960.

[5] A. Rahman, "Correlations in the motion of atoms in liquid argon," *Physical Rev.*, vol. 136, pp. A405–A411, Oct 1964.

[6] P. Hohenberg and W. Kohn, "Inhomogeneous electron gas," *Physical review*, 1964.

[7] W. Kohn and L. J. Sham, "Self-consistent equations including exchange and correlation effects," *Physical review*, vol. 140, no. 4A, pp. A1133—-A1138, 1965.

[8] R. Car and M. Parrinello, "Unified Approach for Molecular Dynamics and Density-Functional Theory," *Physical Review Letters*, vol. 55, pp. 2471–2474, Nov. 1985.

[9] L. E. Ratcliff, S. Mohr, G. Huhs, T. Deutsch, M. Masella, and L. Genovese, “Challenges in large scale quantum mechanical calculations,” *WIREs Computational Molecular Science*, vol. 7, no. 1, p. e1290, 2017.

[10] Z. Li, J. R. Kermode, and A. De Vita, “Molecular dynamics with on-the-fly machine learning of quantum-mechanical forces,” *Physical Review Letters*, vol. 114, no. 9, pp. 1–5, 2015.

[11] F. H. Stillinger and T. A. Weber, “Computer simulation of local order in condensed phases of silicon,” *Physical review*, vol. B31, no. 8, pp. 5262–5271, 1985.

[12] J. Tersoff, “New empirical approach for the structure and energy of covalent systems,” *Physical Review B*, vol. 37, pp. 6991–7000, Apr. 1988.

[13] D. W. Brenner, “The art and science of an analytic potential,” *physica status solidi(b)*, vol. 217, pp. 23–40, 2000.

[14] Y. Mishin, “Atomistic modeling of the γ and γ’-phases of the NiAl system,” *Acta Materialia*, vol. 52, pp. 1451–1467, apr 2004.

[15] A. C. T. van Duin, S. Dasgupta, F. Lorant, and W. A. Goddard, “ReaxFF: A Reactive Force Field for Hydrocarbons,” *The Journal of Physical Chemistry A*, vol. 105, pp. 9396–9409, Oct. 2001.

[16] G. A. Cisneros, K. T. Wikfeldt, L. Ojamäe, J. Lu, Y. Xu, H. Torabifard, A. P. Bartók, G. Csányi, V. Molinero, and F. Paesani, “Modeling Molecular Interactions in Water: From Pairwise to Many-Body Potential Energy Functions,” *Chemical Reviews*, vol. 116, pp. 7501–7528, July 2016.

[17] S. K. Reddy, S. C. Straight, P. Bajaj, C. Huy Pham, M. Riera, D. R. Moberg, M. A. Morales, C. Knight, A. W. Götz, and F. Paesani, “On the accuracy of the MB-pol many-body potential for water: Interaction energies, vibrational frequencies, and classical thermodynamic and dynamical properties

from clusters to liquid water and ice," *The Journal of Chemical Physics*, vol. 145, pp. 194504–14, Nov. 2016.

[18] A. J. Skinner and J. Q. Broughton, "Neural networks in computational materials science: Training algorithms," *Modelling and Simulation in Materials Science and Engineering*, vol. 3, p. 317, 1995.

[19] T. B. Blank, S. D. Brown, A. W. Calhoun, and D. J. Doren, "Neural network models of potential energy surfaces," *The Journal of Chemical Physics*, vol. 103, no. 10, pp. 4129–4137, 1995.

[20] D. F. R. Brown, M. N. Gibbs, and D. C. Clary, "Combining ab initio computations, neural networks, and diffusion monte carlo: An efficient method to treat weakly bound molecules," *The Journal of Chemical Physics*, vol. 105, no. 17, pp. 7597–7604, 1996.

[21] J. Behler and M. Parrinello, "Generalized Neural-Network Representation of High-Dimensional Potential-Energy Surfaces," *Physical Review Letters*, vol. 98, pp. 146401–146404, apr 2007.

[22] E. V. Podryabinkin and A. V. Shapeev, "Active learning of linearly parametrized interatomic potentials," *Computational Materials Science*, vol. 140, pp. 171–180, 2017.

[23] A. P. Bartók, M. C. Payne, R. Kondor, and G. Csányi, "Gaussian approximation potentials: The accuracy of quantum mechanics, without the electrons," *Physical Review Letters*, vol. 104, p. 136403, Apr 2010.

[24] A. Glielmo, P. Sollich, and A. De Vita, "Accurate interatomic force fields via machine learning with covariant kernels," *Physical Review B*, vol. 95, no. 21, pp. 1–13, 2017.

