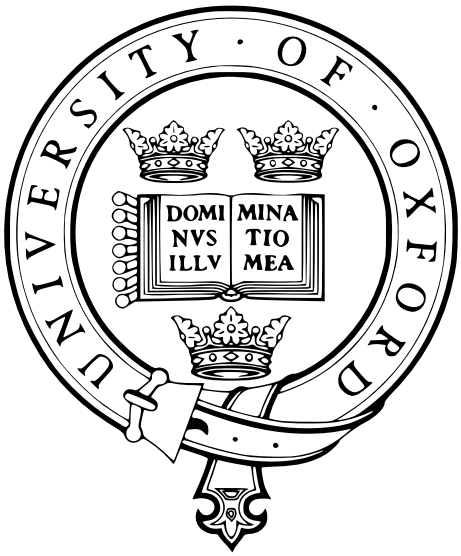


Advanced Neural Network Methods *for* Atomistic Materials Chemistry

John L. A. Gardner

This thesis provides a written account of (some of) the research that I, John L. A. Gardner*, have undertaken while a DPhil candidate under the supervision of* Prof. Volker L. Deringer *in the* Inorganic Chemistry Laboratory *at the* University of Oxford *between* 2021 *and* 2026.

~

It is dedicated to my father,[1] mother,[2] *and* sister,[3] *each of whom has been an inspiration and tremendous source of support throughout all of my endeavours.*

Abstract

Machine learned force fields (MLFFs) are functions trained to map the positions of atoms within a material or molecule to the potential energy and atomic forces generated by expensive, quantum-mechanical calculations. Their high accuracy and low cost are enabling faithful simulations of matter at the atomic scale over unprecedented time- and length-scales, revolutionising the field of computational atomistic modelling. Despite their great success, however, key problems inhibit the potential and widespread adoption of MLFFs. In this thesis, I present new methodologies to address several of these in turn.

To address the high upfront costs of labelling MLFF training data, I present work that uses cheap and approximate methods to generate and label large datasets of atomic structures. I demonstrate that "pre-training" MLFFs on this "synthetic data" can simultaneously enhance model accuracy, reduce the overall cost of training data generation, and improve the robustness of models' predictions during dynamic simulations.

To address the sub-optimal inference speeds of large MLFF models, and in particular of recently developed atomistic foundation models, I present a new approach to MLFF "distillation". I develop an automated pipeline to explore chemical space using an expensive "teacher" model, before training a cheaper "student" model to mimic the teacher's predictions on this dataset.

To address the lack of interpretability of modern MLFFs, which are typically black-box neural network models, I present work that learns symbolic force field models directly from data, and in an end-to-end manner. These simple models recover the speed and interpretability of existing empirical models while (greatly) improving upon their accuracy.

In all of the above, I use gradient descent to train neural-network-based MLFF models on large datasets of labelled atomic structures. I therefore begin by exploring and motivating this approach, starting from a simple toy system, and building up to a real world use case. As part of this demonstration, I present `graph-pes`, a software package I have developed over the course of my DPhil for training, evaluating and applying MLFF models.

Acknowledgements

The past five years have undoubtedly been the most fulfilling of my life; I am incredibly fortunate to have been surrounded by exciting ideas, and to have been supported by many wonderful people.

Within the context of this thesis, my principle thanks go to *Prof. Volker Deringer*, for providing me with the opportunity to undertake this research, his excellent guidance, and fruitful discussions, and to all of my collaborators for their various and valuable contributions.

More widely, the VLD Group has been a fantastic team in which to work and have fun: thank you to everyone for putting up with my idiosyncrasies, bringing me joy, and exposing me to all kinds of new ideas. I am particularly grateful to:

- *Joe*, for a friendship that extends far beyond the office — long may it continue — and a shared enthusiasm for ornamental gourd futures,
- *Louise*, *Natascia* and *Chiheb*, for healthy doses of both cynicism and good humour, and for excellent style and life advice,
- *Andy*, for your infectious enthusiasm, and non-MLFF distractions,
- *Krystian*, for an unexpectedly fresh perspective, your honest opinions and care, and for proofreading this thesis,
- *Daniel*, for your friendship, and frequent comical outbursts,
- and *Zoé*, of course, for all of our adventures. Together with *Nina*, *Nathalie* and *Bertrand*, you all have made me laugh, given great advice, and changed the way I see the world - *merci pour tout.*

I have been very fortunate to spend time in industry around my DPhil studies; at Microsoft Research, thank you to *Gregor* for your support, friendship and guidance, as well as to *Lixin*, *Jean* and *Hannes* for welcoming me into your amazing team. At Valence Labs, thank you to *Rokas* for half a year of intense collaboration, coffee, and conversation, to *Prudencio* and *Hatem* for making it all happen, and to *Francesco*, *Kerstin*, *Jake* and all the gang for your excellent company and insights.

Away from my research, I've been blessed with an amazing set of friends. To *Rob* and *Elysia*: thank you for your wisdom, for dragging me round many a ride both at home and abroad, and for including me in some of the most important days of your lives. To *George* and *Rachel*: thank you for continuing to look out for me, for all the fun we've had down in Devon and Cornwall, and for the honour of being your best man. To *Harri* and *Alice*: thank you for your wholesome energy, love of birds, and for excitedly sharing in many irresponsible sporting endeavours. And to *Vincent*, *Guy* and *Håvard*: thank you, I suppose, for ridiculing me at every turn — I've (mostly) deserved it.

Lastly, to my family, *Mum*, *Dad*, and *Doro*: thank you for your unconditional support. I love you all dearly.

Works

Below, and in chronological order, I list works I have co-authored since commencing my DPhil. Those highlighted in red are explicitly presented and discussed in this thesis. Relevant, per-chapter attributions are provided in the main text.

2023

Chapter 3

Synthetic data enable experiments in atomistic machine learning
John L.A. Gardner, Z. Faure Beaulieu, V.L. Deringer,
Digital Discovery, 2023, Volume 2, Issue 3, Pages 651-662

Coarse-grained versus fully atomistic machine learning for zeolitic imidazolate frameworks
Z. Faure Beaulieu, T.C. Nicholas, John L.A. Gardner, A.L. Goodwin, V.L. Deringer,
Chemical Communications, 2023, Volume 59, Issue 76, Pages 11405-11408

Chapter 2

How to validate machine-learned interatomic potentials
J.D. Morrow, John L.A. Gardner, V.L. Deringer,
The Journal of Chemical Physics, 2023, Volume 158, Issue 12

2024

Chapter 3

Synthetic pre-training for neural-network interatomic potentials
John L.A. Gardner, K.T. Baker, V.L. Deringer,
Machine Learning: Science and Technology, 2024, Volume 5, Issue 1, Pages 015003

Data as the next challenge in atomistic machine learning
C. Ben Mahmoud, John L.A. Gardner, V.L. Deringer,
Nature Computational Science, 2024, Volume 4, Issue 4, Pages 384-387

2025

Assessing zero-shot generalisation behaviour in graph-neural-network interatomic potentials
C. Ben Mahmoud, Z. El-Machachi, K. A. Gierczak, John L.A. Gardner, V.L. Deringer,
Digital Discovery, 2025, Volume 4, Issue 4, Pages 3389-3399

Chapter 4

Distillation of atomistic foundation models across architectures and chemical domains
John L.A. Gardner, D.F. Thomas du Toit,* C. Ben Mahmoud, Z. Faure Beaulieu, V. Juraskova, L.-B. Paşca, L.A.M. Rosset, F. Duarte, F. Martelli, C.J. Pickard, V.L. Deringer,*
arXiv preprint, 2025, arXiv:2506.10956

Understanding multi-fidelity training of machine-learned force-fields
John L.A. Gardner, H. Schulz, J. Helie, L. Sun, G.N.C. Simm,
arXiv preprint, 2025, arXiv:2506.14963

An automated framework for exploring and learning potential-energy surfaces
Y. Liu, J.D. Morrow, C.E. Ertural, N.L. Fragapane, John L.A. Gardner, A.A. Naik, Y. Zhou, J. George, V.L. Deringer,
Nature Communications, 2025, Volume 16, Issue 1, Pages 7666

A foundation model for atomistic materials chemistry
I. Batatia, P. Benner,* Y. Chiang,* A.M. Elena,* D.P. Kovács,* J. Riebesell,* ..., V.L. Deringer, ..., John L.A. Gardner, ..., G. Csànyi,*
The Journal of Chemical Physics, 2025, Volume 163, Issue 18, Pages 184110

Autonomous interpretation of atomistic scattering data
A.S. Anker, John L.A. Gardner, L.A.M. Rosset, A.L. Goodwin, V.L. Deringer,
arXiv preprint, 2025, arXiv:2510.05938

Contents

1. Introduction 1
- 1.1. Potential Energy Surfaces 2
- 1.2. Three kinds of surface 4
- 1.3. Thesis overview 7

2. Machine-learning potential energy surface models 8
- 2.1. Learning a simple potential for two atoms 8
- 2.2. Extending to structures of arbitrary size 18
- 2.3. Message-passing force fields 25
- 2.4. Limitations to perfect accuracy 31
- 2.5. Scaling laws 31
- 2.6. Validating machine-learned force fields 32
- 2.7. graph-pes 37

3. Synthetic pre-training for machine-learned force fields .. 40
- 3.1. A synthetic dataset for bulk amorphous carbon 41
- 3.2. Descriptor-based local energy models 44
- 3.3. Learning per-structure energies 52
- 3.4. Synthetic pre-training 55
- 3.5. Beyond numerical errors: robustness in dynamics 59
- 3.6. Summary and outlook 61

4. Automated distillation of machine-learned force fields . . 63
- 4.1. Foundation models are expensive and slow 63
- 4.2. Accelerating large models 65
- 4.3. Force field distillation 67
- 4.4. Automated potential energy surface exploration 68
- 4.5. Proof of concept 74
- 4.6. Ablation studies 75
- 4.7. Scaling laws for distillation 79
- 4.8. Applications 81
- 4.9. Summary and outlook 86

5. End-to-end learning of symbolic force fields 89
5.1. Simple, separable energy models 89
5.2. Symbolic regression 93
5.3. End-to-end learning of symbolic energy models 94
5.4. Proof of concept: copper 96
5.5. Application: bulk silicon 106
5.6. Application: amorphous silica 112
5.7. Summary and outlook 121

6. Outlook 123
6.1. A high level perspective 123
6.2. The nuts-and-bolts view 124

Appendix 143
A Smooth Overlap of Atomic Positions (SOAP) 144
B Gaussian Process Regression 145
C Symbolic distillation via `PySR` 147

1. Introduction

> *"Whence arises all that order and beauty we see in the world, but which we cannot explain?"*
>
> Isaac Newton,
> *Opticks*,[4] 1704

> *"Nature isn't classical, dammit [...] If you want to make a simulation of nature, you'd better make it quantum mechanical, and by golly it's a wonderful problem, because it doesn't look so easy."*
>
> Richard Feynman,
> *Simulating Physics with Computers*,[5] 1982

The world we humans inhabit is primarily composed of atoms.

An atom in isolation doesn't tend to do very much. Rather, it is the *interactions* between collections of atoms that lead to many of the useful and beautiful structures and phenomena we observe in both the natural and man-made worlds.

The primary goal of Chemistry is to understand these interatomic interactions, and in turn the properties and behaviours of matter that they induce across both micro- and macroscopic scales. Improvements to this understanding allow us to develop new technologies that tackle many important challenges, including new drugs for a healthier populace, improved batteries and solar cells to combat climate change, and greener plastics to reduce pollution and habitat destruction.

In some perverse sense, the study of Chemistry is already complete: for almost a century, we have possessed a theoretical framework – quantum mechanics – that perfectly describes how collections of atoms interact with each other. In principle, therefore, we can answer any question about the behaviour of chemical systems by running an appropriate **computer simulation**. In practice, however, we have continued to predominantly study and explore chemical systems via physical experimentation. Why is this?

In short, computational cost. While extremely elegant, the main result of quantum mechanics, the Schrödinger equation, generates solutions that *cannot be expressed in closed form* for any system containing three or more entities. To make progress, any simulation we do construct must therefore rely on approximations and numerical methods. Due to the inherent non-linearity, coupling and complexity of quantum

mechanics, sufficiently faithful approximations have traditionally been very expensive to construct and evaluate.

However, recent methodological breakthroughs,[6–12] coupled with rapid and sustained increases in computational power,[13] are starting to change this. Concretely, machine-learned force field (MLFF) models are now enabling fast, highly faithful simulations of arbitrary chemical systems over unprecedentedly large time and length scales, in turn enabling the discovery and study of new materials and molecules for diverse applications.[14–20]

In this thesis, I present new methodologies to train and construct MLFF models that improve their interpretability, accuracy and utility when applied to a wide range of chemical domains. By means of an introduction to this thesis, I motivate the MLFF approach below, before describing the various contributions of this thesis in turn.

1.1. Potential Energy Surfaces

> *"The underlying physical laws necessary for the mathematical theory of [...] the whole of chemistry are thus completely known, and the difficulty is only that the exact application of these laws leads to equations much too complicated to be soluble. [...] Approximate practical methods of applying quantum mechanics should be developed, which can lead to an explanation of the main features of complex atomic systems without too much computation."*
>
> Paul A.M. Dirac
> *Quantum Mechanics of Many-Electron Systems*,[21] 1927

In the mid-1920s, Heisenberg[22] and Schrödinger[23] formalised a new theoretical framework for describing physical phenomena at microscopic scales: quantum mechanics. This approach met with tremendous success, explaining previously enigmatic phenomena such as the stability of atoms,[24] quantisation of energy levels,[25] and the UV catastrophe.[26] Indeed, with the aid of a relativistic correction from Dirac,[21] quantum mechanics has long been able to completely describe the structure and behaviour of all chemical systems.

Of the many shocking observations revealed by this theory of matter, two are particularly relevant to the modelling of chemical systems:

1. sub-atomic entities (such as electrons and nuclei) are **not deterministic point masses**, but instead probabilistically distributed "wave packets", described by wavefunctions and exhibiting both particle- and wave-like behaviour.
2. for systems composed of three or more bodies (*i.e.* all chemical systems of interest beyond an isolated hydrogen atom), there are **no analytical solutions** that perfectly describe the state and evolution of the system, in either its ground or excited states.

The latter result arises from the fact that the Schrödinger equation is a coupled, non-linear, partial differential equation. To this day, intense research effort is being spent in attempting to find efficient ways to numerically approximate solutions to this equation for non-trivial, many-body systems.

1.1.1. The Born-Oppenheimer approximation

A very common and successful approach for generating approximate solutions to the Schrödinger equation for most relevant chemical systems is that of the Born–Oppenheimer (BO) approximation.[27]

Chemical systems are composed of two types of entities: nuclei and electrons. While the interactions between nuclei and electrons mean that their motions are formally coupled, the mass difference (nuclei are between 10^3 and 10^5 times heavier than an electron) means that, in practice, we can ignore this coupling without significant loss of accuracy. To anthropomorphise classically: light, and therefore fast-moving, electrons "see" nuclei as near stationary, and can instantaneously adapt to any change in nuclear position. This is the Born–Oppenheimer approximation.

The main corollary of this approximation is that the potential energy of a chemical system is a function only of the nuclear positions. This function is known as the potential energy surface (PES).

1.1.2. Classical nuclei

Under the BO approximation, nuclei are still treated as quantum mechanical entities, with the (squared magnitude of the) nuclear wavefunction describing the probability distribution of the nuclear positions in space.

A further approximation is to assume, again because of their large masses and hence short de Broglie wavelengths, that nuclei are sufficiently localised in space that they can be treated as classical point masses.

This approximation allows us to define a chemical structure as a collection of nuclei at precise locations in space, and to calculate its energy via a single evaluation of the potential energy surface. Furthermore, we can simulate the time evolution of these systems using classical mechanics:

$$\frac{d^2 R_i}{dt^2} = -\frac{1}{m_i}\frac{\partial E(R)}{\partial R_i}$$

where R_i, m_i are the position and mass of the i-th nucleus, and $E(R)$ is the potential energy surface evaluated at the current nuclear configuration, R. The use of the gradient of the PES as a localised, atomic force therefore means that PES models are also often termed "force fields".

1.1.3. Limitations

> *"Eine beliebige Konfiguration von Elektronen und Kernen läßt sich natürlich nicht mit einem allgemeinen Näherungsverfahren behandeln."*
>
> *"An arbitrary configuration of electrons and nuclei cannot, of course, be treated by a general approximation procedure."*
>
> Max Born, J. Robert Oppenheimer
> *Zur Quantentheorie der Molekeln*,[27] 1927

The PES approach allows us to recover a quasi-classical description of chemical systems, trading fidelity for computational efficiency and ease of analysis. However, it is important to note that the above approximations impose a number of limitations:

- **BO breakdown**: the BO approximation breaks down whenever electronic and nuclear motions become significantly coupled. This can occur at high temperatures, in systems containing light nuclei (*e.g.* H, He, Li), and in photochemical and excited state dynamics, where several electronic states approach each other closely or intersect.
- **delocalisation**: the de Broglie wavelength,[28] Λ, of a particle defines the spatial extent of its wavefunction:

  $$\Lambda \propto \frac{1}{\sqrt{mT}}$$

  where m is the mass of the particle and T is the temperature. At low temperatures, and for the lightest of elements, these wavelengths become large, and a classical, point mass approximation is no longer valid (see *e.g.* the behaviour of liquid helium).
- **quantum tunnelling**: very light nuclei can "tunnel" through classically insurmountable barriers in the PES.
- **zero point energy**: even at zero temperature, nuclei have non-zero kinetic energy (due to the Heisenberg uncertainty principle).[29]

Thankfully, nuclear quantum effects can be modelled within the classical nuclei approximation via the use of (admittedly expensive) path integral molecular dynamics,[30,31] and so do not prohibit the use of PES models.

1.2. Three kinds of surface

The approximations set out above are now 99 years old. Since then, a central challenge in atomistic modelling has been finding ways to accurately evaluate the PES for relevant nuclear configurations. While many strategies have been considered in this search, three main approaches have come to dominate:

1.2.1. *Ab initio*

The potential energy surface is mediated by electron–electron and electron–nucleus interactions; these are still very much quantum mechanical in nature, and are dictated by the (electronic) Schrödinger equation under the BO approximation. As their name suggests, *ab initio* methods evaluate the PES for a given nuclear configuration by "solving" this equation from *first principles*. This from-scratch approach ensures that *ab initio* methods are very general, and can work universally across all relevant chemical spaces.

A key problem such approaches need to tackle is how to deal with, and approximate, electron–electron correlations. Many families of *ab initio* methods have been developed that tackle this problem in various ways; of these, the two most prominent are Density Functional Theory (DFT, the workhorse of the field)[32,33] and Coupled Cluster (CC, a "gold standard" reference method).[34–38]

These *ab initio* methods are **slow**, for two reasons. First, rather than being direct functional mappings from nuclear positions to an energy, both rely on iteratively evaluating and updating solutions until some convergence threshold is reached. Second, the individual operations used in each of these iterations are themselves computationally expensive. Concretely, for a system containing N_e electrons, common DFT codes require $O(N_e^3)$ operations to evaluate the PES for a given electron density,[39,40] while CC codes require a pathological $O(N_e^7)$ operations per iteration.[38] These scalings limit the practical application of DFT methods to systems containing a thousand atoms or less, and of CC methods to systems containing fewer than a hundred atoms.

1.2.2. Empirical

Empirical models ignore explicit electrons, and instead define the PES as a sum of simple, interpretable terms that act solely on nuclear positions, and that can be evaluated in a single "forward pass".[41–49] Most empirical models only consider terms corresponding to local interactions, allowing their evaluation cost to scale linearly with the number of atoms in the system, and lending naturally to efficient parallelisation. Empirical models are therefore **fast**: on specialised hardware, empirical models can evaluate the PES 1,000 times per second for a 10k atom system![50]

However, these fixed functional forms also lead to severe limitations:

- **overly simple**: empirical models typically only comprise functions of simple terms (such as pairwise interatomic distances, angles, and torsions). This renders them completely incapable of capturing the subtle but important many-body interactions present in many chemical systems.
- **inaccurate forms**: even in systems where simple interactions dominate, the exact functional forms for the various terms are themselves often inaccurate, *e.g.* modelling chemical "bonds" as classical springs. A particularly

common corollary is that most empirical models are incapable of modelling chemical reactivity.
- **lack of generalisation**: all empirical models contain free parameters that modulate the strength and lengthscales of the various interactions they model. The values of these parameters must be carefully chosen, typically so that the model reproduces experimental observations or *ab initio* energies and forces. Crucially, these parameters are chosen in the context of a specific system and under specific conditions: outside of this context, correct behaviour is not guaranteed.

In summary, while empirical models can be fit to recover experimentally observed macroscopic properties, they generally fail to faithfully model the microscopic behaviour of the system. This severely limits their utility, especially for mechanistic studies.

1.2.3. Machine-learned

Machine-learned force fields (MLFFs) employ a similar approach to empirical models: they ignore explicit electrons, and generate energy predictions solely as a function of nuclear positions in a single forward pass. The major difference is that MLFFs **do not** pre-suppose a fixed functional form. Instead, they incorporate very flexible model components, typically containing very many parameters (usually 10^4–10^7, *cf.* 2–100 for empirical models), allowing them to simultaneously model two-, three- and many-body interactions of arbitrary shape and scale.

MLFFs are "trained" (*i.e.* values for each model parameter are chosen) so as to reproduce the energies and forces generated by a reference *ab initio* method on a dataset of representative chemical structures. While the flexibility of MLFFs' functional forms allows them to successfully interpolate these data points, and hence to approach *ab initio* accuracy, it also introduces several challenges:

- **data efficiency**: to accurately reproduce a given reference method, MLFFs typically need to be trained on large and diverse datasets.[51–53] The (major) bottleneck and cost in training MLFFs is therefore the *ab initio* labelling of the training data.
- **speed**: MLFF evaluations typically require (orders of magnitude) more mathematical operations compared to empirical models, due to their larger parameter counts and more complex internal processing. Despite strong synergy with modern hardware accelerators, MLFFs are therefore (much) slower than empirical models.
- **interpretability**: most MLFF architectures are black-boxes: with no easily interpretable function form, there are no guarantees that MLFF models will exhibit qualitatively correct behaviour when applied to systems that are not within their training set.[54,55]

Despite these challenges, MLFFs have met with great success! Provided they are carefully trained and validated, MLFFs can quickly drive simulations at, or very near, *ab*

initio accuracy, over unprecedentedly large time and length scales (10^3–10^6 atoms over ns–ms).[15,18,56] As such, they are rapidly transforming the field of atomistic modelling, driving the discovery of new, and improving our understanding of existing, molecules and materials for diverse and important applications.[14–20]

1.3. Thesis overview

Above, I highlighted three key drawbacks of MLFFs: data inefficiency, sub-optimal speeds, and a lack of interpretability. In the following chapters, I present work that addresses and mitigates each of these in turn:

- In Chapter 3, I explore the use of "synthetic data" as a means to pre-train MLFFs, improving their accuracy, reducing the amount of expensive *ab initio* data required to train them, and making them more robust during molecular dynamics simulations.
- In Chapter 4, I present a means to automatically "distil" information from a large, slow MLFF into a smaller, faster one, preserving simulation accuracy while providing speed-ups of up to 100×.
- In Chapter 5, I present work that uses an MLFF-inspired approach to learn symbolic PES models in an end-to-end manner, recovering the speed and interpretability of existing empirical models while (significantly) improving their accuracy.

In all of the above, I train MLFF models using gradient descent on datasets of labelled structures. I therefore begin in Chapter 2 by motivating this approach, starting with a toy example to illustrate key concepts, before training a series of increasingly accurate models for a real-world use case. To do this, I rely upon and present `graph-pes`,[57] a software package I have developed over the course of my DPhil, and that implements many MLFF architectures, training procedures and validation pipelines.

Finally, in Chapter 6, I discuss the implications of the work presented in this thesis, and provide some outlooks and opinions on the future of MLFF research and their application.

2. Machine-learning potential energy surface models

Modern, state-of-the-art MLFFs are graph neural networks (GNNs), trained via gradient descent to reproduce the PES labels of some reference method over a large dataset of training structures. By means of a methods review, I break down, explain and motivate this approach below.

I start by considering how to machine-learn a simple PES model for a molecule composed of two carbon atoms. I use this toy system to introduce the key concepts of a loss function, optimisation via gradient descent, the use of neural networks in PES models, and over-fitting.

Training more sophisticated PES models for larger systems is a straightforward extension of these techniques; I continue by generalising to modelling the system of bulk carbon, and in the process introduce means of representing structures of arbitrary size, linearly scaling models that act on these representations, and properties that these models should have to be both accurate and physically correct. Ultimately, I train and compare several models, ranging from a simple pair potential to a state-of-the-art GNN, to demonstrate important recent developments in MLFF architectural design.

Before using an MLFF to simulate a system of interest, it is vital to ensure that the model is "trustworthy". Beyond numerical accuracy, I therefore discuss various validation strategies, and apply them to the models I train.

Finally, I introduce and describe `graph-pes`, a software package I have developed over the course of my DPhil, that implements many MLFF architectures, training procedures and validation pipelines, and that I have used to train all models presented in this chapter.

2.1. Learning a simple potential for two atoms

I start by considering the simplest non-trivial chemical system: two atoms of the same element, arranged arbitrarily in space. Such a system has a single degree of freedom: the interatomic distance, r. In Figure 2.1, I plot the total energy of a C_2 dimer, evaluated at various interatomic distances using DFT with the LDA functional,[33] as taken from *Deringer & Csányi, 2017*.[58]

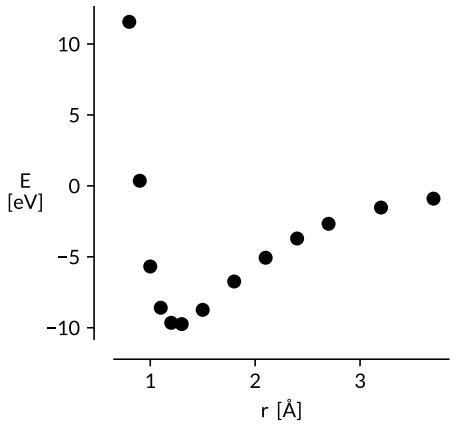


Figure 2.1: *formation energy of a C_2 dimer, evaluated at various interatomic distances, r.*

To find a model that fits this data well, we might start by trying to guess the functional form of this curve, for instance:

$$\hat{E}(r; a, b) = \frac{a}{r^6} - be^{-r}$$

where the polynomial term models short-ranged repulsions, the exponential term models longer-ranged attractions, and where a and b are unknown **parameters** modulating the strength of these interactions that we need to determine. Note that this functional form, while related to the Buckingham potential, is contrived: these terms do not correspond to any physically motivated interactions, and have merely been chosen for their simplicity, and ability to fit the data reasonably well.

Beyond arbitrarily picking parameters until we find ones that "look good", we can make progress by defining a **loss function**: a quantitative measure of how well our model fits the data. A common choice is the root-mean-square error (RMSE):

$$\mathrm{RMSE}_f(\mathcal{D}) = \sqrt{\frac{1}{|\mathcal{D}|} \sum_{(x_i, y_i) \in \mathcal{D}} [f(x_i) - y_i]^2}$$

where:

- $\mathcal{D}$ is a **dataset** composed of $|\mathcal{D}|$ pairs of input-output data, (x_i, y_i)
- $f(x_i)$ is the model's output at input x_i, *i.e. the "prediction"*
- y_i is the true function's value at input x_i, *i.e. the "label", or "ground truth"*

In Figure 2.2, I plot the *loss landscape* for this empirical model, *i.e.* the RMSE on the dataset from Figure 2.1 as a function of the parameters a and b. Each (x, y) location in the plot corresponds to a unique model, colour-coded according to its loss (more red = lower loss = better model). The lowest point on this loss landscape corresponds to the model that best fits the data (marked with a star).

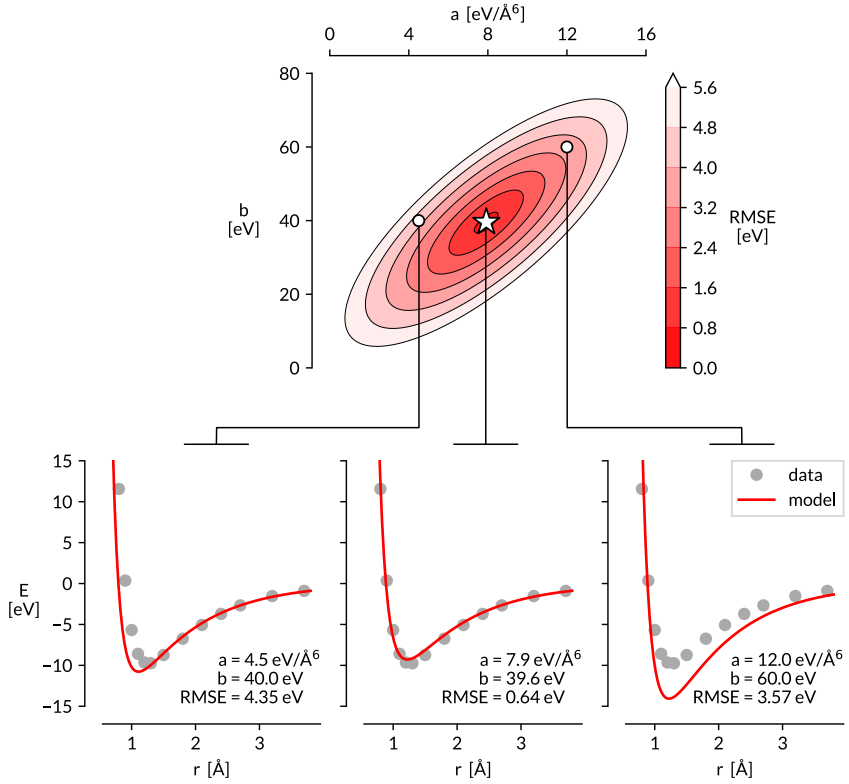


Figure 2.2:
(top) *RMSE loss landscape for an empirical PES model*
(bottom) *three example models with specific values for a and b, corresponding to the marked points above*

2.1.1. Gradient-based optimisation

Exhaustively searching through parameter space for the global minimum is expensive, and quickly becomes infeasible as the number of model parameters increases ("standard" MLFF parameter counts range from 10k-50M).

An alternative, scalable, and dominantly used approach to find optimal parameter values in many settings is to perform a local search using gradient information. Such a fitting strategy involves:

1. starting from some initial guess for each of the model parameters, θ_0
2. calculating the gradient of the loss function with respect to each of the model parameters at their current values:

$$\nabla_{\theta_t} = \left.\frac{\partial L}{\partial \theta}\right|_{\theta=\theta_t}$$

3. updating the parameters by some small amount in the opposite direction of this gradient, for instance using some small multiplicative factor (frequently called the "learning rate"), α:

$$\theta_{t+1} = \theta_t - \alpha \nabla_{\theta_t}$$

4. repeating steps 2 and 3 until no further improvement, or some other stopping criterion is met.

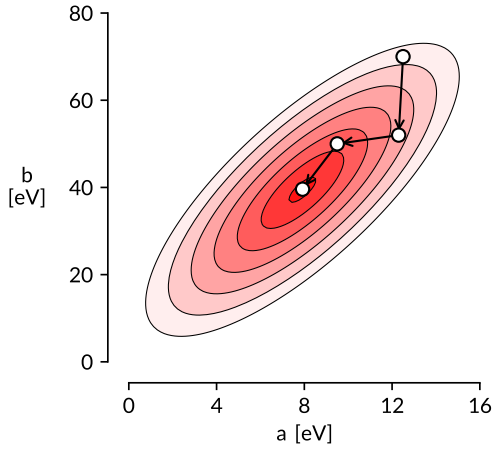


Figure 2.3: *gradient descent trajectory through parameter space for an empirical PES model*

In Figure 2.3, I plot an example trajectory that so-called "gradient descent" might take through parameter space to "learn" the best-fitting model for this simple, two parameter model. Whereas exhaustive, grid-based approaches require $O(k^d)$ model evaluations (where k is the number of grid points sampled for each of the d parameters), gradient-based methods typically require a constant number of model evaluations, independent of the number of parameters. This makes gradient-based methods extremely scalable.

While it may seem cumbersome to derive and calculate the gradient of the loss with respect to each model parameter, modern computational frameworks such as `PyTorch`[59] and `Jax`[60] make this straightforward, and parallelisable, via a process called (reverse-mode) "automatic differentiation".[61]

The above loss landscape is very simple. In practice, several caveats apply when applying gradient descent to real-world MLFFs:

- **non-convexity**:
  "Real" loss landscapes are often (highly) non-convex:[62,63] running gradient descent from different starting points (*i.e.* different random parameter initialisations) will typically find different local minima, many of which will often be undesirable (*i.e.* well above the true global minimum). Taking the best from several differently initialised models is a common strategy used to mitigate this problem.
- **dimensionality**:
  The model we are considering above only has two parameters, and so

the loss surface is two-dimensional. In practice, most MLFFs have many more parameters, and so the loss surface is of very high dimensionality. This exacerbates the non-convexity problem: careful design of model architectures, with appropriate initialisation,[64] normalisation[65,66] and residual connections[67] is therefore required to "smooth out" the loss landscape and make it easier to converge to a good set of parameter values.[63]

- **batching**:
  Evaluating the loss function over the full dataset can be computationally expensive. Instead, good estimates of the true loss–parameter gradients can often be estimated using small subsets of the data, called "batches". Faster convergence (measured in wall-clock time) can often be achieved by taking more steps using smaller batches. Batching is also a necessity as dataset sizes increase, and it becomes impossible to perform a single forward pass over the entire dataset in a single step due to memory constraints.
- **vanilla gradient descent is slow**:
  The simple descent algorithm described above is a memoryless process — this makes convergence to minima slow, and prone to *e.g.* cycle convergence.[68] More sophisticated optimisers, most notably `Adam`,[69] keep track of past gradient values, and can be used to considerably improve convergence speed (as measured by both wall-clock time and number of model evaluations), particularly in settings where noise is induced via batching.

2.1.2. Neural networks & the bitter lesson

> *"With four parameters I can fit an elephant, and with five I can make him wiggle his trunk."*
>
> John von Neumann[70]

> *"The biggest lesson [...] from 70 years of AI research is that **general methods** that leverage computation are ultimately the most effective, and by a large margin."*
>
> Richard Sutton,
> *The Bitter Lesson*[71], 2019

So far, I have adopted an empirical modelling approach by pre-supposing a specific functional form for our model. While such models require a small number of interpretable parameters (like Mr. von Neumann's elephant), and are guaranteed to exhibit **qualitatively** correct behaviour under limiting conditions, they are **rarely accurate enough** to be useful in practice.[72,73]

A more general approach, employed extensively throughout machine learning, is to use *"universal function approximators"* (UFAs) for all learnable components of a model. As their name suggests, UFAs are able to approximate, to arbitrary accuracy, any continuous mapping from a fixed-dimensional input space to a fixed-dimensional

output space, given sufficient capacity and appropriate training. Model classes that are provably UFAs include: polynomial expansions,[74] radial basis function networks,[75] kernel methods,[76] Gaussian Processes,[77] and artificial neural networks (NNs), most notably multi-layer perceptrons (MLPs).[78]

While UFA-based models are substantially more flexible than empirical models, this flexibility comes at a cost: they generally require a significantly larger number of parameters, increased computational resources for training and inference, and careful regularisation to avoid over-fitting. Nevertheless, UFAs find favour in practice, and in particular neural networks have emerged as a dominant paradigm, owing to a favourable combination of expressive power, controllable inductive biases, compatibility with large-scale, gradient-based optimisation methods, and synergy with modern hardware accelerators (the oft cited "hardware lottery").[79]

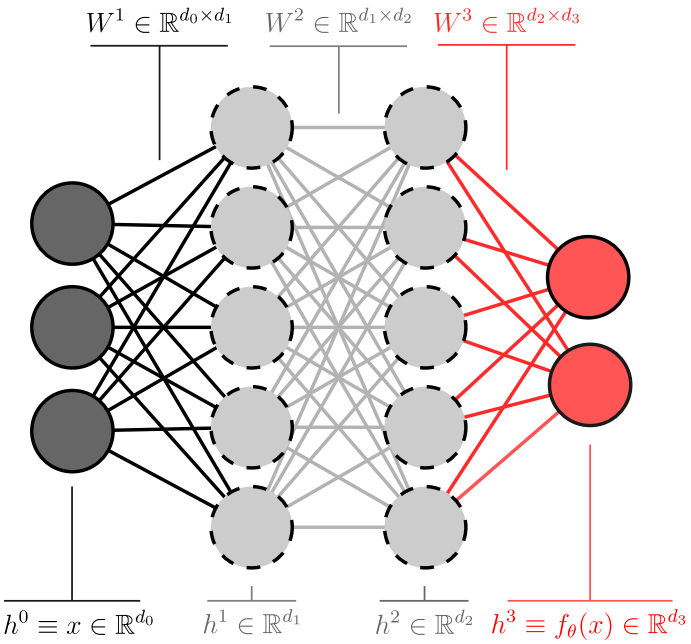


Figure 2.4: *diagrammatic representation of a multi-layer perceptron (MLP) that maps an input vector, $x \in \mathbb{R}^3$ (black circles) via two hidden layers each of width 5 (grey circles) to generate an output vector, $y \in \mathbb{R}^2$ (red circles), i.e.* $\mathrm{MLP}_{[3\to5\to5\to2]}$ *Each line connecting two nodes corresponds to a single element of the appropriate weight matrix. Biases are omitted for clarity.*

Here, I focus on multi-layer perceptrons as a representative neural-network model (Figure 2.4). An MLP defines a function $f_\theta : \mathbb{R}^{d_0} \to \mathbb{R}^{d_L}$ as a composition of affine

transformations and element-wise non-linearities that act to iteratively transform representations of data, h^l:

$$\begin{aligned} h^0 &\equiv x, \\ h^l &= \sigma\big(W^l h^{l-1} + b^l\big), \quad l = 1, ..., L-1, \\ f_\theta(x) &= W^L h^{L-1} + b^L \end{aligned}$$

where:

- $x \in \mathbb{R}^d$ is the input data point
- W^l, b^l are the weight matrix and bias vector for the l-th layer
- $\sigma : \mathbb{R} \to \mathbb{R}$ is a point-wise non-linear "activation function", for instance `sigmoid`, `tanh`, or `ReLU`
- $\theta = \{W^l, b^l\}_{l=1}^L$ are the full set of model parameters

Under mild conditions on σ, such an MLP with $L \geq 2$ is provably a UFA given sufficiently "wide" layers.[78]

Common choices for the activation function include the `sigmoid`, `tanh`, Rectified Linear Unit (`ReLU`), and Continuously-differentiable Exponential Linear Unit (`CELU`):

name	`sigmoid`	`tanh`	`ReLU`	`CELU`
$\sigma(x) =$	$\frac{1}{1+\exp(-x)}$	$\frac{e^x - e^{-x}}{e^x + e^{-x}}$	$\max(0, x)$	$\begin{cases} x & x \geq 0 \\ e^x - 1 & x < 0 \end{cases}$

In Figure 2.5, I fit an MLP to the same dimer dataset, and compare to the empirical model above. I use two hidden layers, each containing 256 neurons for a total of 65k parameters, `CELU` activations to ensure smoothness, full-batch `Adam` optimisation with an initial learning rate of 0.01 and `CosineAnnealingLR`[80] learning rate decay over 10k epochs.

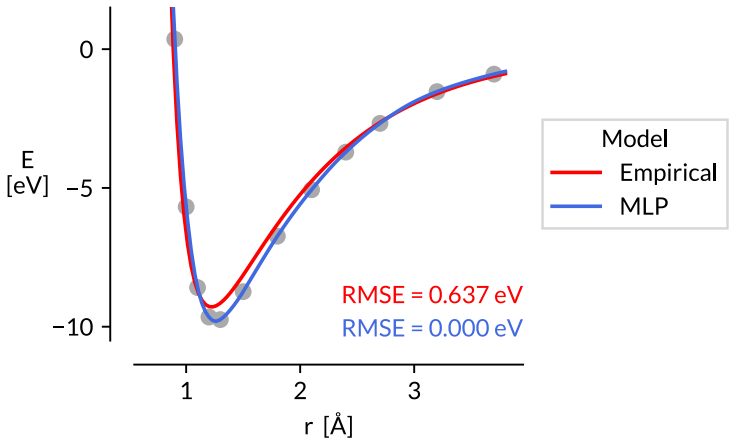


Figure 2.5: *a neural network (NN) fits the dimer curve much more accurately than an empirical model*

The MLP fit (blue line) is near perfect, with an RMSE of only 1.05 μeV: this is 5 orders of magnitude better than the empirical model. Despite this significant improvement in accuracy, it is important to note two downsides to the NN approach:

1. **increased cost**:
   A proxy for the cost of a model is the total number of floating point operations (FLOPs) required to evaluate it. A single evaluation of the empirical model requires 15 FLOPs (assuming 8 FLOPs are required for exponentiation as per *Huss et al.*[81]). In contrast, the MLP model requires $\approx 10^5$ FLOPs per evaluation. While specialised hardware accelerators (in particular GPUs and TPUs) can provide significant speed-ups via parallelisation, the NN model is still ≈20 times slower to evaluate as measured in wall-clock time on representative hardware (M2 MacBook Pro, and NVIDIA A100 GPU).
2. **a lack of interpretability**:
   The symbolic nature of empirical models makes it very straightforward to interpret the functional form of the model, and the physical meaning of each parameter. Simple inspection of the closed form expression can also demonstrate the model's behaviour is guaranteed to be physically correct under limiting conditions (*e.g.* at both very small and very large interatomic distances). In contrast, and by definition, UFAs such as NNs can represent arbitrary functions, such that there is no fixed functional form to interpret. This makes is (extremely) difficult to analyse the model's behaviour in any way other than via repeated, "black-box" trial evaluations.[82] For the specific case of NNs, the values of individual parameters are not generally interpretable: I demonstrate this in Figure 2.6, by altering two arbitrarily chosen parameters in the MLP model, while keeping all other parameters fixed at their optimal values.