[25] V. Botu and R. Ramprasad, "Learning scheme to predict atomic forces and accelerate materials simulations," *Physical Review B*, vol. 92, pp. 94305–94306, sep 2015.

[26] G. Ferré, T. Haut, and K. Barros, “Learning potential energy landscapes using graph kernels,” *situations*, vol. 9, p. 10, 2017.

[27] A. Takahashi, A. Seko, and I. Tanaka, “Linearized machine-learning interatomic potentials for non-magnetic elemental metals: Limitation of pairwise descriptors and trend of predictive power,” *Journal of Chemical Physics*, vol. 148, no. 23, 2018.

[28] F. A. Faber, L. Hutchison, B. Huang, J. Gilmer, S. S. Schoenholz, G. E. Dahl, O. Vinyals, S. Kearnes, P. F. Riley, and O. A. Von Lilienfeld, “Prediction Errors of Molecular Machine Learning Models Lower than Hybrid DFT Error,” *Journal of Chemical Theory and Computation*, vol. 13, no. 11, pp. 5255–5264, 2017.

[29] K. T. Schütt, F. Arbabzadah, S. Chmiela, K. R. Müller, and A. Tkatchenko, “Quantum-chemical insights from deep tensor neural networks,” *Nature communications*, vol. 8, p. 13890, 2017.

[30] K. T. Schütt, H. E. Sauceda, P.-J. Kindermans, A. Tkatchenko, and K.-R. Müller, “SchNet A deep learning architecture for molecules and materials,” *Journal of Chemical Physics*, vol. 148, no. December 2017, 2018.

[31] R. Ramprasad, R. Batra, G. Pilania, A. Mannodi-Kanakkithodi, and C. Kim, “Machine learning in materials informatics: recent applications and prospects,” *npj Computational Materials*, vol. 3, no. 1, p. 54, 2017.

[32] M. J. Willatt, F. Musil, and M. Ceriotti, “Atom-density representations for machine learning,” *Journal of Chemical Physics*, vol. 154110, no. April, 2019.

[33] J. S. Smith, B. T. Nebgen, R. Zubatyuk, N. Lubbers, C. Devereux, K. Barros, S. Tretiak, O. Isayev, and A. E. Roitberg, “Approaching coupled cluster accuracy with a general-purpose neural network potential through transfer learning,” *Nature communications*, vol. 10, no. 1, pp. 1–8, 2019.

[34] J. R. Boes, M. C. Groenenboom, J. A. Keith, and J. R. Kitchin, "Neural network and ReaxFF comparison for Au properties," *International Journal of Quantum Chemistry*, vol. 116, pp. 979–987, Mar. 2016.

[35] C. Zeni, F. Ádám, and A. Glielmo, "MFF a Python package for building nonparametric force fields from machine learning." `https://github.com/kcl-tscm/mff`, 2018.

[36] F. Baletto and R. Ferrando, "Structural properties of nanoclusters: Energetic, thermodynamic, and kinetic effects," *Review Modern Physics*, vol. 77, pp. 371–423, May 2005.

[37] D. M. Foster, R. Ferrando, and R. E. Palmer, "Experimental determination of the energy difference between competing isomers of deposited, size-selected gold nanoclusters," *Nature Communications*, vol. 9, p. 1323, dec 2018.

[38] Z. Wang and R. E. Palmer, "Experimental Evidence for Fluctuating, Chiral-Type Au_{55} Clusters by Direct Atomic Imaging," *Nano Letters*, vol. 12, pp. 5510–5514, nov 2012.

[39] R. Ferrando, J. Jellinek, and R. L. Johnston, "Nanoalloys : From Theory to Applications of Alloy Clusters and Nanoparticles," *Chemical Reviews*, vol. 108, no. 3, pp. 846–910, 2008.

[40] J. M. Rahm and P. Erhart, "Beyond Magic Numbers: Atomic Scale Equilibrium Nanoparticle Shapes for Any Size," *Nano Letters*, vol. 17, pp. 5775–5781, 2017.

[41] Z. Li and H. A. Scheraga, "Monte Carlo-minimization approach to the multiple-minima problem in protein folding," *Proceedings of the National Academy of Sciences*, vol. 84, no. October, pp. 6611–6615, 1987.

[42] J. P. K. Doye and D. J. Wales, "Global minima for transition metal clusters described by suttonchen potentials," *New Journal of Chemistry*, vol. 22, pp. 733–744, 1998.

[43] G. Rossi and R. Ferrando, "Searching for low-energy structures of nanoparticles: a comparison of different methods and algorithms," *Journal of Physics: Condensed Matter*, vol. 21, p. 084208, jan 2009.

[44] G. Barcaro, A. Fortunelli, G. Rossi, F. Nita, and R. Ferrando, "Electronic and Structural Shell Closure in AgCu and AuCu Nanoclusters," *Journal of Physical Chemistry B*, vol. 110, pp. 23197–23203, 2006.

[45] F. Calvo, D. Schebarchov, and D. J. Wales, "Grand and Semigrand Canonical Basin-Hopping," *Journal of Chemical Theory and Computation*, vol. 12, pp. 902–909, 2016.

[46] C. J. Pickard, "Hyperspatial optimization of structures," *Physical Review B*, vol. 054102, pp. 1–10, 2019.

[47] R. L. Johnston, "Evolving better nanoparticles: Genetic algorithms for optimising cluster geometries," *Dalton Transactions*, pp. 4193–4207, 2003.

[48] S. Heiles and R. L. Johnston, "Global Optimization of Clusters Using Electronic Structure Methods," *International Journal of Quantum Chemistry*, vol. 113, pp. 2091–2109, 2013.

[49] T. Lazauskas, A. A. Sokol, and S. M. Woodley, "An efficient genetic algorithm for structure prediction at the nanoscale," *Nanoscale*, vol. 9, pp. 3850–3864, 2017.

[50] J. Lv, Y. Wang, L. Zhu, and Y. Ma, "Particle-swarm structure prediction on clusters," *The Journal of Chemical Physics*, vol. 137, no. 8, p. 084104, 2012.

[51] S. V. Lepeshkin, V. S. Baturin, Y. A. Uspenskii, and A. R. Oganov, "Stability of Nanoclusters in a Wide Area of Compositions," *The Journal of Physical Chemistry Letters*, vol. 10, pp. 102–106, 2018.

[52] S. A. Trygubenko and D. J. Wales, "Kinetic analysis of discrete path sampling stationary point databases Kinetic analysis of discrete path sam-

pling stationary point databases," *Molecular Physics ISSN:*, vol. 104, no. 9, pp. 1497–1507, 2006.

[53] E. F. Koslover and J. D. Wales, "Comparison of double-ended transition state search methods Comparison of double-ended transition state search methods," *Journal of Chemical Physics*, vol. 12, no. October 2007, 2017.

[54] A. Laio and M. Parrinello, "Escaping free-energy minima," *Proceedings of the National Academy of Sciences*, vol. 99, no. 20, pp. 12562–12566, 2002.

[55] A. Laio and F. L. Gervasio, "Metadynamics : a method to simulate rare events and reconstruct the free energy in biophysics , chemistry and material science," *Reports on Progress in Physics*, vol. 71, 2008.

[56] L. Pavan, K. Rossi, and F. Baletto, "Metallic nanoparticles meet metadynamics," *The Journal of Chemical Physics*, vol. 143, p. 184304, nov 2015.

[57] A. L. Gould, K. Rossi, C. R. A. Catlow, F. Baletto, and A. J. Logsdail, "Controlling Structural Transitions in AuAg Nanoparticles through Precise Compositional Design," *The Journal of Physical Chemistry Letters*, vol. 7, pp. 4414–4419, nov 2016.

[58] K. Rossi and F. Baletto, "The effect of chemical ordering and lattice mismatch on structural transitions in phase segregating nanoalloys," *Physical Chemistry Chemical Physics*, vol. 19, pp. 11057–11063, may 2017.

[59] J. Skilling, "Nested sampling for general Bayesian computation," *Bayesian Analysis*, vol. 1, pp. 833–859, dec 2006.

[60] L. B. Partay, A. P. Bartok, and G. Csanyi, "Efficient Sampling of Atomic Configurational Spaces," *The Journal of Physical Chemistry B*, vol. 114, pp. 10502–10512, aug 2010.

[61] K. Rossi, L. B. Pártay, G. Csányi, and F. Baletto, "Thermodynamics of CuPt nanoalloys," *Scientific Reports*, vol. 8, p. 9150, dec 2018.

[62] J. Dorrell and L. B. Pártay, "Thermodynamics and the potential energy landscape: case study of small water clusters," *Physical Chemistry Chemical Physics*, vol. 21, pp. 7305–7312, apr 2019.

[63] J. P. K. Doye and D. J. Wales, "On potential energy surfaces and relaxation to the global minimum," *The Journal of Chemical Physics*, vol. 8428, no. June 1998, 1996.