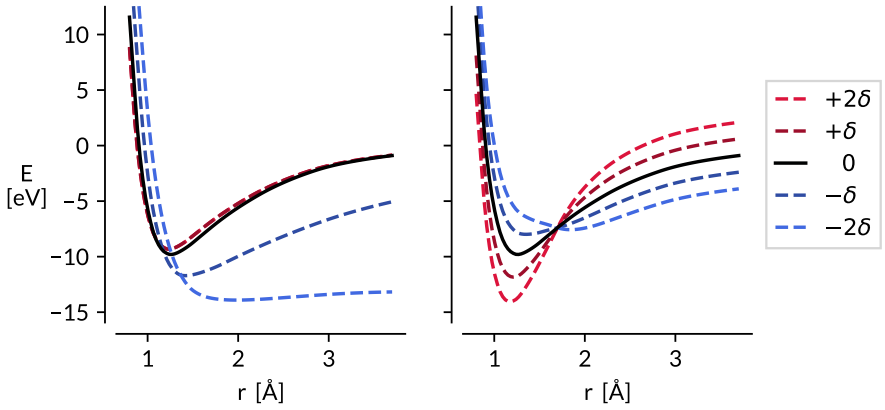


Figure 2.6: *a scan over 5 values for two arbitrarily chosen parameters in the MLP model, keeping all other parameters fixed at their optimal values. (LHS: $W^1_{18,4}$, RHS: $W^2_{30,1}$, $\delta = 5$)*

In Chapter 5, I focus on addressing these issues by converting learned MLP components into symbolic representations.

2.1.3. Over-fitting

So far, I have trained and evaluated models on the same dataset. In all realistic scenarios, this gives an overly optimistic estimate of the model's actual performance on unseen data points, and can mislead us into believing that the model is more accurate and useful than it actually is.

In practice, it is therefore important to evaluate models on data they have never seen during training. This provides a more realistic estimate of the model's actual performance, and can also help to unmask the phenomenon of **overfitting**, whereby a model preferentially fits to, or memorises, the training data, at the expense of poor generalisation to new data points. I present a (contrived) example of what overfitting might look like for this simple 1D system in Figure 2.7.

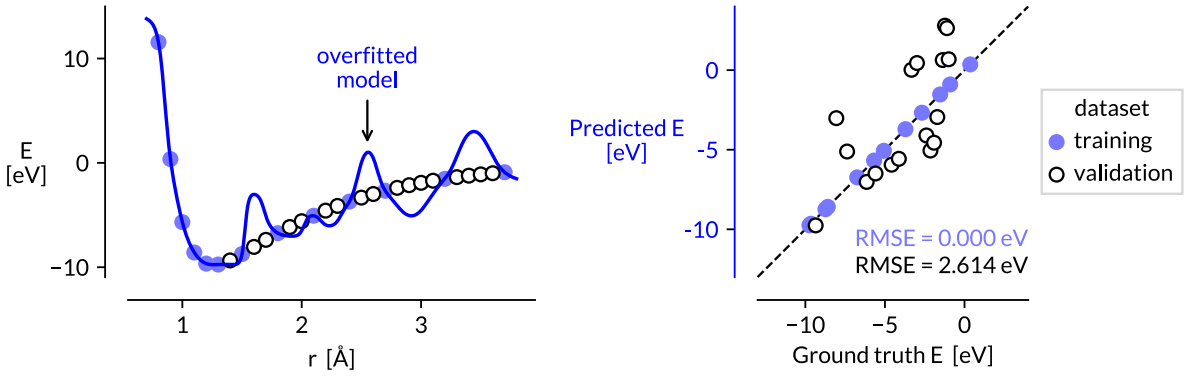


Figure 2.7: *an overfitted pair potential (blue line) passes through all training points near-perfectly, but performs poorly on a hold-out test set*

On the left, I plot a distorted pair potential that nevertheless passes through all training points (blue) near-perfectly. However, on a test set of interatomic distances not seen during training, the model's performance is poor. In general, PESs are multi-dimensional objects, and explicit visualisations in 2D are not possible. **Parity plots** (Figure 2.7 RHS) are a useful visualisation tool that circumvents this limitation: points that lie on the diagonal $y = x$ correspond to data points for which the model's output is in perfect agreement with the true value. Points that deviate from this line indicate non-zero model error.

While the above example is somewhat contrived, overfitting poses a very real problem when moving to realistic chemical systems and MLFFs (see below). Many ways to combat overfitting exist, including model-specific forms of regularisation (see *e.g.* Chapter 3.2.4). A very general approach, and one used throughout the rest of this thesis, is **early stopping**. In this approach, a small amount of validation data is held out from the training set, and never explicitly used to calculate gradients or update model parameters. At regular intervals during training, the model's performance on the

validation set is evaluated. Once the validation loss plateaus, or even starts to increase, training is stopped early, and the model with the lowest validation loss is selected.

Data splitting strategies

A common and useful strategy is to define *in-domain* and *out-of-domain* test splits. In-domain splits sample from the same distribution as the training data, and measure how well the model interpolates within the training distribution. Out-of-domain splits, in contrast, deliberately contain structures that differ from the training data in some meaningful way (*e.g.* different chemical compositions, phases, or thermodynamic conditions), and provide a more stringent test of the model's ability to generalise beyond its training distribution. The degree to which a model can successfully extrapolate to out-of-domain structures is a key indicator of its reliability in practical applications.

When training data are sourced from molecular dynamics (MD) trajectories, an additional subtlety arises: consecutive snapshots along a trajectory are highly correlated, and random splitting would therefore produce test sets that are artificially similar to the training data. In such cases, it is important to split *by trajectory* or *by time* rather than by individual structure, ensuring that the test set contains genuinely independent configurations. Throughout this thesis, I adopt splitting strategies appropriate to the data source in question, as detailed in the relevant sections below.

2.2. Extending to structures of arbitrary size

Above, I fit a very simple PES model to a very simple system. At a high level, this involved a four step process:

1. collecting a dataset of labelled structures
2. defining a model that maps structures to energies
3. defining a loss function that measures how well the model fits the data
4. using gradient descent to find model parameter values that minimise this loss over the dataset

Interesting and important chemical systems are composed of more than just two atoms. Fortunately, to train accurate PES models that can act on structures of arbitrary size, we need only make small adjustments to the process outlined above. Concretely, beyond the single interatomic distance required to describe a C_2 molecule, we need a means to represent structures containing an arbitrary number of atoms. With this in place, we also need to adapt the architecture of our model so that (i) it can act on this new representation to produce a scalar, per-structure energy output, and (ii) we can ensure that these predictions are guaranteed to be physically correct.

Below, I do exactly this, introducing the geometric graph representation of chemical structures and the concept of linearly scaling models that act on such.

2.2.1. Chemical structures

Interesting chemical systems are composed of (significantly) more than just two atoms. The most fundamental representation of an isolated, uncharged chemical structure is as an annotated point cloud of atoms:

$$S = \{(\vec{r}_i, z_i) : 1 \leq i \leq N\}$$

where $\vec{r}_i \in \mathbb{R}^3$ is the position of the i-th atom, and $z_i \in \mathbb{Z}^+$ is its atomic number. When modelling bulk matter, we typically also include a "unit cell", $C \in \mathbb{R}^{3\times3}$, with periodic boundary conditions, such that the structure is repeated periodically in a manner that tiles 3D space.

Beyond the total potential energy, $E \in \mathbb{R}$, several other properties of the PES for a given structure are also important, and are frequently used as training labels:

- **atomic forces**: the gradients of the total energy with respect to perturbations of the atomic positions are interpreted as classical force vectors on each atom. These can be used to drive structural relaxations (*i.e.* via gradient based energy minimisation) and molecular dynamics simulations:

  $$\vec{F}_i(S) = -\frac{\partial E(S)}{\partial \vec{r}_i}$$

  where $\vec{F}_i \in \mathbb{R}^3$ is the force vector for the i-th atom.
- **stress tensors**: the gradient of the total energy with respect to perturbations of the unit cell indicate de/compressive stresses on bulk structures. These can be used to identify ground-state densities, and to help drive molecular dynamics under constant pressure conditions. Assuming a perturbation, $\lambda \in \mathbb{R}^{3\times3}$, that acts to distort the unit cell and atomic positions according to:

  $$\Big(R, C\Big) \to \Big(R(I+\lambda), C(I+\lambda)\Big)$$

  elements of the stress tensor, $\sigma \in \mathbb{R}^{3\times3}$, are given by:

  $$\sigma_{ij} = \frac{1}{V} \cdot \frac{\partial E}{\partial \lambda_{ij}}\bigg|_{\lambda_{ij}=0}$$

  where $I \in \mathbb{R}^{3\times3}$ is the identity matrix, $R \in \mathbb{R}^{N\times3}$ are the atomic positions, and V is the volume of the unit cell.

2.2.2. Physical models

Naïvely, we could apply any model architecture capable of acting on point clouds to the task of learning a PES. In practice, however, we typically want to restrict the class of functions we use to those that display the correct "inductive biases". These ensure that the model learns efficiently, and that it is guaranteed to be "physically correct" no matter the values of its parameters. I detail these inductive biases below:

We observe that the universe is isotropic and homogeneous. As such, all scalar properties of a system are invariant to rigid translations and rotations. Energy is no exception, and hence PES models, which we can write as $E = E(R, Z, C)$ where $R \in \mathbb{R}^{N\times 3}$ are vectorised atomic positions, $Z \in \mathbb{Z}^N$ are vectorised atomic numbers, and $C \in \mathbb{R}^{3\times 3}$ is the unit cell, must obey:

$$[1]\ \ E(R, Z, C) = E(R + T, Z, C) = E(R, Z, C + T)$$
$$[2]\ \ E(R, Z, C) = E(RW, Z, CW)$$

for any translation vector $T \in \mathbb{R}^3$ and rotation matrix $W \in \mathrm{SO}(3)$. Furthermore, rotationally invariant energy predictions naturally and correctly lead to rotationally equivariant force predictions:

$$\vec{F}_i(R, Z, C)W = \vec{F}_i(RW, Z, CW)$$

Direct inspection of the Schrödinger equation reveals two further important inductive biases. First, the total energy of a system is independent of the *order* in which atoms are considered. Permutation of atoms in any vectorised representation of the structure should therefore have no effect:

$$E(R, Z, C) = E(PR, PZ, C)$$

for any permutation matrix P. Second, all terms in the Schrödinger equation are continuous with respect to the nuclear positions (providing that no two nuclei occupy the same position). A direct consequence of this is that the total energy is also a continuous function of the nuclear positions, and hence atomic forces are well defined everywhere.

Of course, PES models can be learned using architectures without these baked-in inductive biases, but this lack of physicality can impede their usefulness (see Chapter 4 for a more thorough discussion).[83–86]

2.2.3. Linear scaling: the locality assumption

Beyond being physically correct, MLFF models are only useful if they can be evaluated efficiently. Many experiments in computational materials modelling require the (repeated) evaluation of PES models on large ($N \gg 1000$ atoms) systems. In order to be useful in this context, the evaluation cost (in time and memory) of a MLFF model must scale linearly, $O(N)$, with the number of atoms in the system, in contrast to *e.g.* the $O(N^3)$ cost of typical DFT codes.

This linear-scaling constraint immediately precludes approaches that model global interactions, where *e.g.* summing over interactions between **all** pairs of atoms costs at least $O(N^2)$ in real space, and $O(N \log N)$ in reciprocal space.[87]

The dominantly used strategy to achieve linear-scaling is to break down the total energy into a sum of **local, per-atom** energy contributions:

$$E(S; \theta) = \sum_{i\in S} \varepsilon\Big(\mathcal{N}_i(S); \theta\Big)$$

where $\mathcal{N}_i(S)$ is the local neighbourhood of the i-th atom in the structure S, typically defined as the set of atoms within some cutoff radius distance c:

$$\mathcal{N}_i(S) = \left\{ \left(\vec{r}_j, z_j\right) \ : \ \|\vec{r}_i - \vec{r}_j\| < c \right\}$$

Modelling the total energy as a sum over atomic contributions is a natural and effective way to achieve linear-scaling, and has been very successful in practice. However, it is important to note that:

1. there is no quantum mechanical observable that corresponds to the local energy of an atom, and so these have no physical meaning. It is not necessarily obvious whether there is a unique way to partition the total energy into local contributions.[88,89]
2. quantum mechanical interactions are only mostly local.[58,90] Models based on this assumption are fundamentally limited in their ability to perfectly model QM.[58,91]

A natural way to encode locality is to represent structures as (geometric) graphs:

$$G = (S, T)$$

where S is as above, and T is a set of directed edges:

$$T = \left\{ \left(i, j, \mathrm{Im}_j\right) : i, j \in S, r_{ij} < c \right\}$$

where $r_{ij} = \|\vec{r}_i - \vec{r}_j\|$ is the euclidean distance between atoms i and j, and Im_j is the unit cell image in which neighbour j is located. Models that act only on the nodes and edges of such graphs are guaranteed to be local, and hence linearly-scaling, by construction.

2.2.4. Pair potentials as local energy models

With the above preliminaries in place, we are now situated to construct a linearly scaling, physically correct MLFF model that can generate predictions for, and hence be trained on, structures of arbitrary size. Here I consider the simplest possible non-trivial model that does this: an extension of the pair potential model introduced above.

Any model that makes energy predictions as a sum over finite-body-order terms:

$$E = \sum_{i \neq j} E_{2b}\left(\vec{r}_i, \vec{r}_j\right) \ + \ \sum_{i \neq j \neq k} E_{3b}\left(\vec{r}_i, \vec{r}_j, \vec{r}_k\right) \ + \ \ldots$$

can be re-written as a sum over per-atom contributions:

$$\varepsilon_i = \frac{1}{2} \cdot \sum_{j} E_{2b}\left(\vec{r}_i, \vec{r}_j\right) \ + \ \frac{1}{6} \cdot \sum_{j,k} E_{3b}\left(\vec{r}_i, \vec{r}_j, \vec{r}_k\right) \ + \ \ldots$$

Pair potentials are a special case of such models, where only the two-body term is considered. By incorporating a suitable envelope function:[6]

$$E_{2b}(\vec{r}_i, \vec{r}_j) = V(r_{ij}) \cdot \phi_c(r_{ij}) \qquad \text{with} \quad \phi_c(r \geq c) = 0$$

we can easily arrive at a linearly-scaling, local energy model that uses a single pair potential, $V(r)$ to model the energy for systems of arbitrary size:

$$E = \sum_i \varepsilon_i \quad \text{with} \quad \varepsilon_i = \frac{1}{2} \sum_{\substack{j \neq i, \\ r_{ij} < c}} V(r_{ij}) \cdot \phi_c(r_{ij})$$

I demonstrate the performance of such a model in Figure 2.8 by using:

- the same functional form as above: $V(r) = \text{MLP}_{[1 \to 256 \to 256 \to 1;\ \texttt{CELU}]}(r)$
- a cosine envelope function with cutoff radius $c = 5.0$ Å:

$$\phi_c(r) = \begin{cases} \frac{1}{2} \cdot \left[1 + \cos\left(\frac{\pi r}{c}\right)\right] & r < c \\ 0 & r \geq c \end{cases}$$

- disjoint training and testing sets, each composed of 400 randomly sampled bulk structures taken from the `C-GAP-17`[58] dataset. While there is naturally some degree of overlap in the configurations sampled, the structures in `C-GAP-17` are very varied — spanning crystalline, amorphous, liquid and surface geometries — ensuring genuine novelty in the test set
- a loss function that is a weighted sum of component-wise force RMSE and energy RMSE as normalised by the number of atoms in each structure:

$$\mathcal{L}(B) = \sqrt{\frac{1}{|B|} \sum_{S \in B} \frac{\left(\hat{E}(S) - E(S)\right)^2}{|S|}} + \frac{1}{4} \sqrt{\frac{1}{3 \sum_{S \in B} |S|} \sum_{S \in B} \sum_{i \in S} \sum_{\alpha \in \{x,y,z\}} \left(\hat{F}_{\alpha,i}(S) - F_{\alpha,i}(S)\right)^2}$$

  where B is a batch of structures, $\hat{E}(S), E(S) \in \mathbb{R}$ are the predicted and ground truth energies for structure S, and $\hat{F}_{\alpha,i}(S), F_{\alpha,i}(S) \in \mathbb{R}$ are the predicted and ground truth force components along the α-axis for atom i in structure S.
- the `Adam` optimiser, with a learning rate of 0.001, a `CosineAnnealingLR` scheduler, a batch size of 100 structures, for a total of 4k epochs, and retroactively selecting the best performing model as measured on a held-out validation set.

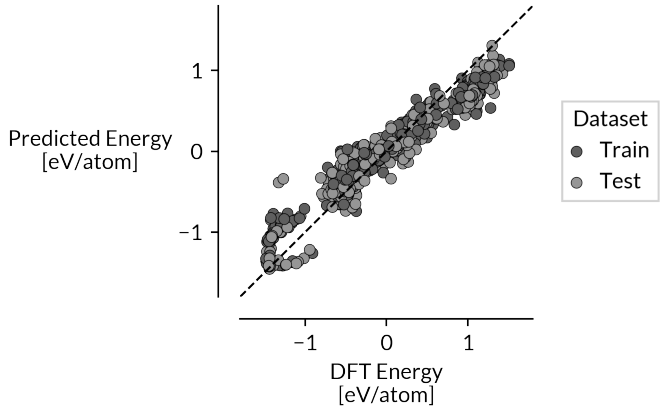


Figure 2.8: *parity plot for a pair potential model trained on a subset of the* `C-GAP-17` *dataset*

Two things are immediately apparent. First, there is a clear correlation between the ground truth energy and the model's predictions — in this sense, our model is a success: training has clearly led to a model with some predictive power. Second, however, the model is far from perfect, with a test-set energy mean absolute error (MAE) of 189 meV/atom, and a test-set force MAE of 1.59 eV/Å. Ultimately, this poor performance is expected: the simple pairwise architecture we are using here is incapable of modelling important many-body effects such as directionality, delocalisation, hybridisation and polarisation, all of which are present in the PES of bulk carbon.

This does not, however, invalidate the approach: MLP-parameterised models that act on graph representations can and do achieve very strong performance (see below) — we merely need to change the "architecture" of our model to something more powerful. Below, I consider increasingly sophisticated models that improve over this simple pairwise architecture, building up to internally equivariant, message-passing graph neural networks.

2.2.5. Embedded atom models

Beyond pair potentials, embedded atom models (EAMs) aim to capture many-body effects by building a local density term, ρ_i, and applying some non-linear function to it:

$$\varepsilon_i = f(\rho_i) \quad \text{with} \quad \rho_i = \sum_{j \neq i} g(r_{ij}) \cdot \phi_c(r_{ij})$$

Provided that f is non-linear, these models are provably more "expressive" than simple pairwise models. Empirical EAM models use a single scalar for the density field, $\rho \in \mathbb{R}$.[47] A natural, and deep-learning-inspired, generalisation is to allow this to be a collection of h "features" for each atom, $\rho_i \in \mathbb{R}^h$.

In Figure 2.9, I train such a generalised EAM model with $h = 64$ features of the density field, $g = \text{MLP}_{[1\to190\to190\to h;\ \text{CELU}]}$, $f = \text{MLP}_{[h\to190\to190\to1;\ \text{CELU}]}$ and using the same radial cutoff, envelope function, train/test split and loss function as above.

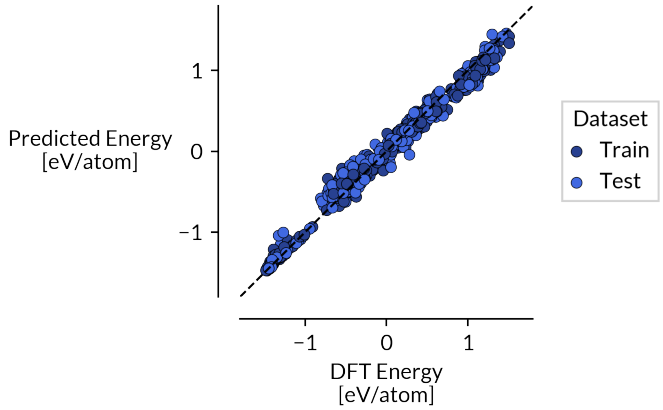


Figure 2.9: *parity plot for an EAM model trained on a subset of the `C-GAP-17` dataset*

With more learnable components, and a more powerful expressive form, the EAM model improves by a large margin over the simple pair potential, with a test-set energy MAE of 85 meV/atom, and a test-set force MAE of 1.16 eV/Å.

However, this is still far from "perfect" performance: ignoring early stopping criteria and continuing to train for longer (see Figure 2.14 below) leads to a training error that is still non-zero: despite relying on universal function approximators for each learnable component, this architecture **is not** a universal function approximator over geometric graphs.

Is there something more powerful we can do?