[64] D. Guedes-Sobrinho, W. Wang, I. P. Hamilton, J. L. Da Silva, and L. M. Ghiringhelli, "(meta-) stability and core–shell dynamics of gold nanoclusters at finite temperature," *The journal of physical chemistry letters*, vol. 10, no. 3, pp. 685–692, 2019.

[65] A. Glielmo, C. Zeni, and A. De Vita, "Efficient nonparametric *n*-body force fields from machine learning," *Physical Review B*, vol. 97, no. 18, pp. 1–12, 2018.

[66] C. Zeni, K. Rossi, A. Glielmo, and F. Baletto, "On machine learning force fields for metallic nanoparticles," *Advances in Physics: X*, vol. 4, no. 1, p. 1654919, 2019.

[67] M. Born and R. Oppenheimer, "Zur quantentheorie der molekeln," *Annalen der physik*, vol. 389, no. 20, pp. 457–484, 1927.

[68] B. G. Sumpter and D. W. Noid, "Potential energy surfaces for macromolecules. a neural network technique," *Chemical Physics Letters*, vol. 192, no. 5, pp. 455 – 462, 1992.

[69] H. Gassner, M. Probst, A. Lauenstein, and K. Hermansson, "Representation of intermolecular potential functions by neural networks," *The Journal of Physical Chemistry A*, vol. 102, pp. 4596–4605, Jun 1998.

[70] W. Kohn, "Density functional and density matrix method scaling linearly with the number of atoms," *Physical Review Letters*, vol. 76, pp. 3168–3171, Apr 1996.

[71] R. Ramakrishnan, P. O. Dral, M. Rupp, and O. A. von Lilienfeld, “Big data meets quantum chemistry approximations: The δ-machine learning approach,” *Journal of Chemical Theory and Computation*, vol. 11, no. 5, pp. 2087–2096, 2015. PMID: 26574412.

[72] M. Welborn, L. Cheng, and T. F. Miller, “Transferability in machine learning for electronic structure via the molecular orbital basis,” *Journal of Chemical Theory and Computation*, vol. 14, no. 9, pp. 4772–4779, 2018. PMID: 30040892.

[73] N. Artrith, T. Morawietz, and J. Behler, “High-dimensional neural-network potentials for multicomponent systems: Applications to zinc oxide,” *Physical Review B*, vol. 83, p. 153101, Apr 2011.

[74] T. Bereau, D. Andrienko, and O. A. von Lilienfeld, “Transferable atomic multipole machine learning models for small organic molecules,” *Journal of Chemical Theory and Computation*, vol. 11, no. 7, pp. 3225–3233, 2015. PMID: 26575759.

[75] T. Bereau, R. A. DiStasio, A. Tkatchenko, and O. A. von Lilienfeld, “Non-covalent interactions across organic and biological subsets of chemical space: Physics-based potentials parametrized from machine learning,” *The Journal of Chemical Physics*, vol. 148, no. 24, p. 241706, 2018.

[76] K. Yao, J. E. Herr, D. W. Toth, R. Mckintyre, and J. Parkhill, “The tensormol-0.1 model chemistry: a neural network augmented with long-range physics,” *Chemical Science*, vol. 9, pp. 2261–2269, 2018.

[77] P. Bleiziffer, K. Schaller, and S. Riniker, “Machine learning of partial charges derived from high-quality quantum-mechanical calculations,” *Journal of Chemical Information and Modeling*, vol. 58, no. 3, pp. 579–590, 2018. PMID: 29461814.

[78] B. Nebgen, N. Lubbers, J. S. Smith, A. E. Sifain, A. Lokhov, O. Isayev, A. E. Roitberg, K. Barros, and S. Tretiak, “Transferable dynamic molecular charge

assignment using deep neural networks," *Journal of Chemical Theory and Computation*, vol. 14, no. 9, pp. 4687–4698, 2018. PMID: 30064217.

[79] A. Grisafi and M. Ceriotti, "Incorporating long-range physics in atomic-scale machine learning," *The Journal of Chemical Physics*, vol. 151, no. 20, p. 204105, 2019.

[80] V. L. Deringer, C. J. Pickard, and G. Csányi, "Data-Driven Learning of Total and Local Energies in Elemental Boron," *Physical Review Letters*, vol. 120, no. 15, pp. 1–5, 2018.

[81] C. E. Rasmussen and C. K. I. Williams, *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[82] I. Macêdo and R. Castro, *Learning divergence-free and curl-free vector fields with matrix-valued kernels*. Instituto Nacional de Matematica Pura e Aplicada, 2008.

[83] V. Botu and R. Ramprasad, "Adaptive machine learning framework to accelerate ab initio molecular dynamics," *International Journal of Quantum Chemistry*, vol. 115, no. 16, pp. 1074–1083, 2015.