2.3. Message-passing force fields

Message-passing (MP) graph neural networks (GNNs) have been the unifying architecture of all state-of-the-art, linearly scaling MLFFs since 2020.[10,85,92–96] Below, I introduce the key components of MP-GNNs, provide a (brief) history of their use as MLFFs, and train a series of increasingly sophisticated models to more accurately model the PES of bulk carbon.

In some sense, message-passing MLFFs generalise the above EAM model by stacking multiple "layers" of neighbour-summation to learn richer representations of an atom's local environment. These representations, otherwise known as "node features" or "embeddings", are tensors: structured collections of numbers that "describe" the local environment of an atom. The representations of a well trained model are similar for atoms in similar environments, and (very) different for atoms in (very) different environments.

A single message-passing operation updates node features, $\rho_i^{(l)} \to \rho_i^{(l+1)}$, via a three step process:

message, $m_{j\to i}^{(l)}$

First, a "message" is generated for each connected pair of nodes that sends information about neighbour j to node i:

$$m_{j\to i}^{(l)} = f\left(\vec{r}_{ij}, \rho_i^{(l-1)}, \rho_j^{(l-1)}\right) \cdot \phi_c(r_{ij})$$

This message is usually, but not always, conditioned on i's own features or identity in order to extract more relevant information from j. Multiplication by an envelope function ensures the message is smooth with respect to atomic positions, and in particular with respect to atoms entering and leaving the local neighbourhood.

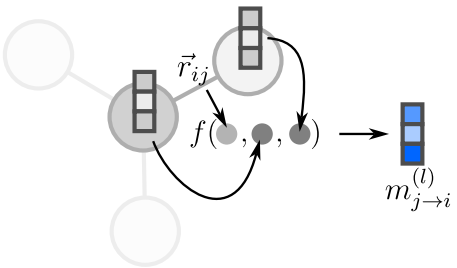


Figure 2.10: *message generation*

total message, $M_i^{(l)}$:

Next, for each node i, all messages from its neighbours are aggregated into a "total message" that summarises information about i's local neighbourhood:

$$M_i^{(l)} = \bigoplus_{j \in \mathcal{N}_i} m_{j \to i}^{(l)}$$

To ensure permutation invariance of the model, this accumulation must itself be a permutation invariant operation, most typically incorporating an (unnormalised) summation.

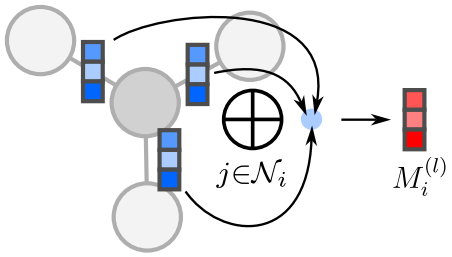


Figure 2.11: *total message accumulation*

update, $\rho_i^{(l)}$:

Finally, the node features of atom i are updated to incorporate information from the total message:

$$\rho_i^{(l)} = f\left(M_i^{(l)}, \rho_i^{(l-1)}\right)$$

Often this update contains a residual connection[67] to stabilise training, and create a natural hierarchy of features.

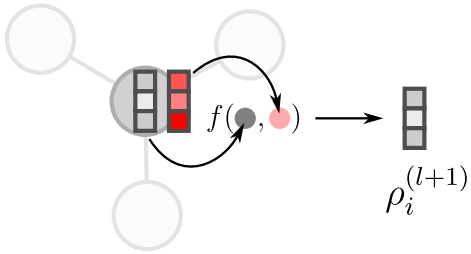


Figure 2.12: *node feature update*

In direct analogy with NNs, message passing MLFFs stack multiple message-passing layers to generate local energy predictions:

$$\begin{aligned}
\rho_i^{(0)} &= f(z_i) \\
\rho_i^{(l)} &= \mathrm{MP}^l\left(\rho_i^{(l-1)}, \left\{\left(\vec{r}_{ij}, \rho_j^{(l-1)}\right) \mid j \in \mathcal{N}_i\right\}\right) \quad l = 1, ..., L \\
\varepsilon_i &= f'\left(\rho_i^{(L)}\right)
\end{aligned}$$

Initial representations are generated purely as a function of each atom's identity. Subsequent message passing steps iteratively update these representations with geometric information about the atom's local environment. As a final step, the local energy is "read-out" via some final, learnable function (typically an MLP).

In each layer, the message creation, aggregation and update functions contain MLP components, such that the entire model can be trained end-to-end via gradient

descent. All message-passing operations contain highly parallelisable operations that scale linearly with the number of atoms in the system.

2.3.1. The search for a perfect architecture

The first MLFF architecture to incorporate message passing was `SchNet`[8] in 2017. While it improved numerically over existing descriptor-based models on various benchmarks (including `QM9`[97] and `MD17`[98]), widespread adoption was hampered by its lack of robustness in MD simulation, and its provably incomplete nature: pairs of non-unique structures can be constructed for which any `SchNet` model will always make identical predictions.[99]

Since then, researchers have instigated a wide-ranging search over the design space of possible message passing architectures,[100] *viz.* by changing the form of the message creation, aggregation and update functions, and the kinds of information stored in the node features. Indeed, between 2018 and 2023, more than 30 different message-passing MLFF architectures were published and benchmarked.[101] Throughout this search for the "perfect" architecture (a search that is still ongoing), two important model properties have emerged, and been optimised for:

- **expressivity**: whereas early models were provably incapable of approximating arbitrary functions defined over geometric graphs, modern models are capable of doing so. Finding such architectures, and demonstrating their completeness empirically and theoretically, was a major focus between 2019-22.[99]
- **suitable priors**: while the incorporation of directional information leads to message passing architectures that are provably complete,[100,102] the manner in which this is done affects model speed, data efficiency, and robustness. Since 2022, a major focus has been on optimising architectural components so that the overall model has the correct inductive biases for modelling PESs, *i.e.* finding models that can learn reference PESs with small amounts of data while remaining cheap at inference.

While a full analysis of this search is beyond the scope of this thesis, a dominant theme worth highlighting is the move towards increasing **internal equivariance** of models, and its contribution to improved model accuracy and robustness.

`SchNet`[8] (2017) (and others, including `DimeNet`[92] and `GemNet`[92]) generate internal representations, $\rho_i^{(l)}$, that are **invariant** with respect to rigid rotations of the total structure. `PaiNN`[93] (2019) was the first MP-MLFF architecture to incorporate internal equivariance, extending the `SchNet` architecture to also generate equivariant representations, $\rho_i^{(l)} \in \mathbb{R}^{d\times 3}$, such that rigid rotations of the total structure by some rotation matrix $W \in \mathrm{SO}(3)$ lead to identical rotations of the internal representations. Beyond vector representations, `TensorNet`[95] (2023) also generates internal, equivariant representations of order $l = 2$, *i.e.* collections of matrices: $\rho_i^{(l)} \in \mathbb{R}^{d\times 3\times 3}$.

Simultaneous to the development of these cartesian–equivariant models, the `NequIP`[94] (2021), `Allegro`[103] (2022) and `MACE`[10] (2023) architectures developed spherical tensor decompositions,[100] allowing for the use of arbitrary orders of equivariance within the internal representations of a single model.

Empirically, models with higher degrees of internal equivariance (*i.e.* higher l) are more expressive, more robust in MD simulations, and more data efficient.[94,104] As a demonstration, in Figure 2.13 I compare `SchNet` (invariant), `PaiNN` ($l = 1$ equivariance), and `TensorNet` ($l = 2$ equivariance) models trained on the same train/test split of the `C-GAP-17` dataset, and in the same manner as described above. To ensure a fair comparison, I use the recommended number of layers for each architecture (5 for `SchNet` and `PaiNN`, 2 for `TensorNet`), and alter each model's internal dimension so that each model has 10^5 learnable parameters.

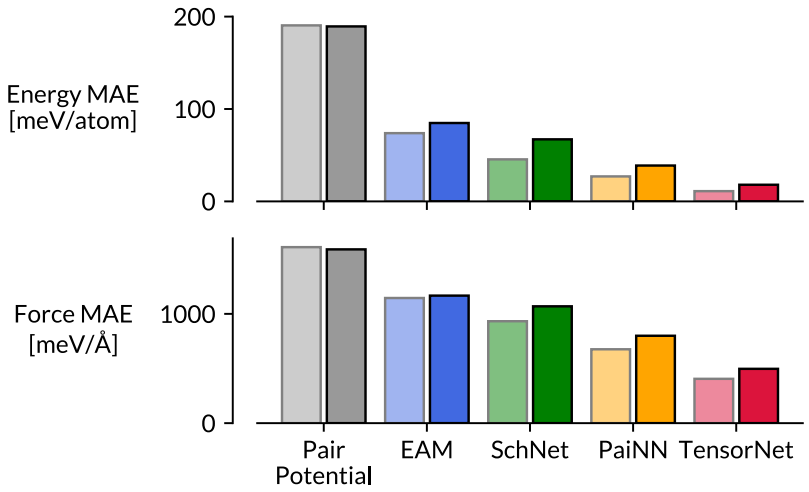


Figure 2.13: *train- (light) and test-set (dark) energy (top) and force (bottom) MAEs for a range of different architectures (colours) trained and tested on disjoint subsets of 400 randomly sampled bulk structures from the `C-GAP-17` dataset*

From left to right, we can see a clear improvement in accuracy (quantified by both energy and force MAEs) on both training and test sets as the degree of internal equivariance is increased. In all cases, early stopping was used to retro-actively select the best performing model after training had completed. In Figure 2.14, I plot the complete training and validation loss curves as a function of the number of gradient descent steps taken during training.

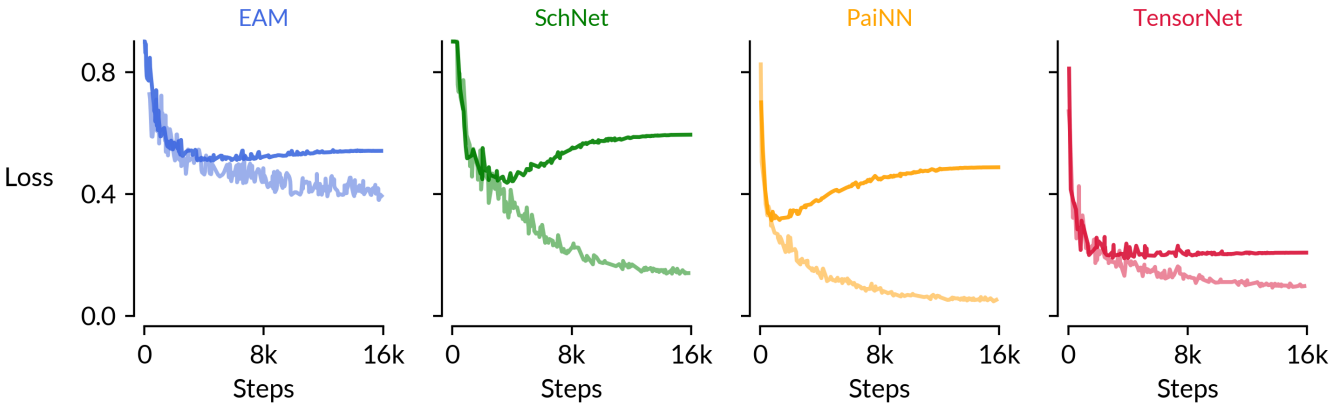


Figure 2.14: *training (light) and validation set (dark) loss evaluations as a function of the number of gradient descent steps taken during training for four different models, each with 10^5 parameters.*

From left to right:

- the `EAM` model dramatically under-fits the training data, clearly demonstrating its lack of expressivity. This naturally leads to a high validation loss.
- the `SchNet` model is more expressive, but not perfectly so. Therefore, the training loss approaches, but does not reach, zero. **Overfitting** is clearly visible: as the model continues to improve its fit to the training data, the validation loss gets worse, demonstrating a clear conflict between the model's inductive biases and the function it is trying to learn.
- the `PaiNN` model is more expressive again, and reaches near perfect training loss. Overfitting is still present, but occurs at a later onset during training, demonstrating closer inductive bias alignment. The validation loss greatly improves over the previous models.
- the `TensorNet` model is similarly expressive, and also reaches near-zero training loss. However, rather than dramatic overfitting, the validation loss instead plateaus at some low value, indicating a good inductive bias alignment with the reference PES.

To visually demonstrate the efficacy of message-passing MLFFs, I generate a parity plot for the `TensorNet` model in Figure 2.15, and invite the reader to compare to the pair potential (Figure 2.8) and EAM (Figure 2.9) models above. The trained model performs very well on the test set, with energy MAE of 18 meV/atom and force MAE of 0.49 eV/Å.

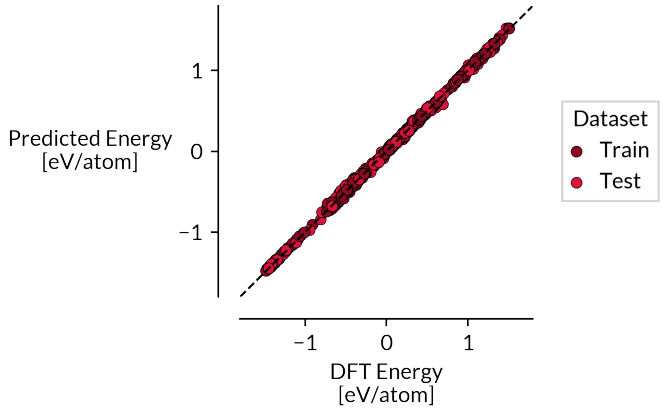


Figure 2.15: *parity plot for a* `TensorNet` *model trained on a 400-structure subset of the* `C-GAP-17` *dataset*

2.3.2. Other architectural trends

Beyond increasing internal equivariance, several other architectural features have been explored, and are worth mentioning in passing:

- **normalisation layers**: small architectural details can dramatically improve deep-learning model performance. The adoption of normalisation layers have been very beneficial in the context of computer vision,[66,105] and language models.[106,107] Recent inclusion of these layers in MLFF architectures (`LayerNorm`[65] in `TensorNet`[95], `RMSNorm`[106] in `Orb`[86]) improves training stability and final accuracy.
- **many-bodied messages**: beyond pairwise messages, $m_{j\to i}$, some architectures (`DimeNet`[92], `M3GNet`[108]) generate three-body messages, $m_{(k,j)\to i}$ to explicitly extract and act on directional information (*i.e.* bond angles). Beyond this, `GemNet`[92] generates four-body messages to also act on torsional information (*i.e.* dihedral angles). In general, the number of explicit summations over n-bodies scales exponentially with n, meaning improved accuracy is accompanied by a dramatic increase in computational cost. To mitigate this, `VisNet`[96] and `QuinNet`[102] use carefully constructed dot products over vector representations to implicitly extract angular information from sums over two-body messages. Finally, `MACE`[10] relies on the kernel trick employed by `ACE` descriptors[109] to extract messages of arbitrarily high order via iterated tensor product contractions from sums over two-body messages.
- **attention**: the attention mechanism of transformers[110] allows for efficient learning over (unordered) sets of objects, and has revolutionised the field of language modelling in recent years.[111] Recently proposed architectures (`TorchMD-NET`[112], `Equiformer`[113], `Orb`[86]) have incorporated this mechanism as a more expressive form of message aggregation.

2.4. Limitations to perfect accuracy

Even when training state-of-the-art architectures on large datasets, MLFF models still exhibit non-zero error on external test sets. Why is this? Several factors contribute:

- **locality assumptions**: most MLFFs rely on a finite radial cutoff (c), rendering them inherently incapable of explicitly modelling long-range interactions, (*e.g.* London dispersion forces, macroscopic electrostatic effects). While message-passing frameworks theoretically extend a model's receptive field — where the maximum interaction distance is $L \times c$ for L layers — the phenomenon of over-squashing severely limits the practical range of information propagation in GNNs.[114]
- **model capacity**: neural networks empirically require ≈1 parameter per unique bit of training data to achieve perfect accuracy on training sets.[115] Modern datasets contain as many as 100M structures, encompassing 1B+ unique atomic environments.[51–53] While the exact information density of such datasets has yet to be quantified, models with significantly fewer parameters than atomic environments are empirically incapable of perfectly modelling their training data (see Figure 2.16 below), even when using expressive architectures. This impaired training performance directly leads to impaired test set performance.
- **out-of-distribution (OOD) generalisation**: MLFFs are strong interpolative models, but often fail to extrapolate well: when applied to structures far outside the "convex hull" of the training data, such as extreme temperatures or high-pressure phases not present in the reference set, most MLFF models' predictive reliability degrades significantly, and in proportion to the degree of the OOD-nature of test structures.

The magnitude of error that these factors contribute is difficult to quantify, and depends on the specific architecture, training dataset and parameter regime. While some pose insurmountable challenges, model error can often still be systematically improved by increasing model size and/or amount of training data.

2.5. Scaling laws

In many ML applications and settings, model performance on some task is found to scale predictably with both (i) the size of the model (*i.e.* the number of parameters), and (ii) the amount of training data. Most notably, the scaling laws for LLMs observed by *Kaplan et al.*,[116] and subsequent Chinchilla re-formulations,[117] are perhaps the single most important driving force behind the massive scale up in compute and data used to train LLMs over the last 5 years.

MLFF models are also subject to scaling laws.[94,118] Across a wide range of architectures, datasets, and training regimes, it is empirically observed that model error scales as a power law of the amount of training data:

$$\varepsilon = a \cdot |\mathcal{D}|^{-b}$$

where the exponent b typically depends only on the training dataset and model architecture. For a fixed amount of training data, MLFF error typically also decreases as model size increases, but in a less predictable manner.

To demonstrate, in Figure 2.16 I plot the force RMSE for `TensorNet` models of increasing size, trained on increasing amounts of the `C-GAP-17` dataset, and evaluated on a hold-out test set. This is the first of many such "learning curves" presented in this thesis.

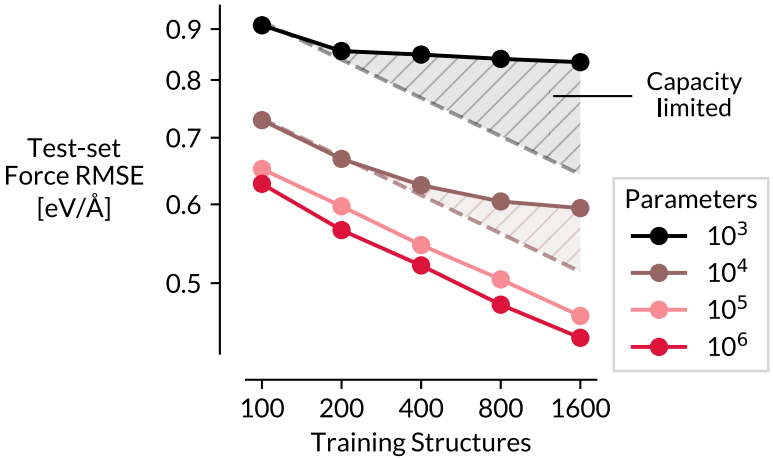


Figure 2.16: *force RMSE for `TensorNet` models of increasing size, trained on increasing amounts of the `C-GAP-17` dataset, and evaluated on a hold-out test set.*

In all instances where model size (measured in number of parameters) is greater than the amount of environments in the training dataset, model error falls on predictable power law scalings (straight lines on a log-log plot). In cases where the model is capacity–limited (highlighted via shading), errors plateau, and the model is unable to benefit from further amounts of training data.

While this simple demonstration only extends to 1,600 structures, empirical scaling laws have been observed to hold for as many as 100M structures, and to billion parameter models.[12]

2.6. Validating machine-learned force fields

The primary purpose of MLFFs is to extend the application of atomistic simulations into regimes that are inaccessible to *ab initio* methods, for instance high throughput searches over large chemical spaces, and dynamic simulations over large time- and length-scales. A fundamental question immediately arises: how can we rely on the outcomes of such MLFF-driven simulations when no direct comparison can be made to the *ab initio* ground truth? **Validating** MLFFs is the non-trivial process of

designing and applying a range of tests to build trust in an MLFF's predictions when extrapolated into these regimes.

Above, I have so far assessed MLFF accuracy by evaluating models on hold-out test sets of structures not seen during training. Such evaluations are useful: poor performance (*e.g.* low R^2 correlation with *ab initio* labels) is a clear indication that an MLFF is not reliable and provides concrete examples of structures for which the MLFF makes bad predictions. However, while strong test-set performance is necessary for a model to be reliable, it is not sufficient. Concretely:

1. **coverage**: test sets rarely span the entirety of relevant chemical space, and are usually sampled from the same distribution as the training data. This can lead to over-confidence in the model's performance: regions of chemical space may exist where the model makes bad predictions, but that are not represented in the test set, and hence not detected.
2. **averaging**: by far the most common practice is to report *mean* test-set error metrics, such as RMSE or MAE on energy, force and stress labels. This averaging can act to obscure small regions of chemical space where the model makes bad predictions, and also to bias the error values towards regions of chemical space that are most well-represented in the test set.
3. **interpretability**: all MLFFs exhibit some degree of error. An error metric in isolation is hard to interpret. For instance, an energy MAE of 10 meV/atom would represent shockingly bad performance for a copper PES evaluated on bulk solid structures,[119] but would be perfectly acceptable for a carbon PES evaluated on bulk amorphous structures.[58]

Modifications to the test-set evaluation procedure can help to mitigate some of these issues. In particular, stratified test splits can be used to separately evaluate MLFF performance on different regions of chemical space, while robust statistics, such as the median or maximum error values, can be reported to better capture the *distribution* of errors.