[84] C. Zeni, K. Rossi, A. Glielmo, Á. Fekete, N. Gaston, F. Baletto, and A. De Vita, "Building machine learning force fields for nanoclusters," *Journal of Chemical Physics*, vol. 148, no. 24, 2018.

[85] A. P. Bartók, R. Kondor, and G. Csányi, "On representing chemical environments," *Physical Review B*, vol. 87, no. 18, pp. 1–16, 2013.

[86] A. P. Bartók and G. Csányi, "Gaussian approximation potentials: A brief tutorial introduction," *International Journal of Quantum Chemistry*, vol. 115, no. 16, pp. 1051–1057, 2015.

[87] M. Rupp, A. Tkatchenko, K.-R. Müller, and O. A. von Lilienfeld, "Fast and accurate modeling of molecular atomization energies with machine learning," *Physical Review Letters*, vol. 108, p. 058301, Jan 2012.

[88] H. Huo and M. Rupp, “Unified representation of molecules and crystals for machine learning,” *arXiv:1704.06439*, 2017.

[89] A. V. Shapeev, “Moment tensor potentials: A class of systematically improvable interatomic potentials,” *Multiscale Modeling & Simulation*, vol. 14, no. 3, pp. 1153–1173, 2016.

[90] T. Xie and J. C. Grossman, “Crystal graph convolutional neural networks for an accurate and interpretable prediction of material properties,” *Physical Review Letters*, vol. 120, p. 145301, Apr 2018.

[91] O. Isayev, C. Oses, C. Toher, E. Gossett, S. Curtarolo, and A. Tropsha, “Universal fragment descriptors for predicting properties of inorganic crystals,” *Nature Communications*, vol. 8, pp. 15679 EP –, Jun 2017.

[92] C. Sutton, L. M. Ghiringhelli, T. Yamamoto, Y. Lysogorskiy, L. Blumenthal, T. Hammerschmidt, J. R. Golebiowski, X. Liu, A. Ziletti, and M. Scheffler, “Crowd-sourcing materials-science challenges with the nomad 2018 kaggle competition.,” *npj Comput Mater*, vol. 5, no. 111, 2019.

[93] A. Glielmo, C. Zeni, Á. Fekete, and A. De Vita, “Building nonparametric n-body force fields using gaussian process regression,” *arXiv:1905.07626*, 2019.

[94] J. E. Jones and S. Chapman, “On the determination of molecular fields. &#x2014;ii. from the equation of state of a gas,” *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character*, vol. 106, no. 738, pp. 463–477, 1924.

[95] V. L. Deringer and G. Csányi, “Machine learning based interatomic potential for amorphous carbon,” *Physical Review B*, vol. 95, p. 094203, Mar 2017.

[96] J. Vandermause, S. B. Torrisi, S. Batzner, A. M. Kolpak, and B. Kozinsky, “On-the-Fly Bayesian Active Learning of Interpretable Force-Fields for Atomistic Rare Events,” *arXiv:1904.02042v1*, 2019.

[97] N. Bernstein, G. Csányi, and V. L. Deringer, “De novo exploration and self-guided learning of potential-energy surfaces,” *arXiv:1905.10407*, 2019.

[98] F. Cleri and V. Rosato, “Tight-binding potentials for transition metals and alloys,” *Physical Review B*, vol. 48, pp. 22–33, Jul 1993.

[99] R. Jinnouchi and R. Asahi, “Predicting Catalytic Activity of Nanoparticles by a DFT-Aided Machine-Learning Algorithm,” *Journal of Physical Chemistry Letters*, vol. 8, no. 17, pp. 4279–4283, 2017.

[100] R. Jinnouchi, H. Hirata, and R. Asahi, “Extrapolating energetics on clusters and single-crystal surfaces to nanoparticles by machine-learning scheme,” *The Journal of Physical Chemistry C*, vol. 121, pp. 26397–26405, Nov 2017.

[101] K. Hornik, “Approximation capabilities of multilayer feedforward networks,” *Neural Networks*, vol. 4, no. 2, pp. 251 – 257, 1991.

[102] B. C. Csáji, “Approximation with Artificial Neural Networks,” *MSc Thesis, Eötvös Loránd University (ELTE), Budapest, Hungary*, 2001.

[103] M. R. G. Marques, J. Wolff, C. Steigemann, and M. A. L. Marques, “Neural network force fields for simple metals and semiconductors: construction and application to the calculation of phonons and melting temperatures,” *Phys. Chem. Chem. Phys.*, vol. 21, pp. 6506–6516, 2019.

[104] N. Artrith and A. Urban, “An implementation of artificial neural-network potentials for atomistic materials simulations: Performance for TiO2,” *Computational Materials Science*, vol. 114, pp. 135–150, 2016.

[105] D. Yoo, K. Lee, W. Jeong, D. Lee, S. Watanabe, and S. Han, “Atomic energy mapping of neural network potential,” *Physical Review Materials*, vol. 3, no. 9, p. 093802, 2019.

[106] S. Chiriki, S. Jindal, and S. S. Bulusu, “Neural network potentials for dynamics and thermodynamics of gold nanoparticles,” *Journal of Chemical Physics*, vol. 146, no. 8, 2017.

[107] S. Jindal, S. Chiriki, and S. S. Bulusu, "Spherical harmonics based descriptor for neural network potentials: Structure and dynamics of Au147nanocluster," *Journal of Chemical Physics*, vol. 146, no. 20, 2017.

[108] S. Jindal and S. S. Bulusu, "A transferable artificial neural network model for atomic forces in nanoparticles," *The Journal of Chemical Physics*, vol. 149, p. 194101, nov 2018.

[109] S. Chiriki and S. S. Bulusu, "Modeling of DFT quality neural network potential for sodium clusters : Application to melting of sodium clusters (Na 20 to Na 40)," *Chemical Physics Letters*, vol. 652, pp. 130–135, 2016.

[110] N. Artrith and A. M. Kolpak, "Understanding the composition and activity of electrocatalytic nanoalloys in aqueous solvents: A combination of DFT and accurate neural network potentials," *Nano Letters*, vol. 14, no. 5, pp. 2670–2676, 2014.

[111] E. L. Kolsbjerg, A. A. Peterson, and B. Hammer, "Neural-network-enhanced evolutionary algorithm applied to supported metal nanoparticles," *Physical Review B*, vol. 94, no. 195424, pp. 1–9, 2018.

[112] S. Hajinazar, E. D. Sandoval, A. J. Cullo, and A. N. Kolmogorov, "Multitribe evolutionary search for stable Cu Pd Ag nanoparticles using neural network models ," *Physical Chemistry Chemical Physics*, vol. 21, pp. 8729–8742, 2019.

[113] N. Artrith, B. Hiller, and J. Behler, "Neural network potentials for metals and oxides First applications to copper clusters at zinc oxide," *Physica Status Solidi(b)*, vol. 1203, no. 6, pp. 1191–1203, 2013.

[114] N. Artrith and A. M. Kolpak, "Grand canonical molecular dynamics simulations of CuAu nanoalloys in thermal equilibrium using reactive ANN potentials," *Computational Materials Science*, vol. 110, pp. 20–28, 2015.

[115] J. Kang, S. H. Noh, J. Hwang, H. Chun, H. Kim, and B. Han, "First-principles database driven computational neural network approach to the discovery of active ternary nanocatalysts for oxygen reduction reaction," *Physical Chemistry Chemical Physics*, vol. 20, pp. 24539–24544, oct 2018.

[116] A. Kamath, R. A. Vargas-Hernndez, R. V. Krems, T. Carrington, and S. Manzhos, "Neural networks vs gaussian process regression for representing potential energy surfaces: A comparative study of fit quality and vibrational spectrum accuracy," *The Journal of Chemical Physics*, vol. 148, no. 24, p. 241702, 2018.

[117] G. C. Sosso, V. L. Deringer, S. R. Elliott, and G. Csányi, "Understanding the thermal properties of amorphous solids using machine-learning-based interatomic potentials," *Molecular Simulation*, vol. 44, no. 11, pp. 866–880, 2018.

[118] P. C. Jennings, S. Lysgaard, J. S. Hummelshøj, T. Vegge, and T. Bligaard, "Genetic algorithms for computational materials discovery accelerated by machine learning," *npj Computational Materials*, vol. 5, no. 1, p. 46, 2019.

[119] C. M. Bishop, *Pattern Recognition and Machine Learning*. Information Science and Statistics, New York, NY: Springer, 2006.

[120] F. Ercolessi and J. B. Adams, "Interatomic potentials from first-principles calculations: The force-matching method," *Europhysics Letters (EPL)*, vol. 26, pp. 583–588, jun 1994.

[121] 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. Schtt, 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," *Journal of Physics: Condensed Matter*, vol. 29, p. 273002, jun 2017.

[122] T. Oliphant, *Guide to NumPy*. 01 2006.

[123] E. Jones, T. Oliphant, P. Peterson, *et al.*, "SciPy: Open source scientific tools for Python," 2001–.