Ultimately, however, these limitations motivate validation strategies that go beyond static error analysis. In our paper:[54]

"How to validate machine-learned interatomic potentials"
J.D. Morrow, John L.A. Gardner, V.L Deringer
The Journal of Chemical Physics, 2023, Volume 158, Issue 12

we propose "physical validation" as the best way to build trust in an MLFF's behaviour and reliability. Such an approach involves constructing model evaluations that measure *properties* of the PES that the model induces, and that can be compared directly to *ab initio* and experimental ground truths.

In a materials context, the simplest such physical validation procedure is to calculate the equilibrium lattice constants for known crystal structures. As an example, in

Figure 2.17 I use the best performing `TensorNet` model from above (1M parameters, trained on 1,600 `C-GAP-17` structures) to scan over a range of scaled diamond (LHS) and graphite (RHS) structures, measuring the total energy as a function of the relevant lattice constant. Comparison to both the *ab initio* labelling method and to experiment reveals good agreement for this model (see Table 2.1).

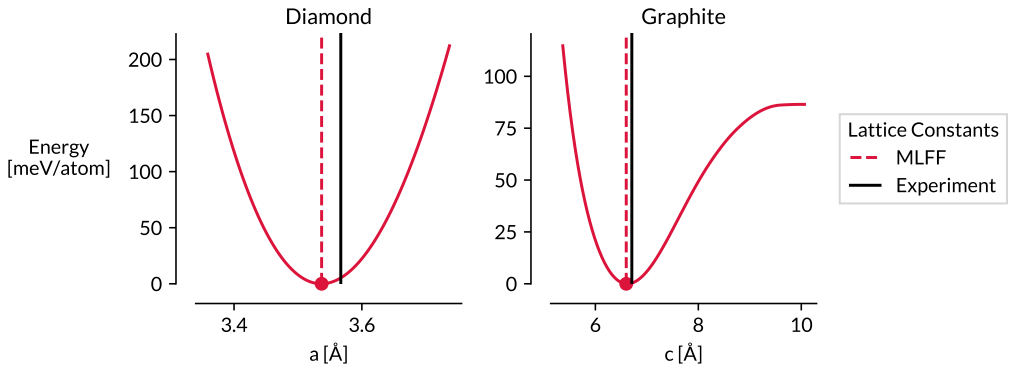


Figure 2.17: *scanning over lattice constants to find the equilibrium value of (LHS) the lattice constant for cubic diamond, using an isotropic expansion of the unit cell, and (RHS) the c lattice constant for graphite, by expanding along the z-axis only.*

Table 2.1: *equilibrium lattice constants for cubic diamond and graphite, calculated using the best performing* `TensorNet` *model from above, and compared to LDA-DFT*[58] *and experiment.*

Lattice constant	MLFF	LDA-DFT	Experiment
diamond, a [Å]	3.537	3.532[58]	3.567[120]
graphite, c [Å]	6.610	6.625[58]	6.708[121]

Other experimental observables can also be calculated in a lightweight and automated fashion, including *e.g.* phonon dispersion relations,[58,122–124] elastic constants,[125–127] and the relative stabilities of different phases.[72,73,128] Even simple validation procedures such as these provide useful indicators of MLFF reliability: models are never explicitly trained to reproduce these properties, and small changes in model predictions can lead to large deviations from the ground truth. Strong performance on these tasks is therefore indicative that the model has captured the underlying physics and subtleties of the PES around relevant equilibrium structures.

Beyond these static tests, the most stringent evaluations of MLFF reliability are those that involve integrating model predictions through time,[55,129,130] most typically via the application of Molecular Dynamics (MD).[131–133] All MLFFs exhibit non-zero error;

MD validation seeks to test whether these errors accumulate during time integration, leading to systematically unphysical behaviour, or whether they act as small perturbative noise with no harmful effects. In order to "pass" MD-based validations, MLFFs must surmount a hierarchy of challenges:

1. **stability**: the first and most basic challenge is to ensure that the MLFF can drive "stable" MD simulations. MD simulations use MLFFs to numerically integrate Newton's equations of motion. Rapidly changing and very large forces can lead to numerical instability, "pumping" energy into the system, and even causing the explosion of unit cells and the loss of atoms from the system.
2. **physicality**: a simulation being stable does not preclude the possibility of it visiting unphysical structures. At a high level of abstraction, MD is an (efficient) sampler of the Boltzmann distribution. If an MLFF model contains "holes" within its PES (*i.e.* predicts unrealistically low energies) in certain regions of chemical space, these structures will act as "attractors" for the MD sampler, leading to simulations becoming trapped in these regions. Most commonly, holes appear at unphysically small interatomic separations, or for structures containing large vacuum regions, or isolated atoms.
3. **observables**: if an MLFF can drive both stable and physically plausible MD simulations, we can extract a range of physical observables, and use these to compare to experiment. Each observable typically probes some particular property of the system, and can often be very sensitive to small changes in model predictions. By comparing a suite of these observables to experiment, we can build a comprehensive picture of the model's reliability and trustworthiness.

Commonly used observables for MD-based validation include radial distribution functions (which can be extracted from experimental structure factors), diffusion constants, and equilibrium densities.[58,134] Inspection of the structures visited over the simulation can also provide useful information about the model's behaviour, for instance by counting the distribution of n-fold coordinated atoms, rings of certain sizes, co-locations of defects, *etc.*[18]

Finally, beyond generally applicable MD validations, expert domain knowledge can be used to construct protocols that probe specific phenomena of interest. As an example, one literature-standard test for a carbon PES is to drive a melt-quench-anneal simulation of a bulk structure at 2 g/cm^3: under these conditions, it is known that well behaved PES models should recover the graphitic-type structures that are experimentally observed at this density.[58,72]

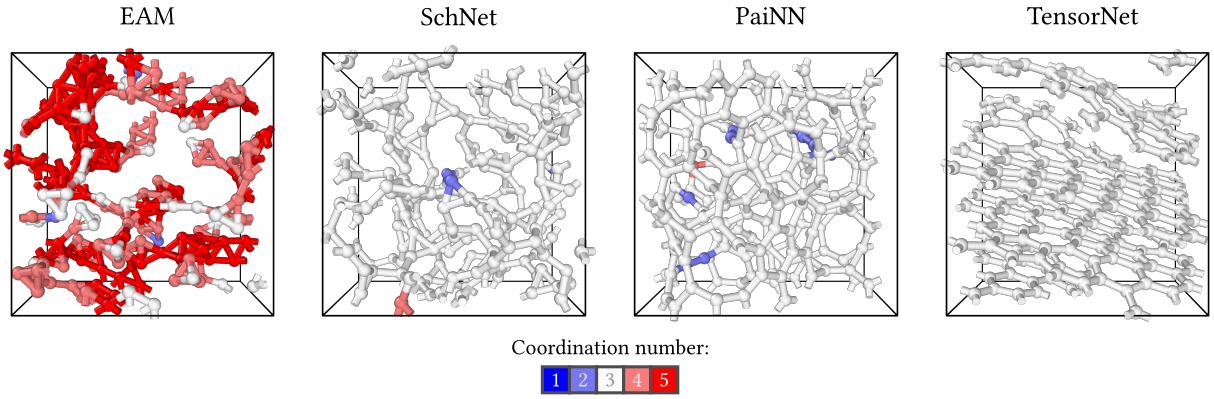


Figure 2.18: *final structures generated by melt-quench-anneal simulations of bulk carbon at 2 g/cm³, as driven by various MLFF models. Each atom is coloured according to the number of neighbouring atoms within a radius of 1.65 Å. Structural renders generated using* `Ovito`.[135]

As a demonstration, in Figure 2.18, I compare the final structures from MD trajectories driven by various MLFF models — `EAM`, `SchNet`, `PaiNN` and `TensorNet` — all of which use 1M parameters, and have been trained on 1,600 structures from the `C-GAP-17` dataset as described above. The same melt-quench-anneal protocol was used for all simulations: a 5 ps melt at 9,000 K, followed by a 5 ps quench to 3,000 K, a 100 ps anneal, and finally a 50 ps quench to 300 K.

The quality and correctness of the final structures generated by each model correlate with the model's test-set performance. From left to right, structures improve from completely unphysical to qualitatively correct:

- the `EAM` model generates an unphysical structure, with large regions of vacuum and highly coordinated carbon clusters.
- the `SchNet` model generates mostly 3-fold coordinated atoms (correct), but the geometry of these environments is strained, forming unphysical chains, and very tight bond angles.
- the `PaiNN` model generates broadly correct local environments (i.e. mostly "flat", sp^2 carbon atoms), but mid-range order, and in particular the relative orientation of local environments, is incorrect, leading to an extended 3d network rather than planar sheets
- the `TensorNet` model generates physically correct graphitic sheets, with no coordination defects

Ultimately, MLFF validation involves a combination of approaches, probing both the static and dynamic behaviour of the model. As MLFFs are increasingly relied upon for scientific applications, the importance of rigorous and comprehensive validation pipelines will only continue to grow.

2.7. graph-pes

Historically, new families of MLFF architectures have been developed and released as standalone software packages.[i] While each of these packages has advanced the state of the art in at least one dimension, the lack of a unified interface has led to a fragmented ecosystem, with little interoperability between packages. For MLFF practitioners, and the field as a whole, this has several undesirable consequences:

- **slow adoption**: migrating to new packages to try new architectures can be slow and painful, requiring re-formatting of datasets, re-optimisation of hyper-parameters, and more.
- **difficult direct comparisons**: each package implements its training procedures in different ways. Subtle changes in loss function constructions, optimisation processes, learning rate schedules and more can lead to dramatically different model performance for a given dataset. This makes it difficult to directly isolate and compare model architectures on common benchmarks.
- **varied downstream support**: different MLFF softwares typically support different subsets of downstream MD drivers and other simulation packages. This can make it difficult, if not impossible, to run identical simulations with different MLFFs.
- **slow development**: re-implementing functionality and training pipelines for each new architecture is time-consuming and error-prone.

These challenges collectively hinder reproducibility, extensibility, and interoperability in MLFF research and application. In an attempt to address these issues, I present `graph-pes`,[57] a software package for defining, training and deploying MLFF models with a unified, architecture agnostic interface.

At its heart, `graph-pes` provides two fundamental abstractions for (i) representing data and (ii) defining models that act on it:

`AtomicGraph` objects represent chemical structures, using collections of `PyTorch`[59] tensors to store atomic positions, atomic numbers, unit cell vectors, edge connectivity, and PES labels. I provide a wide range of methods to calculate properties of such graphs (such as the number of atoms, the number of incident edges per atom, vectors along each edge *etc.*), and to perform common operations upon them (such as indexing and summations over relevant neighbours, edge trimming, edge-weighted attention, *etc.*). All of these functions can act on both single graphs, as well as batched collections of structures — this allows for efficient parallelisation over multiple structures, while maintaining high degrees of readability and composability. Internally, batching is

[i] for example, `schnetpack`[136,137] provides `SchNet`[8] and `PaiNN`,[93] `nequip`[138] provides `NequIP`,[94] `mace-torch` provides `MACE`,[10] `fairchem`[139] provides `ESEN`,[85] `torch-md`[140] provides `TorchMD-NET`[140] and `TensorNet`,[95] *etc.*

implemented by creating a single `AtomicGraph` object that contains multiple disjoint sub-graphs.

The `GraphPESModel` base class defines a common interface for all MLFF models that generate PES predictions by acting on graph representations of chemical structures (*i.e.* `AtomicGraph` objects). Assuming that implementations produce local energy predictions, or a scalar total energy, this base class provides mechanisms for automatically calculating forces and stress tensors from energy predictions in a conservative manner. Specifically, for force predictions this is achieved by inserting a zero-initialised atomic displacement vector, $D = 0^{N\times 3}$, into the computational graph as $R \to R + D$, and then using `torch.autograd.grad`[59] to calculate the gradient of the model's total energy prediction with respect to D, and hence with respect to the atomic positions. Similarly, for stress predictions, a zero-initialised unit cell distortion tensor, $\sigma = 0^{3\times 3}$, is inserted into the computational graph as $C \to C(I^{3\times 3} + \sigma)$, and the gradient of the model's total energy prediction with respect to σ is calculated. I provide independent implementations of many architectures, including both basic PES models, such as `PairPotentials`, `ZBLCoreRepulsions`,[141] `StillingerWeber`,[142] *etc.*, as well as many message-passing architectures, including `SchNet`,[8] `PaiNN`,[93] `TensorNet`,[95] `NequIP`,[94] `MACE`,[10] and `Orb`.[86,143] Beyond these re-implementations, I also provide wrapper classes so that models from other software packages can be used as `GraphPESModels`. In particular, this allows for the use and fine-tuning of foundation models such as `orb-v2`[86] and `orb-v3`[143] from `orb-models`, `MACE-MP0`,[11] `MACE-OFF`[144] and `GO-MACE`[145] from `mace-torch`, and `MatterSim`[146] from `mattersim` (see Chapter 4).

Alongside these core abstractions and concrete implementations, I provide training scripts, model visualisation tools, and MD-driver integrations (including to `ASE`,[147] `LAMMPS`[148] and `torch-sim`[149]), all of which are designed to be *architecture agnostic*, and hence to work with all models implemented in `graph-pes`. These provide several advantages over existing tools:

- **reduced barrier to entry**: to investigate a new architecture, researchers need only implement the forward pass: `graph-pes` provides all other functionality to train, evaluate and deploy models. I have made use of this above: the `EAM` architecture took 10 minutes, and 20 lines of code to implement, after which I could use it like any other model in `graph-pes`.
- **feature sharing**: many useful training features and procedures are tied to a single model architecture in existing software packages. `graph-pes` allows for the use of arbitrary features with arbitrary architectures, including *e.g.* stochastic weight averaging,[68] frozen transfer learning,[150] arbitrary learning rate schedulers, custom loss functions, *etc.*
- **reproducibility**: by changing a single line of configuration, practitioners can swap out model architectures and models in training and evaluation scripts, allowing for reproducible and direct comparisons of model performance across architectures, datasets, and application domains.

Since its release, `graph-pes` has started to drive new research directions and applications. So far, we have used `graph-pes` to fit `NequIP` models to datasets created using `autoplex`,[151] to assess zero-shot performance of different MLFFs,[152] and to distil atomistic foundation models (see Chapter 4).[153]

Attribution and Acknowledgements

All experiments presented in the above chapter were performed solely by me, John Gardner, and are new results prepared solely for this thesis.

My contributions to "*How to validate machine-learned interatomic potentials*"[54] include discussion of various error metrics, application of stratified and targeted test sets, and the use of robust statistics and measures of error distribution, all as discussed above. In collaboration with J.D. Morrow and V.L. Deringer, I also helped to shape the discussion and promotion of physical validation as an important and necessary step in the development of MLFFs.

I developed the core framework of `graph-pes` between October 2023 and December 2024. I am grateful for useful discussions with Dr. Daniel Thomas du Toit and Rokas Elijošius, and for the early testing and feedback of Dr. Zoé Faure Beaulieu, Krystian Gierczak, Natascia Fragapane, and Dr. Zakariya El-Machachi.

3. Synthetic pre-training for machine-learned force fields

In the previous chapter, I explored how to train an MLFF model on a dataset of structures that have been labelled using a single *ab initio* method. As I demonstrated, and has been well established in the literature,[12,94,118] the accuracy of models trained in this way improves systematically with the size of the training set, typically according to some **power law**:

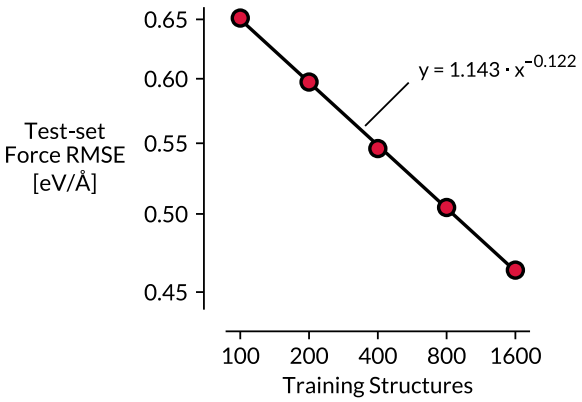


Figure 3.1: *empirical power scaling law for the accuracy of a 10^5 parameter* `TensorNet` *model trained on increasingly large subsets of the* `C-GAP-17` *dataset, and evaluated on a hold-out test set.*

Such scaling laws are both a blessing and a curse. On the one hand, an obvious and clear path to improving model accuracy is always available to us: use more compute to label more data, and use this to train a better model. On the other hand, the amount of new data required to improve model accuracy by some fixed amount grows **exponentially** with the amount of data already available. Since *ab initio* calculations are expensive, this means that (by far) the largest cost in training an MLFF from scratch is the labelling of the training data.

Beyond direct training on single datasets, smarter training strategies that improve the **data efficiency**[i] of MLFF training are therefore of very real practical importance.

[i] *i.e.* generate more accurate models for the same amount of data, or reach the same accuracy while requiring less data

A common approach in established machine learning applications is to first "pre-train" a model on some related task or dataset. During this training phase, deep-learning-based models learn meaningful representations of data. Providing that these representations are relevant to the target task, subsequent "fine-tuning" requires much less data to achieve the same level of performance compared to training from scratch. Two examples of where this approach is essential are:

1. **modern large language models (LLMs)**: pre-training to predict the next token in samples drawn from large corpora of text equips "base models" with robust and nuanced representations of textual data, leading to extremely general language modelling abilities. Without this large-scale pre-training, subsequent fine-tuning (via reinforcement learning from both human feedback[154] and verifiable rewards[155,156]) to produce intelligent and useful final models would be far too expensive to be feasible.
2. **self-driving cars**: modern, deep-learning based models for autonomous driving are all pre-trained on massive datasets generated by computer simulations, before then being fine-tuned on comparatively scarce real-world sensor data.[157]

Crucially, in both of these examples, the data from the target task (human feedback, expensive model rollouts, real-world car journeys) is much more expensive to collect relative to the pre-training data (the internet, game-engine generated simulations). Large-scale pre-training can therefore be a cost-effective means to achieve high-accuracy final models.

Inspired by these approaches, in this chapter I explore the use of **synthetic** data as a source of pre-training data for MLFFs. Concretely, I use a cheap, approximate method (here an existing MLFF model) to (i) generate and (ii) label a large dataset of atomic structures. Pre-training on this synthetic data improves final model accuracy after fine-tuning on the target, *ab initio* dataset.

I begin by generating a synthetic dataset for bulk amorphous carbon, using an existing MLFF model to drive many uncorrelated MD simulations. I go on to show that the local energies of these synthetic structures can be well modelled by simple local energy models, and that **pre-training on synthetic data improves model accuracy** after fine-tuning on the *ab initio* dataset. Finally, beyond "just" numerical error improvements, I show that pre-training on synthetic data can also improve the **robustness** of the MLFF model during molecular dynamics simulations.

3.1. A synthetic dataset for bulk amorphous carbon

In this chapter, I focus on modelling bulk amorphous carbon. While consisting of only a single element, this domain spans many varied sub-systems, including graphite, diamond, fullerenes, tetrahedral amorphous carbon, and other interesting carbon

polymorphs. This system therefore acts as a stringent and useful test-bed for MLFF models.[58,72,158]

To generate a synthetic dataset that attempts to cover the entire configurational space of this system, I make use of an existing, and carefully validated MLFF model (`C-GAP-17`[58]), to drive MD simulations in the NVT ensemble, as implemented in `LAMMPS`.[148] The final dataset comprises equally spaced snapshots along multiple, uncorrelated MD trajectories, each generated according to the following, three-phase protocol (see also Figure 3.2):

1. MELT
   an initial, random configuration of density ρ g/cm^3, generated by randomly placing hard-spheres with a radius of 0.5 Å in a cubic box of appropriate volume, is heated at 9000 K for 5 ps – ① in Figure 3.2
2. QUENCH
   the system is cooled to 300 K over 5 ps, effectively relaxing the structure into a high-energy amorphous state – ② in Figure 3.2
3. ANNEAL
   the system is heated back up to T K over 50 ps, held at constant temperature for 100 ps, then cooled back down to 300 K over 50 ps. This gradually orders the structure, eventually leading to low-energy, semi-ordered structures – ④ in Figure 3.2

Similar protocols have been used elsewhere to generate bulk carbon structures.[159–162]

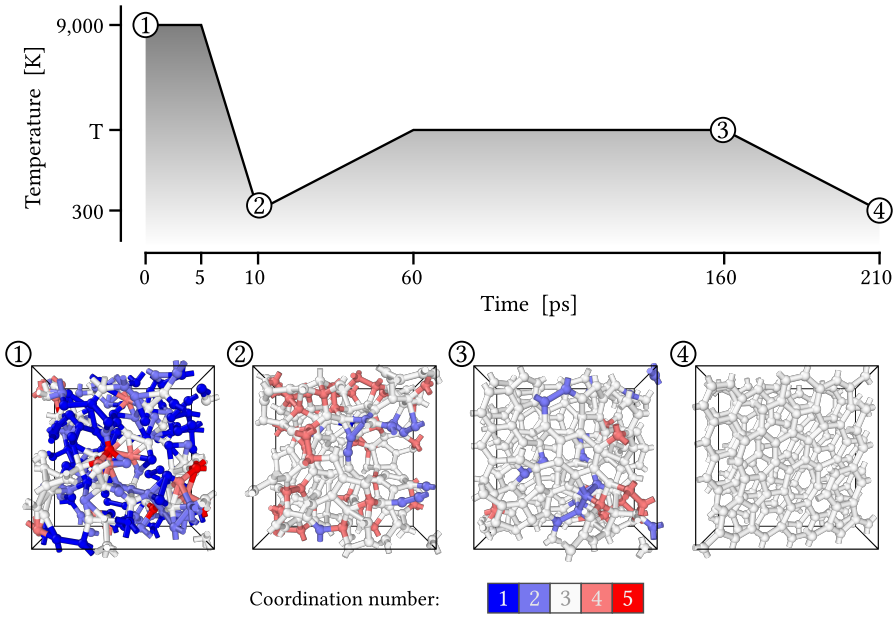


Figure 3.2: *protocol for generating the synthetic dataset for bulk carbon. Four structures from the ρ = 2.5 g/cm³, T = 3000 K trajectory are shown, with atoms coloured based on their local coordination number, defined as the number of neighbouring atoms within a radius of 1.65 Å. Structural renders generated using* `Ovito`.[135]

Each trajectory therefore samples a range of energies, temperatures and degrees of dis-/order. To generate a large number of diverse structures, I scan over a grid of densities and temperatures, with ρ ranging from 1.0 to 3.5 g/cm³ in increments of 0.1 g/cm³, and T ranging from 2,000 K to 4,000 K in increments of 100 K. I sub-sample each MD trajectory every 10 ps to arrive at a final dataset (`C-SYNTH-23M`) containing 115,206 structures, for a total of 23,041,200 unique atomic environments.

A selection of the final structures from each trajectory in the $[\rho, T]$ grid is shown in Figure 3.3. Increasing ρ facilitates the exploration from low-density structures (where sp and sp² bonding dominates) to high-density structures (where sp³ bonding dominates). Increasing T provides the system with more kinetic energy during the anneal phase, leading to quicker re-ordering, and hence more ordered, lower-energy final structures.

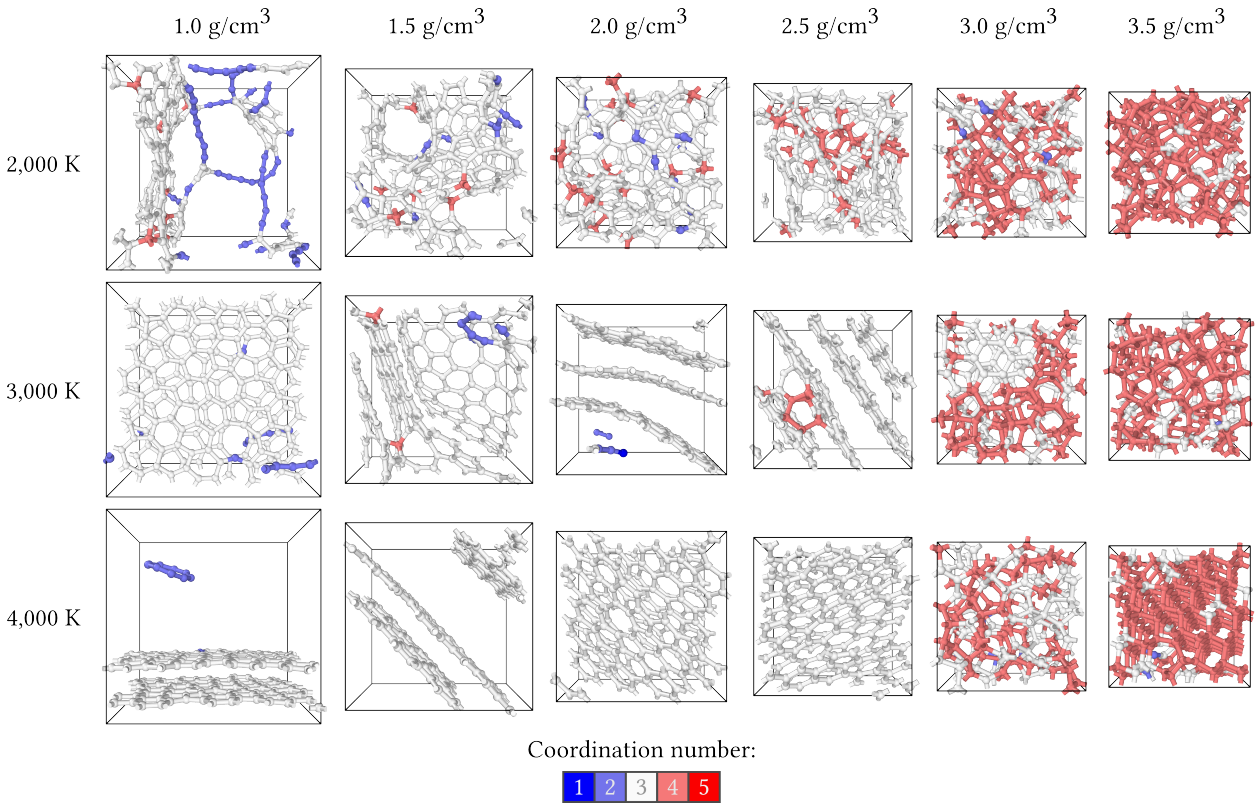


Figure 3.3: *map of the synthetic dataset for bulk carbon. Atoms are coloured based on their local coordination number, defined as the number of neighbouring atoms within a radius of 1.65 Å. Structural renders generated using* `Ovito`.[135]

3.2. Descriptor-based local energy models

Before pre-training and fine-tuning a complete MLFF model, I begin by studying the proposed pre-training task in the simplest possible manner: learning to predict each atom's synthetic local energy label solely as a function of its local environment.

Under the hood, the `C-GAP-17` model describes each atom's local environment using a Smooth Overlap of Atomic Positions (SOAP) descriptor (a fixed length vector of features that "describes" the local environment — see Appendix A).[7,163] This descriptor is then mapped through an adapted Gaussian Process Regression (GPR)[7,77] model to generate a local energy prediction for each atom. Below, I investigate the ability of simple models to reproduce this mapping by training on environments taken from the synthetic dataset.

To describe each environment, I use SOAP descriptors generated using a radial cutoff of 3.7 Å, a Gaussian smoothing width of $\sigma = 0.5$ Å, a maximum radial order of $n_{\text{max}} = 12$ and a maximum angular order of $l_{\text{max}} = 6$, as implemented in `quippy`.[164,165]

3.2.1. Three simple model classes

To compare and contrast the suitability of different model classes for this task, I consider the following (implementations for architectures and training pipelines are available online):[166]

Subset of Regressors Gaussian Process Regression (SoR-GPR)[14,77]

As discussed in Appendix B, SoR-GPR models are a sparse approximation to full-rank GPR models, and make predictions as a sum over kernel-weighted coefficients associated with a (small) set of M inducing points.

In keeping with the original motivation of SOAP vectors, I use a linear-power kernel function to measure the similarity between two environments:

$$k(x, x') = (x \cdot x')^{\zeta}$$

with $\zeta = 4$. I use the simple sparse point selection strategy of randomly sampling M points from the training set. Code is implemented in pure `NumPy`.[167]

Neural Networks (NN)

As discussed in Chapter 2, NNs are sequential models composed of alternating linear and non-linear transformations, and are provably capable of approximating any continuous function to arbitrary accuracy given sufficient parametrisation.

Here, I use a simple feed-forward MLP, consisting of L hidden layers, each of width h. The first layer projects the SOAP vector to a latent space of size h. The hidden layers (each of which applies an affine transformation followed by an element-wise non-linearity) update the latent representation. The final layer maps the final hidden representation to a single, scalar output (the local energy prediction), with no activation function applied.

To prevent over-fitting, I employ four common regularisation techniques:

1. **dropout**:[168] I randomly set each of the model's parameters to zero during each forward pass during training, with probability p. This prevents the model from over-fitting to the training data by forcing it to learn more robust and generalisable features.[169]
2. **weight decay**: NNs are often too expressive for the task at hand; in particular, large weights have been empirically associated with less-smooth models, and hence over-fitting.[170] I mitigate this by adding an additional term to the loss function that penalises the squared magnitude of the weights:

$$\mathcal{L}(\theta) = \lambda_{\text{WD}} \sum_i \theta_i^2$$

where λ_{WD} is a scalar hyper-parameter that controls the strength of the regularisation.

3. **early stopping**: it is often the case that NNs trained for many optimisation steps will start to over-fit to the specific training data points. Monitoring the loss on a hold-out validation set during training and stopping training when this loss ceases to improve for a given number of steps is a simple and effective way to prevent over-fitting.[171]

4. **feature standardisation**: training on features with different natural scales can be problematic when optimising NNs using gradient-based methods: the varying scales induce a wide range of gradient magnitudes, and act to skew the loss surface in heterogeneous ways, delaying convergence. In addition, weight initialisation strategies often assume that features are normally distributed.[64,172] Pre-processing the SOAP vector feature matrix so that each column has zero mean and unit variance is a simple and effective way to mitigate this issue.[173]

All models are implemented in pure `PyTorch`,[59] and trained using the `Adam` optimiser.[69]

Deep Kernel Learning (DKL)[174,175]

A common limitation of GPR models is the choice of kernel function: special structure may not be well modelled by a simple, stationary kernel function.

DKL models aim to combine the representational power of NNs with the data efficiency of GPR models by creating a composite, "learnable" kernel function:

$$k(x, x') = k'(h(x), h(x'))$$

where

- $h(x)$ is a NN that maps the input x to a latent space
- $k'(\cdot, \cdot)$ is some standard kernel function that acts on the latent space

In many regression and classification tasks, DKL has been shown to surpass both GPR and NN models.[174–177]

In this work, I explore the use of DKL models composed of feed-forward NNs and the radial basis function (RBF) kernel. I use the same architecture as the NN models above, without the final readout layer, and employ the same regularisation and optimisation techniques. Models are implemented in `GPyTorch`,[178] which provides a GPU-accelerated method to train combined NN and GPR models in a batched manner.

3.2.2. Model optimisation

Each of the above model classes has a variety of hyper-parameters (settings that are kept fixed within a given optimisation trajectory), each of which can be tuned to improve performance. To ensure a fair comparison between model classes, I optimise

the hyper-parameters for each model class for each size of training dataset, before comparing the best performing models.

To optimise and compare models, I employ a 5-fold cross validation procedure: I split the total synthetic dataset into 5 equally sized subsets, sampling in such a way that environments from a single trajectory are assigned to a single fold to avoid data leakage. As discussed in Chapter 2, splitting by trajectory rather than by individual structure is essential when working with MD-generated data, since consecutive snapshots along a trajectory are highly correlated; assigning entire trajectories to a single fold therefore ensures that the validation and test sets contain genuinely independent configurations. Deciding on the optimal hyper-parameters for each model class involves comparing the mean performance on the **validation sets**. Final performance is then evaluated as the mean over all 5 **test set** evaluations.

The SoR-GPR models I train have two hyper-parameters: the number of inducing points, M, and the regularisation strength, σ. The performance of SoR-GPR models strictly increases with the number of inducing points, and so I set $M = \min(|D|, 5000)$ for each training dataset size, where 5000 points corresponds to the maximum number of training points I could accessibly train (due to the poor scaling of memory required for GPR fits with increasing dataset size). I sample the regularisation strength, σ, using a log-linearly spaced grid of 100 values between 0.1 and 1000 meV (see Figure 3.5 below for more details).

The NN and DKL models have many more hyper-parameters to tune. An exhaustive grid search of all possibilities would be (far) too expensive. I instead adopt the common approach of using a random (Hammersley[179]) search over a pre-defined hyper-parameter space. Ranges and sampling strategies for the various hyper-parameters are given in Table 3.1 and Table 3.2. For each size of training dataset, I sample 100 hyper-parameter configurations.

Table 3.1: *hyper-parameter ranges and sampling strategies for training NN models*

Parameter	Values	Sampling
learning rate	$[10^{-4}, 10^{-3}]$	log
hidden layers	$[2, 7]$	linear
neurons per layer	$[10^2, 2 \cdot 10^4]$	log
activation	`CELU`[180], `RelU`, `tanh`	choice
lr decay	$[0.985, 1]$	log
early stopping patience	$[40, 400]$	log
weight decay	$[10^{-4}, 10^{-1}]$	log
dropout	$[0.0, 0.5]$	uniform

Table 3.2: *hyper-parameter ranges and sampling strategies specifically for training DKL models*

Parameter	Values	Sampling
latent dimension	[5, 500]	log
activate latent	Yes/No	choice
inducing points	[50, 4000]	log

3.2.3. Learning curves

To compare the performance of the different model classes, I train instances of each on increasing amounts of data points (SOAP vectors and corresponding local energies). In Figure 3.4, I plot the best mean test-set local energy MAE as averaged over 5-fold cross validation. To prevent data leakage between folds, I assign all the environments from a single trajectory to a single fold. This ensures that the model is evaluated on environments from structures that it has not seen during training.

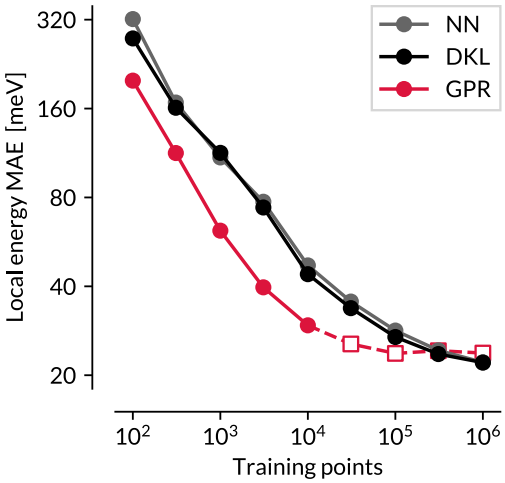


Figure 3.4: *number of training points vs test-set local energy MAE for optimised GPR, DKL and NN models, taken as a mean over 5-fold cross validation. The dashed line and square markers indicate models trained on specialist compute (requiring more than 64 GB RAM)*

In the low data regime, all models show log-log linear scaling laws, as expected. In this regime, GPR models are most accurate, thanks to their strong regularisation, Bayesian nature, and limited capacity (and hence lack of tendency to over-fit).

As the amount of data increases, the performance of all models begins to saturate. GPR models saturate at marginally higher errors (≈24 meV) than the NN and DKL models (≈22 meV).

In general, model saturation can be attributed to two possible causes:

1. **insufficient capacity**: in this regime, the model lacks sufficient flexibility or parametrisation even to fit to the training data: using even more data cannot overcome this limitation, and hence model performance plateaus.
2. **generalisation gap**: in this regime, the model is still able to fit its training data, but improved performance does not transfer to regions of the test-set that are truly out-of-distribution.

I attribute the saturation seen for all of these models to the second cause, a true generalisation gap, for the following reasons:

1. all models saturate at similar levels of accuracy, despite large differences in capacity: the SoR-GPR models only have 5k learnable coefficients, in contrast to the best NN and DKL models that have 1M+ parameters.
2. the ground truth function (the `C-GAP-17` mapping from SOAP vector to local energy) has bounded information content.
3. relaxing regularisation methods for all three methods leads to near perfect accuracy on the training data ($\ll$1 meV) for all three model classes.

3.2.4. Gaussian process regression ablations

Unlike *ab initio* labelled datasets, the synthetic dataset I have developed was cheap to generate, and hence is much larger than any other existing dataset for the atomistic modelling of bulk carbon.

This size, together with the lightweight nature of descriptor-based local energy models, allows for a thorough investigation of the effects of hyper-parameters on model performance, particularly at scales that would be infeasible for fitting Gaussian Approximation Potentials (GAP) models.

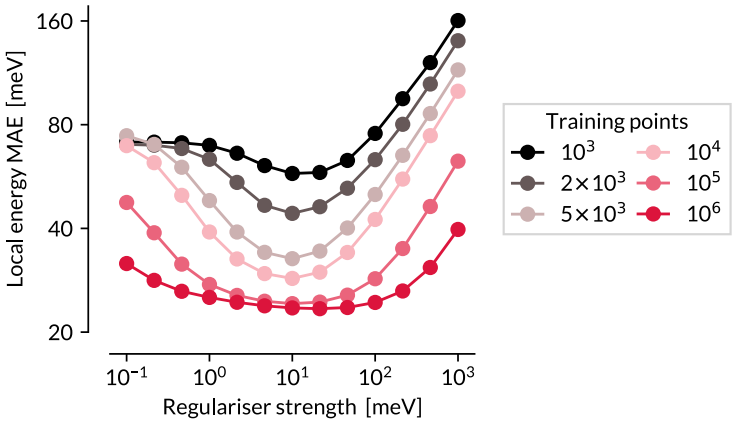


Figure 3.5: *effect of regulariser strength on the learning curves for a SoR-GPR model trained on the synthetic dataset for bulk carbon*

In Figure 3.5, I vary the regularisation strength of a SoR-GPR model with 5000 inducing points trained on up to 1M training points. At low regularisation strengths

(LHS), the model over-fits to the training data, leading to high error. At high regularisation strengths (RHS), the model is over-smoothed, under-fitting the training data, and hence also leading to high error.

As the amount of training data increases (black to red in Figure 3.5):

1. optimal regularisation strength remains static (at ≈10 meV). This is a nice property: training on very large datasets is expensive, and so being able to optimise hyper-parameters on lower dataset sizes and transfer optimality to larger datasets is a significant advantage.
2. the range of regularisation strengths that lead to near-optimal performance gets much broader (by up to two orders of magnitude). This makes sense: GPR is a Bayesian modelling technique. As more information is available to the model, the prior (in this case, regularisation strength) has less of an influence on the posterior (in this case, the learned function, and hence the final model's accuracy).

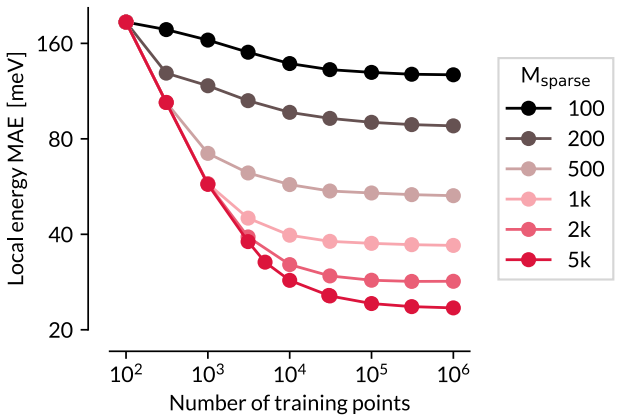


Figure 3.6: *more sparse points lead to better generalisation in the large data regime*

In Figure 3.6, I vary the number of inducing and training points for a SoR-GPR model with regularisation strength of 10 meV. Unsurprisingly, the more inducing points the model has, the better it can fit the training data. Beyond 5000 inducing points, training becomes prohibitively expensive due to the $O(MD + M^2)$ memory cost: this is an important limitation of GPR models.

3.2.5. Deep kernel learning collapse

In many settings, DKL models have been shown to surpass the accuracy of both GPR and NNs.[174–177] Here, however, I find that DKL and NN models perform near identically for all amounts of data considered. Given that DKL models are ≈100× more expensive during inference, and slower to train, I find little use for them in this setting.

To understand why this failure occurred, I analyse the latent space of the most accurate DKL model, and find that the NN component, as opposed to the coefficients of the GPR component, is responsible for all of the learning. I support this claim by noting that the accuracy of DKL models is independent of the number of inducing points used in the GPR component, with near identical accuracies for $M = 1$ and $M = 500$.

To analyse a specific case, I take the best performing DKL model, and project each test environment into its latent space. I find clear proof of collapse in this space: `t-SNE`[181] visualisations show that the rich structure present in the input SOAP space (LHS) collapses to a linear manifold in the latent space (RHS, highlighted with a line).[i] To quantify the degree of this collapse, I use a `PCA`[182] dimensionality reduction of the latent space, and find that 99.4% of the variance is explained by the first principal component!