[124] Theano Development Team, "Theano: A Python framework for fast computation of mathematical expressions," *arXiv e-prints*, vol. abs/1605.02688, May 2016.

[125] S. Chiesa, P. Derlet, S. Dudarev, and H. Van Swygenhoven, "Optimization of the magnetic potential for α-fe," *Journal of physics. Condensed matter : an Institute of Physics journal*, vol. 23, p. 206001, 05 2011.

[126] G. J. Ackland, D. J. Bacon, A. F. Calder, and T. Harry, "Computer simulation of point defect properties in dilute fecu alloy using a many-body interatomic potential," *Philosophical Magazine A*, vol. 75, no. 3, pp. 713–732, 1997.

[127] H. Chamati, N. I. Papanicolaou, Y. Mishin, and D. Papaconstantopoulos, "Embedded-atom potential for fe and its application to self-diffusion on fe(100)," *Surface Science*, vol. 600, pp. 1793–1803, 05 2006.

[128] M. I. Mendelev, S. Han, D. J. Srolovitz, G. J. Ackland, D. Y. Sun, and M. Asta, "Development of new interatomic potentials appropriate for crystalline and liquid iron," *Philosophical Magazine*, vol. 83, no. 35, pp. 3977–3994, 2003.

[129] P. A. Olsson, "Semi-empirical atomistic study of point defect properties in bcc transition metals," *Computational Materials Science*, vol. 47, no. 1, pp. 135 – 145, 2009.

[130] X. Zhou, R. Johnson, and H. Wadley, "Misfit-energy-increasing dislocations in vapor-deposited cofe/nife multilayers," *Physical Review B*, vol. 69, 04 2004.

[131] H. Lambert, Á. Fekete, J. R. Kermode, and A. D. Vita, “Imeall: A computational framework for the calculation of the atomistic properties of grain boundaries,” *Computer Physics Communications*, vol. 232, pp. 256–263, 2017.

[132] R. Ouyang, Y. Xie, and D.-e. Jiang, “Global minimization of gold clusters by combining neural network potentials and the basin-hopping method,” *Nanoscale*, vol. 7, pp. 14817–14821, 2015.

[133] G. Sun and P. Sautet, “Metastable Structures in Cluster Catalysis from First-Principles : Structural Ensemble in Reaction Conditions and Metastability Triggered Reactivity,” *Journal of the American Chemical Society*, vol. 140, pp. 2812–2820, 2018.

[134] H. Zhai and A. N. Alexandrova, “Ensemble-Average Representation of Pt Clusters in Conditions of Catalysis Accessed through GPU Accelerated Deep Neural Network Fitting Global Optimization,” *Journal of Chemical Theory and Computation*, vol. 12, pp. 6213–6226, 2016.

[135] Z. W. Ulissi, M. T. Tang, J. Xiao, X. Liu, D. A. Torelli, M. Karamad, K. Cummins, C. Hahn, N. S. Lewis, T. F. Jaramillo, K. Chan, and J. K. Nørskov, “Machine-learning methods enable exhaustive searches for active Bimetallic facets and reveal active site motifs for CO2 reduction,” *ACS Catalysis*, vol. 7, no. 10, pp. 6600–6608, 2017.

[136] N. Artrith and J. Behler, “High-dimensional neural network potentials for metal surfaces: A prototype study for copper,” *Physical Review B*, vol. 85, p. 045439, Jan 2012.

[137] S. Chiriki, S. Jindal, and S. S. Bulusu, “c-t phase diagram and landau free energies of (agau)55 nanoalloy via neural-network molecular dynamic simulations,” *The Journal of Chemical Physics*, vol. 147, no. 15, p. 154303, 2017.

[138] E. Aprà, F. Baletto, R. Ferrando, and A. Fortunelli, "Amorphization mechanism of icosahedral metal nanoclusters," *Physical Review Letters*, vol. 93, p. 65502, aug 2004.

[139] G. Santarossa, A. Vargas, M. Iannuzzi, and A. Baiker, "Free energy surface of two- and three-dimensional transitions of Au 12 nanoclusters obtained by ab initio metadynamics," *Physical Review B - Condensed Matter and Materials Physics*, vol. 81, no. 17, 2010.

[140] J. P. Perdew, K. Burke, and M. Ernzerhof, "Generalized Gradient Approximation Made Simple," *Physical Review Letters*, vol. 77, pp. 3865–3868, oct 1996.