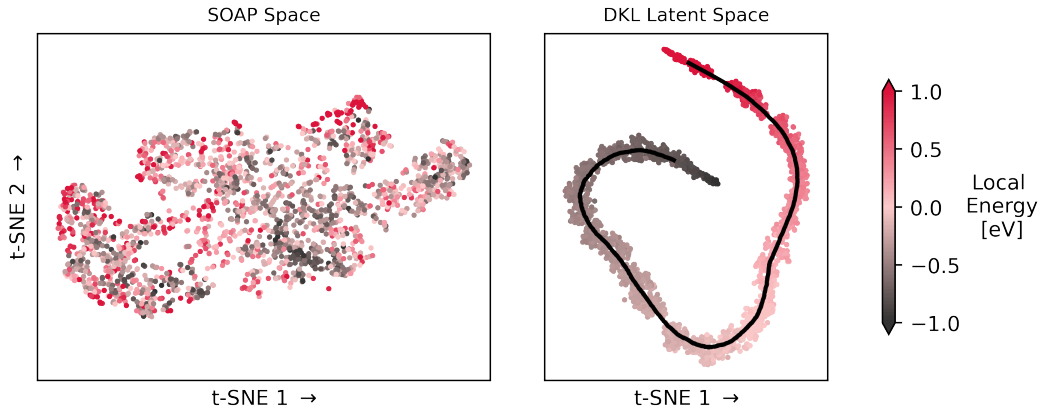


Figure 3.7: *`t-SNE`[181] visualisation of (left) the SOAP input space, and (right) the latent space of the most accurate DKL model. Each data point corresponds to a unique environment, and is coloured according to its `C-GAP-17` local energy.*

Fitting a polynomial of order 2 to this single dimension recovers near perfect accuracy *vs* the full DKL model, clearly demonstrating that the NN component is responsible for all of the learning (Figure 3.8), and has done all of the regression.

[i] to generate these visualisations, I used the `sklearn` implementation of `t-SNE`, with default settings, *i.e.* a perplexity of 30.

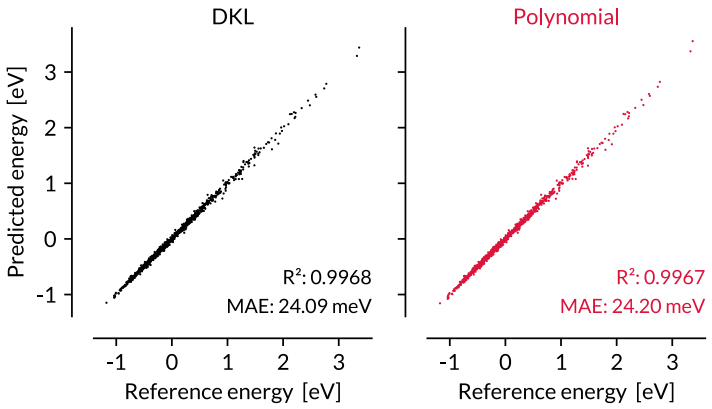


Figure 3.8: *local energy parity plots for (LHS) the most accurate DKL model, and (RHS) a polynomial of order 2 fit to the DKL model's latent space*

3.3. Learning per-structure energies

Above, I have shown that simple, descriptor-based, local energy models are able to learn synthetic local energy labels with near perfect accuracy. In the large data limit, NN models are more accurate than GPR models (due to memory constraints on the GPR models). Another advantage of NN models is that it is very easy to fine-tune them on additional data: starting from the "pre-trained" weights, exactly the same gradient descent procedure can be applied to continue learning from new data.

As a first demonstration that synthetic pre-training can improve model accuracy, I train a series of NN models (with identical SOAP+MLP architecture to above) to reproduce the *per-structure* DFT energies on all 64-atom amorphous structures from the `C-GAP-17` dataset. Local energies are not available from DFT, and so I train directly on per-structure DFT energies:

$$\mathcal{L}(\theta;\mathcal{D}) = \frac{1}{|\mathcal{D}|}\sum_{S\in\mathcal{D}}\left[E_{S,\mathrm{DFT}} - \sum_{x\in S}\varepsilon_\theta(x)\right]^2$$

where total energy predictions are made as a summation over predicted local energies, $\varepsilon_\theta(x)$, for each atom in the structure. I train two sets of models on increasing amounts of data:

1. **direct training**:
   I randomly initialise the model weights, and train the model from scratch using the above loss function.
2. **synthetically pre-trained**:
   I use the weights from the best performing synthetically pre-trained model as a starting point, and fine-tune using the above loss function. Using the validation loss as a guide, I scan over the fine-tuning learning rate

(an important hyper-parameter[183]) to find and select the best performing model.

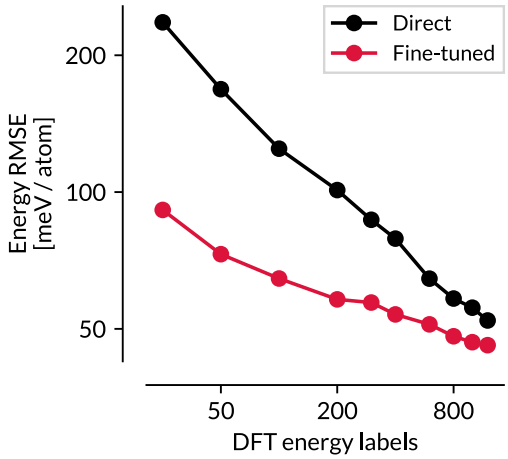


Figure 3.9: *directly trained models are systematically less accurate compared to synthetically pre-trained models*

In Figure 3.9, I plot mean test-set energy RMSE as averaged over 5-fold cross validation. For all amounts of data, the synthetically pre-trained models are more accurate than the directly trained models. This is particularly pronounced in the low data regime, with more than a two-fold improvement in accuracy.

3.3.1. Why does this work?

The success of pre-training is often attributed to the (somewhat nebulous) concept of learning "improved internal representations" of the data. Here, I study the NN models as black boxes (*i.e.* only consider inputs and outputs, and not the internal workings of the model), and find that two other effects are at least partially responsible for the improvement in accuracy:

Learning a useful local energy partition.

Most MLFFs are trained to predict a per-structure energy as a sum over per-atom contributions (as above). During training, such models therefore need to learn to assign low and high energy contributions to specific atomic environments. Because local energies are not a direct physical observable, there is not a guaranteed unique way to perform this partitioning.[184,185] In the low data regime, there are many possible partitions that would fit the training data near-perfectly, and yet will not generalise well to the test set. Without any prior knowledge, models trained from scratch on small amounts of data are therefore likely to learn sub-optimal partitions.

To probe the quality of the partition learned by the simple SOAP-NN models, I compare in Figure 3.10 the local energy predictions of the `C-GAP-17` model to those of three different SOAP+NN models that have been: (i, LHS) directly trained on DFT per-

structure energies, (ii, MID) pre-trained on `C-GAP-17` synthetic local energies, and (iii, RHS) subsequently fine-tuned on DFT per-structure energies. I used 1000 DFT labels, and 1M synthetic labels where appropriate.

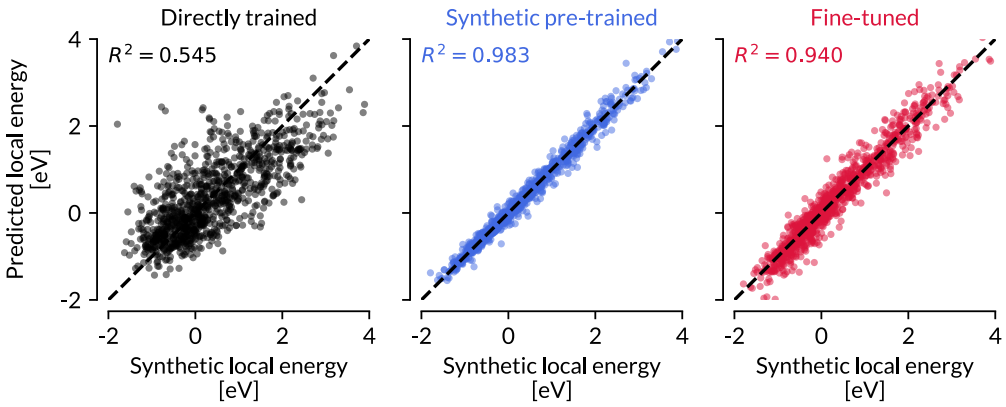


Figure 3.10: *comparing the local energy partition learned by `C-GAP-17` to three different SOAP+NN models: directly trained (LHS), synthetically pre-trained (MID), and fine-tuned on DFT data (RHS)*

I find that the partition learned by a directly trained model (Figure 3.10 LHS) is (very) different to that learned by `C-GAP-17` reference ($R^2 \approx 0.545$). In contrast, the synthetically trained model (Figure 3.10 MID) has a much higher correlation with the `C-GAP-17` reference ($R^2 \approx 0.983$); this is expected as this model was trained exactly to learn this partition. Finally, after fine-tuning on DFT energy labels (Figure 3.10 RHS) I observe that the correlation with the reference model remains high ($R^2 \approx 0.940$).

Synthetic pre-training therefore induces a useful prior on the local energy partition. Fine-tuning acts to perturb this prior without making dramatic changes to the final partition. Providing the local energy partition of the synthetic labelling source is "accurate", this naturally leads to accurate fine-tuned models, even in the low data regime.

Pre-training acts as a regulariser.

Despite employing various regularisation techniques, I find that, in all cases, NN models trained on both synthetic and DFT data exhibit a non-zero generalisation gap between the training and test errors.

However, I find that this generalisation gap is **significantly reduced** when fine-tuning models that have been synthetically pre-trained (see Figure 3.11): while the training error is higher, the test-set error is lower than the directly trained model. Useful prior knowledge, combined with a small learning rate for the fine-tuning stage, clearly acts as an additional regulariser.

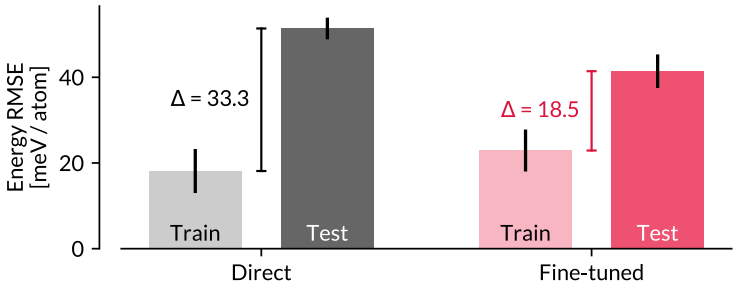


Figure 3.11: *generalisation gap for SOAP+NN energy models trained on per-structure DFT energies. Error bars show min-max ranges over 5-fold cross validation.*

3.4. Synthetic pre-training

Unlike the simple models above, all modern MLFFs are trained on, and can predict, *forces* as well as energies. In this section, I therefore extend the above approach by pre-training a GNN-MLFF on forces and per-structure energy labels from my synthetic dataset, before fine-tuning on increasing amounts of the full `C-GAP-17` dataset, and validating on both energies and forces.

3.4.1. Training details

To train on both energies and forces, I used as a loss function a weighted sum of the squared error in energy, and the mean squared error in force components:

$$\mathcal{L}(\theta; S) = \frac{\lambda_E}{|S|} \cdot \left[E_{\mathrm{DFT}}(S) - \hat{E}(S;\theta)\right]^2 + \frac{\lambda_F}{3 \cdot |S|} \cdot \sum_{i \in S} \sum_{\alpha \in \{x,y,z\}} \left[F_{\alpha,\,\mathrm{DFT}}(i, S) + \frac{\partial \hat{E}(S;\theta)}{\partial r_{\alpha,i}}\right]^2$$

where the energy error is weighted by the number of atoms to avoid bias towards preferentially fitting to larger structures.

I used the `NequIP`[94] architecture, and optimised the hyper-parameters by training on the full `C-GAP-17` dataset, and selecting the best performing model on a validation set. This selected a 4 Å cutoff, 4 message passing layers, `32x0e + 32x1o` node features, an `Adam` optimiser with a learning rate of 0.001, and an exponential moving average decay factor of 0.99. To balance the energy and force performance, I set $\lambda_E = 4, \lambda_F = 0.1$

I also re-labelled `C-SYNTH-23M` using an `ACE`[109] potential (`C-ACE-23`)[158] and an updated Gaussian Approximation Potential (`C-GAP-20U`),[72] as well as the `LCBOP`[186] and `EDIP`[187,188] empirical potentials.

3.4.2. Learning curves

I start by pre-training a series of `NequIP` models on increasing amounts of the synthetic dataset as labelled by the `C-ACE-23` potential. I take each of these models, and

fine-tune on increasing amounts of the full `C-GAP-17` dataset, and compare the test-set accuracies of these fine-tuned models to a series of directly trained models (Figure 3.12).

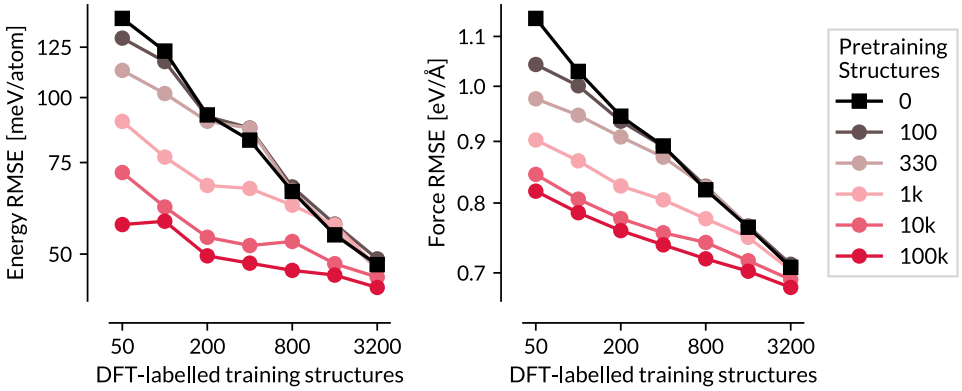


Figure 3.12: *learning curves comparing synthetically pre-trained and directly trained* `NequIP` *models on a hold-out test set of the* `C-GAP-17`

Provided sufficient amounts of synthetic data are used (≈2× the amount of DFT fine-tuning data), synthetic pre-training leads to improved energy and force predictions, as expected. Furthermore, increasing the amount of synthetic pre-training data (black to red) acts to systematically improve model accuracy. This is a clear demonstration that synthetic pre-training is useful in realistic MLFF training settings.

3.4.3. Optimal resource allocation

MLFF researchers often have some fixed budget available to label training data for their model (measured in *e.g.* dollars, cluster credits, CPU hours, days-to-deadline *etc.*). In the direct training setting, deciding how to spend this budget is trivial: 100% is spent on DFT calculations. However, in the context of synthetic pre-training, the final performance of the model depends both on the amount of synthetic and of DFT data used during training. In order to maximise final model accuracy, researchers therefore need to decide how to allocate their computational resources between synthetic and DFT data generation. Below, I demonstrate how small scale experiments can be used to determine this optimal allocation.

Final model performance depends smoothly on both the amount of synthetic and of DFT data used during training. In Figure 3.13, I plot the force RMSE loss surface for the `NequIP` models presented above as a function of these amounts of data:

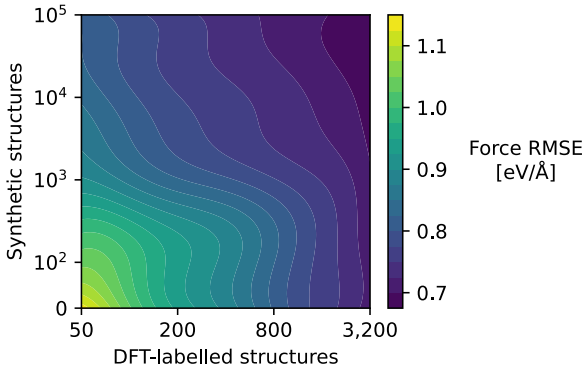


Figure 3.13: *force RMSE loss surface for the* `NequIP` *models trained on* `C-SYNTH-23M` *with* `C-ACE-23` *labels.*

Finding the allocation of resources that maximises final model accuracy requires sampling this loss surface along a line of constant cost, and finding the point along this line that minimises the loss.

Assuming that the cost to generate and label a synthetic structure is 1/100th the cost of a DFT calculation, I plot such iso-cost lines in Figure 3.14a, and track the loss along these paths in Figure 3.14b.

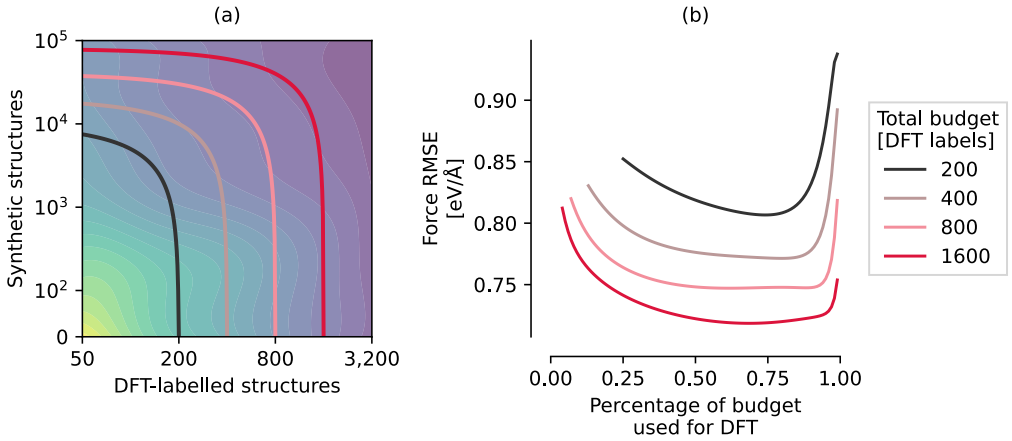


Figure 3.14: *lines of constant labelling cost (as measured in DFT-label equivalents, assuming synthetic structure generation and labelling is 1/100th the cost of DFT labelling) projected onto (a) the force RMSE loss surface, and (b) a shared x-axis*

Pleasingly, and as observed in other subsequent works,[183] loss varies smoothly with the percentage of total budget used for DFT labelling. Moreover, increasing the total budget leads to (a) a flatter landscape, with a wider range of near-optimal allocations, and (b) an approximately static minimum.

These results suggest the following effective strategy for the application of synthetic pre-training at very large scales:

1. use some small amounts of the total budget (*e.g.* 5% each) to both generate synthetic data, and label with *ab initio* methods.
2. use small scale experiments to train several models along lines of iso-cost.
3. use the optimal allocation determined from the small scale experiments to generate and label the final dataset.

3.4.4. Varying the synthetic source

To further probe the effectiveness of synthetic pre-training, I pre-train additional series of `NequIP` models on 10^5 synthetic structures labelled with various approximate methods before fine-tuning on increasing amounts of the full `C-GAP-17` dataset (Figure 3.15).

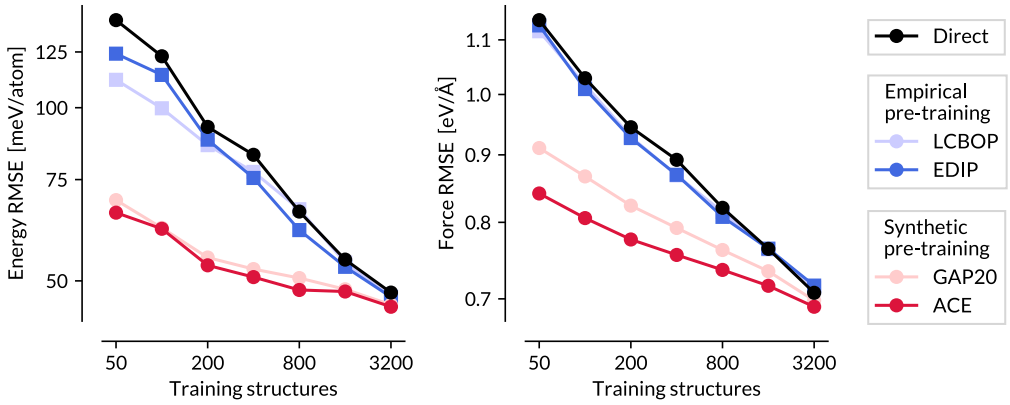


Figure 3.15: *loss surface for the C-ACE-23 model trained on the synthetic dataset for bulk carbon*

Clearly, different synthetic sources lead to different degrees of "positive transfer" (*i.e.* how much improvement is seen in the final model with respect to the directly trained model). The empirical models provide almost no benefit. This is unsurprising: these models are (i) relatively crude, low-body-order approximations, (ii) not designed for application to a wide range of the geometries present in the `C-GAP-17` dataset, and hence (iii) do not provide a useful prior for fitting to the DFT labels. In contrast, the increasingly accurate MLFF models (`C-ACE-23` is more closely aligned with the DFT labels than `C-GAP-20U`) provide increasingly better improvements, as they are (i) explicitly many-bodied, (ii) applicable to the whole of the `C-GAP-17` dataset, and hence (iii) provide a useful prior for fitting to the DFT labels. The strong degree of positive transfer I have demonstrated in the above sections is therefore contingent on my use of a synthetic source that is moderately well aligned with the targetted DFT labels.

To provide quantitative evidence for these conclusions, I report R^2 values between all force labels on the test set, as generated by the synthetic source and the DFT labels,

and quote the respective degree of positive transfer, as measured by the improvement in force RMSE at 200 training structures, in Table 3.3. As can be seen, the degree of positive transfer is largest when the agreement between the synthetic and DFT labels is highest.