[141] G. P. Pun, V. Yamakov, and Y. Mishin, "Interatomic potential for the ternary ni–al–co system and application to atomistic modeling of the b2–l10 martensitic transformation," *Modelling and Simulation in Materials Science and Engineering*, vol. 23, no. 6, p. 065006, 2015.

[142] G. A. Tribello, M. Ceriotti, and M. Parrinello, "A self-learning algorithm for biased molecular dynamics," *Proceedings of the National Academy of Sciences*, vol. 107, no. 41, pp. 17509–17514, 2010.

[143] A. R. Oganov and M. Valle, "How to quantify energy landscapes of solids," *The Journal of Chemical Physics*, vol. 130, no. 10, p. 104504, 2009.

[144] K. Steenbergen and N. Gaston, "Two worlds collide: Image analysis methods for quantifying structural variation in cluster molecular dynamics," *The Journal of chemical physics*, vol. 140, no. 6, p. 064102, 2014.

[145] H. L. Zhen and D. G. Truhlar, "Nanosolids, slushes, and nanoliquids: Characterization of nanophases in metal clusters and nanoparticles," *Journal of the American Chemical Society*, vol. 130, no. 38, pp. 12698–12711, 2008.

[146] R. S. Berry and B. M. Smirnov, "Entropy and phase coexistence in clusters: Metals vs. nonmetals," *Entropy*, vol. 12, p. 13031324, May 2010.

[147] K. Steenbergen, D. Schebarchov, and N. Gaston, “Electronic effects on the melting of small gallium clusters,” *The Journal of chemical physics*, vol. 137, no. 14, p. 144307, 2012.

[148] U. Ojha, K. G. Steenbergen, and N. Gaston, “How a single aluminum atom makes a difference to gallium: First-principles simulations of bimetallic cluster melting,” *The Journal of chemical physics*, vol. 139, no. 9, p. 094309, 2013.

[149] K. G. Steenbergen and N. Gaston, “Quantum size effects in the size–temperature phase diagram of gallium: Structural characterization of shape-shifting clusters,” *Chemistry–A European Journal*, vol. 21, no. 7, pp. 2862–2869, 2015.

[150] U. Ojha, K. G. Steenbergen, and N. Gaston, “Al 20+ does melt, albeit above the bulk melting temperature of aluminium,” *Physical Chemistry Chemical Physics*, vol. 17, no. 5, pp. 3741–3748, 2015.

[151] K. G. Steenbergen and N. Gaston, “A two-dimensional liquid structure explains the elevated melting temperatures of gallium nanoclusters,” *Nano letters*, vol. 16, no. 1, pp. 21–26, 2015.

[152] F. Baletto, “Structural properties of sub-nanometer metallic clusters,” *Journal of Physics: Condensed Matter*, vol. 31, p. 113001, jan 2019.

[153] D. Foster, T. Pavloudis, J. Kioseoglou, and R. Palmer, “Atomic-resolution imaging of surface and core melting in individual size-selected au nanoclusters on carbon,” *Nature communications*, vol. 10, no. 1, p. 2583, 2019.

[154] G. Kresse and J. Furthmüller, “Vienna ab-initio simulation package (VASP),” *Vienna: Vienna University*, 2001.

[155] J. P. Perdew and A. Zunger, “Self-interaction correction to density-functional approximations for many-electron systems,” *Physical Review B*, vol. 23, no. 10, p. 5048, 1981.

[156] A. Stukowski, “Structure identification methods for atomistic simulations of crystalline materials,” *Modelling and Simulation in Materials Science and Engineering*, vol. 20, p. 045021, may 2012.

[157] A. Stukowski, V. V. Bulatov, and A. Arsenlis, “Automated identification and indexing of dislocations in crystal interfaces,” *Modelling and Simulation in Materials Science and Engineering*, vol. 20, p. 085007, oct 2012.

[158] P. Giannozzi, S. Baroni, N. Bonini, M. Calandra, R. Car, C. Cavazzoni, D. Ceresoli, G. L. Chiarotti, M. Cococcioni, I. Dabo, A. D. Corso, S. de Gironcoli, S. Fabris, G. Fratesi, R. Gebauer, U. Gerstmann, C. Gougoussis, A. Kokalj, M. Lazzeri, L. Martin-Samos, N. Marzari, F. Mauri, R. Mazzarello, S. Paolini, A. Pasquarello, L. Paulatto, C. Sbraccia, S. Scandolo, G. Sclauzero, A. P. Seitsonen, A. Smogunov, P. Umari, and R. M. Wentzcovitch, “QUANTUM ESPRESSO: a modular and open-source software project for quantum simulations of materials,” *Journal of Physics: Condensed Matter*, vol. 21, p. 395502, sep 2009.