Table 3.3: *increased alignment between synthetic and ab initio force labels is accompanied by a larger degree of positive transfer*

Source	R^2	Transfer
EDIP[187,188]	0.173	2%
LCBOP[186]	0.252	2%
C-GAP-20[72]	0.859	13%
C-ACE-23[109,158]	0.935	18%

3.5. Beyond numerical errors: robustness in dynamics

MD simulations serve as a primary application for MLFFs, but also function as rigorous "adversarial" tests of a model's reliability and robustness. Concretely, because (high temperature) MD efficiently explores configurational space, it is (rapidly) attracted to unphysical "holes" or artificial basins in the PES should they exist.

Modern GNN-MLFFs trained on limited *ab initio* data are particularly susceptible to these failures: when these models encounter regions of chemical space poorly represented in the training set, they often produce (wildly) inaccurate predictions, leading to a total breakdown of the simulation. A central concern regarding synthetic pre-training is whether fine-tuned models can maintain the robustness required for stable MD despite using only small amounts of *ab initio* data.

To investigate this, I trained two `NequIP` models on 240 carbon nanotube structures from the `C-GAP-20U` dataset.[72] The first model was trained from scratch, while the second was fine-tuned from a checkpoint pre-trained on 10,000 synthetic structures labelled with the `C-ACE-23` potential. Force errors on a carbon nanotube test set were within 5% for the two models. Both models were then used to drive NVT simulations, ramping from 300 K to 1,000 K over 100 ps, and initialised from a challenging structure containing predominantly in-domain sp^2 carbon atoms, but also some out-of-domain sp and sp^3 hybridized carbon atoms. I compare snapshots along these trajectories in Figure 3.16.

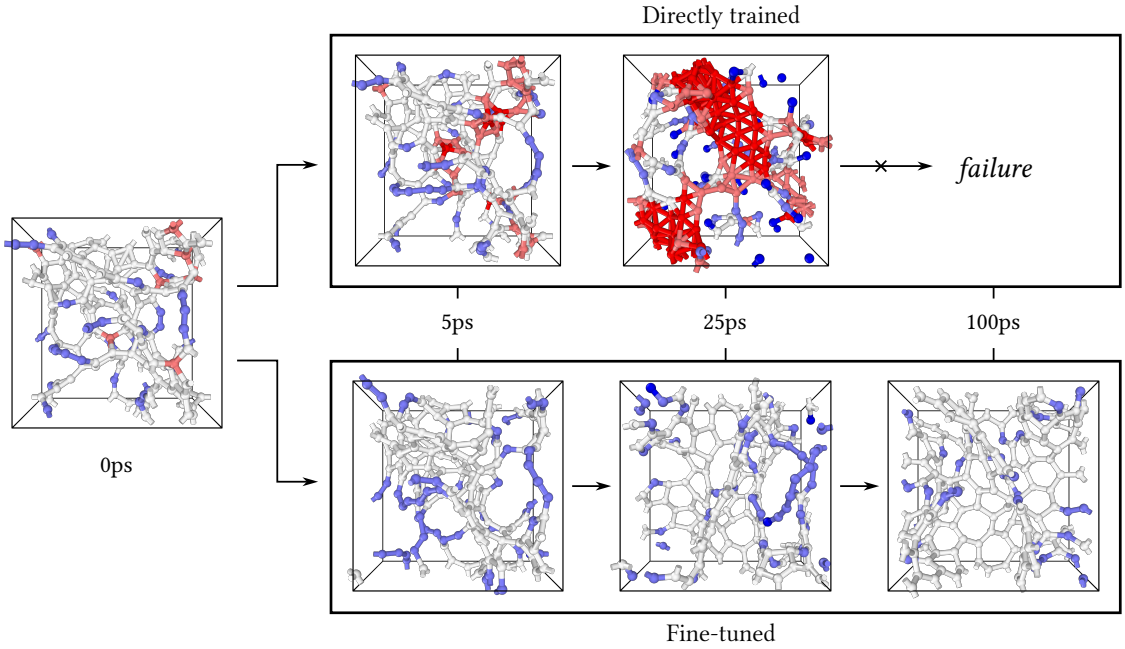


Figure 3.16: *stability of the MD simulations for the synthetic dataset for bulk carbon*

As expected, the directly trained model (Figure 3.16 top) breaks down very quickly (while the temperature is still <500 K), generating completely unphysical, ultra high density structures containing 8+ coordinate carbon atoms. In contrast, the synthetically pre-trained model (Figure 3.16 bottom) is completely stable, and generates a qualitatively correct annealed graphitic structure.

The disparity in performance offers a clear demonstration of the benefits of synthetic pre-training. The diverse configurations encountered during the synthetic pre-training phase appear to "regularise" the PES, preventing the formation of unphysical artifacts. Indeed, testing the performance of the final two models reveals that the synthetically pre-trained model remains numerically accurate across the entire carbon domain (*i.e.* does not over-specialise to the carbon nanotube structures during fine-tuning), while the directly trained model, that has never seen *e.g.* diamond, graphite, *etc.*, displays poor performance (Figure 3.17).[189]

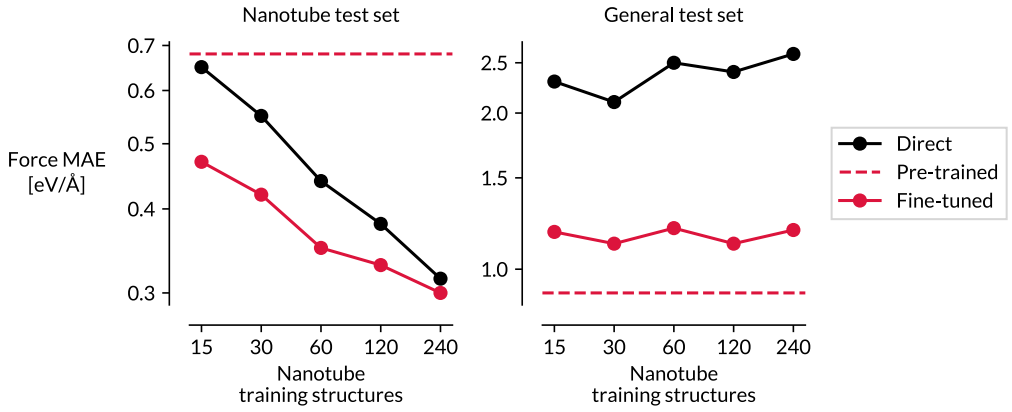


Figure 3.17: *performance of directly trained, synthetically pre-trained, and fine-tuned models on the carbon nanotube and general test sets, as a function of the number of carbon nanotube training/fine-tuning structures*

The broad configurational knowledge gained during pre-training clearly carries over through fine-tuning, resulting in a model that is both numerically precise and physically robust during long-term integration.

3.6. Summary and outlook

Labelling structures with *ab initio* methods is expensive. This makes traditional MLFF training workflows slow and costly. Synthetic pre-training can dramatically reduce this cost, while simultaneously improving model accuracy and robustness.

Despite the successful application to bulk carbon I have demonstrated above, it is important to acknowledge the limitations of this approach: (i) fine-tuning the model on specialised *ab initio* datasets leads to a small decrease in the general purpose accuracy of the model (Figure 3.17), (ii) an approximate PES method needs to be available for the system of interest that is both cheap and has strong agreement with the final *ab initio* labels, and (iii) expert domain knowledge is required to devise and run MD protocols that generate diverse and representative synthetic datasets.

Since the publication of this work, these limitations have been successfully addressed: (i) multi-head replay fine-tuning now allows for efficient fine-tuning on specialised datasets without a loss in general purpose accuracy,[190] (ii) atomistic foundation models can now be used as an approximate PES method for almost all chemical systems of interest,[11,12,86,143,146] and (iii) I have developed an automated synthetic data generation pipeline that requires no expert domain knowledge, while also being more efficient than MD-based approaches.[191] In the following chapter, I introduce this new method, and use it to generate specialised, fast and highly accurate MLFFs for a range of chemical systems.

Attribution and Acknowledgements

The above chapter presented work from two of my first-author papers:

Chapter 3.1 to Chapter 3.3:

> Synthetic data enable experiments in atomistic machine learning
> *John L.A. Gardner, Z. Faure Beaulieu, V.L Deringer*,
> Digital Discovery, 2023, Volume 2, Issue 3, Pages 651-662

I fitted and evaluated all models and carried out numerical experiments.
Z.F.B. carried out a key proof-of-concept study.
We all contributed to discussions.

Chapter 3.4 to Chapter 3.5:

> Synthetic pre-training for neural-network interatomic potentials
> *John L.A. Gardner, K.T. Baker, V.L Deringer*,
> Machine Learning: Science and Technology, 2024, Volume 5, Issue 1, Pages 015003

I fitted and evaluated all models and carried out numerical experiments.
K.T.B. carried out a key proof-of-concept study.
We all contributed to discussions.
I am grateful to Dr. Zakariya El-Machachi, Dr. Zoé Faure Beaulieu, and Dr. Daniel Thomas du Toit for useful discussions, and for assistance in labelling the synthetic dataset with empirical methods.

4. Automated distillation of machine-learned force fields

The central objective of MLFF research is to develop models capable of rapidly and accurately approximating the *ab initio* PES for an arbitrary target chemical system.

Traditionally, researchers have trained bespoke, single-purpose MLFFs for each unique system of interest.[58,72,73,123,128,158] However, the up-front cost of labelling large numbers of atomic structures with *ab initio* methods is a major bottleneck, making this approach slow and expensive when exploring new chemical spaces (development times have typically been on the order of months to years).

Atomistic foundation models[11,12,86,143,146] are currently transforming this landscape by drastically reducing the effective end-user cost to reach chemical accuracy for new target systems. In this approach, a few big companies and labs invest heavily in the one-time, up-front cost of generating and labelling large and diverse datasets of atomic structures. Over expensive, multi-GPU training runs, "foundation models" are then generated by (pre-)training on these datasets. In the process, these models learn to capture all important kinds of chemical interaction, and develop robust and generalisable internal representations of atomic environments. To reach chemical accuracy on new systems, practitioners with modest computational resources can then **fine-tune** these models on **very small** amounts of *ab initio* labelled data, dramatically reducing the time and cost to develop an accurate MLFF for a new system.

Unfortunately, (current) atomistic foundation models are big and slow. This naturally carries over into their fine-tuned counterparts, limiting their practical applicability despite impressive accuracy and robustness.

In this chapter, I propose methodology to tackle this problem by automatically "distilling" (fine-tuned) foundation models into smaller, faster models for use on a targeted domain.

4.1. Foundation models are expensive and slow

To achieve strong numerical accuracy, foundation models (empirically) require:

1. **large and diverse training datasets**, in part to improve the generalisability of the internal representations learned by the model, and in part to ensure that typical use cases are already represented in the training data. Large research efforts have gone into compiling these datasets over the

last few years, and the trend for increased scale appears to be continuing (Table 4.1)

2. **high parameter counts**, in order to accurately model the PES for a very wide range of chemical systems. Scaling laws show no sign of diminishing returns for large models beyond even 1B parameters (see *e.g. Wood et al.*, 2025[12] and *Kreiman et al.*, 2025[192])
3. **large radial cutoffs**, in order to ensure that foundation models are broadly applicable, and specifically to accurately capture important mid-to-long range interactions found in some chemical settings

Table 4.1: *a decade of MLFF scaling*

Model	Params	Cutoff [Å]	Structures	Date
MTP[193]	10k	5	10k	2015/12
SchNet[8]	70k	5	110k	2017/06
DimeNet[92]	2.1M	5	110k	2020/03
M3GNet[108]	230k	5	190k	2022/09
CHGNet[52]	410k	5	1.6M	2023/03
GNome[16]	16M	5	89M	2023/11
MACE-MP0[11]	5M	6	1.6M	2023/12
MatterSim-v1-5M[146]	4.6M	5	17M	2024/05
Orb-v2[86]	25M	10	32M	2024/10
eSEN-30M-MP[85]	30M	6	113M	2025/03
UMA-M[12]	1.4B	6	101M	2025/06
Allegro-OAM-L[138]	10M	7	112M	2025/08

While the size of the training dataset has no influence on final model inference speed, parameter counts and radial cutoffs do. Concretely, inference cost scales:

- as $O(c^3)$, where c is the radial cutoff, for models that consider pair-wise messages only (Figure 4.1a), and as $O(c^6)$ for models that consider triplets (such as *e.g.* MatterSim[146] and M3GNet[108])!
- approximately linearly with parameter count (Figure 4.1)

These scalings make foundation models (with their large parameter counts and radial cutoffs) slow to use in practice, preventing their application by researchers with only modest computational resources and who want to study *e.g.* large systems, or long timescales.

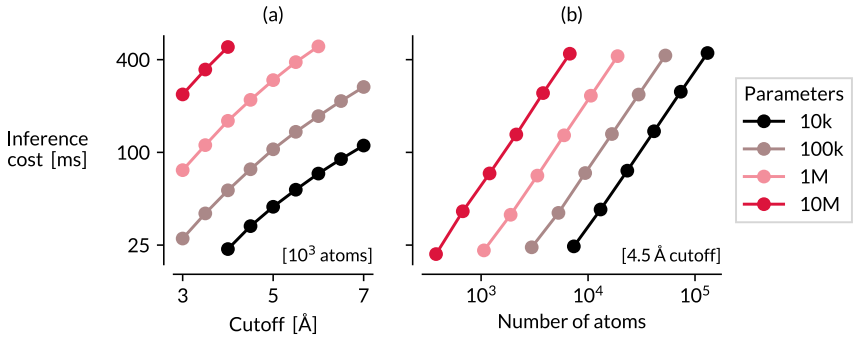


Figure 4.1: *inference cost, measured in ms per prediction (lower is better), for a series of* `PaiNN` *models with different parameter counts and radial cutoffs*

4.2. Accelerating large models

The utility of any MLFF is directly related to its inference speed: smaller, faster models allow higher throughput for a given system size, and/or allow scaling to larger system sizes on a single machine.

Unsurprisingly, therefore, heavy research efforts are being expended on making large MLFFs, and in particular foundation models, faster. Beyond the trivial strategy of relying on Moore's Law[13] to provide "free" speed-ups via hardware improvements, several orthogonal strategies are being explored:

1. **optimised kernels**.
   Many GNN MLFFs rely on sophisticated tensor operations.[10,84,94,100] Naïve implementations of these operations can often be slow and memory-intensive. Optimised, hardware-aware implementations can therefore provide significant speed-ups. In direct analogy to flash attention for transformers,[194] several kernel libraries have been developed for MLFFs, with a particular focus on accelerating the equivariant tensor operations used by spherical tensor models (such as `MACE`, `Allegro` and `Nequip`).[195–197] Concurrently, improved performance from deep learning frameworks, and in particular the compilation features of `PyTorch:2.0`,[59] are also providing similar speed-ups.[138,143]
2. **smaller cutoffs for expensive operations**.
   Even when using optimised kernels, high-order tensor operations remain expensive and, in particular, memory intensive. An emerging strategy to address this is to use smaller radial cutoffs, and hence consider fewer terms, for these expensive operations. In some settings, this can lead to a significant speed-up without noticeably sacrificing accuracy.[198]
3. **mixture of experts**.
   A common strategy in computer vision and natural language processing is

to use a large model composed of multiple smaller models, each of which is an "expert" in a particular region of the input space.[199,200] At inference time, only a single expert is used. Effective model sizes can be increased *without increasing inference costs* by using more experts, rather than increasing the size of each expert. Recently, this strategy has been applied to MLFFs in the `UMA` family of models.[12] To avoid hot-swapping between experts during inference, *Wood et al.* generate a single expert based on a global context vector. Unfortunately, this leads to a lack of physicality in some settings (as discussed by *Batatia et al.*, 2025: Table XII[190]).

4. **eliminating equivariance**.
   Models that maintain equivariant internal features have been shown empirically to be more data efficient.[100] This improved efficiency typically comes at the cost of more sophisticated tensor operations, however. Non-equivariant architectures (including the `Orb` family[86,143] and recent application of vanilla `Transformer` models[110,192,201]) can be (significantly) faster at inference. However, the lack of internal equivariance of these models leads to a lack of both energy invariance and force equivariance under rigid rotation: this unphysicality is a major limitation, and makes these models unsuitable to drive molecular dynamics simulations.[84]
5. **direct force predictions**.
   The dominant strategy for generating force predictions is to numerically calculate the gradient of the energy with respect to atomic positions. This "backward pass" through the model leads to an approximate doubling of model latency (Figure 4.2). One way to avoid this doubling is to predict forces directly, typically using a small MLP readout of the model's latent representation (*e.g.* `Orb`[86,143]). However, this approach severs the direct equality between the model's true energy gradients and the predicted forces: the resulting models no longer maintain conservation of energy, and hence are not suitable for driving: (i) force-based structural relaxation procedures (just following the forces can prevent the system from every converging), and (ii) molecular dynamics simulations (temperatures often increase in an uncontrollable manner).[83]
6. **maximum neighbour counts**.
   The inference cost of pairwise message passing models scales linearly with the number of edges in the graph. Typically, graphs are constructed such that all edges within some cutoff radius are included (leading to $O(c^3)$ inference scaling). By limiting the number of neighbours per atom to some maximum value, k, inference costs are capped at some fixed value as c increases (Figure 4.2), facilitating the modelling of under-dense systems while maintaining fast inference for dense systems. Unfortunately, such an approach leads to a loss of physicality, as the model's energy and

force predictions are no longer guaranteed to be smooth with respect to perturbations of the atomic positions.[85]

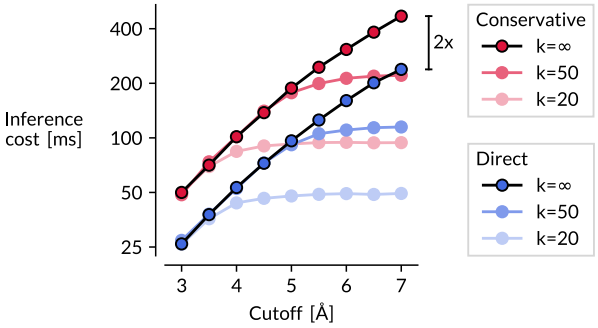


Figure 4.2: *inference cost, measured in ms per prediction for a 1000 atom structure (lower is better), for* `Orb` *models that consider k nearest neighbours, and that make force predictions directly (blue) or conservatively (red)*

While hardware and software improvements continue to provide speed-ups, current scaling trends suggest that foundation models will continue to become more, not less, expensive to use in practice. Is there some way to alter these models to make them faster for specific systems without compromising their physicality?

4.3. Force field distillation

Model distillation, originally proposed by *Hinton et al.*,[202] refers to the class of techniques for training a smaller model (the student) to mimic the outputs of a larger model (the teacher), aiming to accelerate inference while maintaining accuracy. These approaches have found success in a wide range of applications, particularly in computer vision and natural language processing.[203–206]

The primary goal of all model distillation approaches is to extract maximal amounts of relevant information from the teacher model in a fast and cheap manner. While MLFF distillation is a relatively new field, two major axes along which methods tend to vary have emerged:

1. **the teacher-student loss function**: different methods attempt to extract different kinds of information from the teacher model. Initial MLFF distillation approaches "just" train the student to mimic the teacher's force, per-structure-energy and stress predictions.[207] A natural evolution of this method is to also train the student to predict the teacher's local energy contributions (see Chapter 3).[189] Beyond "normal" PES properties, *Amin et al.*[208] proposed also training to predict (random projections of) the teacher's energy Hessian, *i.e.* the *second* derivative of the energy with respect to atomic positions. Finally, *Kelvinius et al.*[209] proposed moving beyond the outputs that are available for arbitrary black box models, and

instead to also align the *latent representations* of the teacher and student models.

2. **the training data**: different distillation procedures use different kinds of dataset to extract information from the teacher. The most basic approach is to use an existing dataset of structures. This works well for systems that have already been well explored, but, by definition, cannot be used to explore arbitrary new systems. To generate new datasets for distillation, existing methods borrow approaches used to generate *ab initio* datasets. In particular, the two dominant approaches are (i) using (teacher- or *ab initio*-driven) MD to generate trajectories of structures, as first introduced by *Morrow and Deringer*,[207] and used in Chapter 3[210] and more recently by *Wang et al.*[211] and (ii) using active learning to iteratively expand a reference dataset according to some uncertainty metric.[212,213]

While existing MLFF distillation methods have met with some success,[18,214] they are difficult to automate and scale: both active learning pipelines and MD simulations require expert domain knowledge to set up and run, and also typically require many teacher model evaluations (1,000+) to generate each uncorrelated training structure. In addition, methods that require specialised model training set-ups (including training on Hessians, and internal representations) restrict their applicability to architectures that are supported by these specialised software packages.

Below, I propose a new approach to MLFF distillation that addresses these issues. In the data generation stage, I use the teacher model to automatically explore the PES of the target system via an automated rattle-relax-repeat protocol. With no requirement for domain expertise, this method efficiently generates very large datasets of synthetic data, upon which student models can then be directly trained in a vanilla fashion using existing, non-specialised training codes.

4.4. Automated potential energy surface exploration

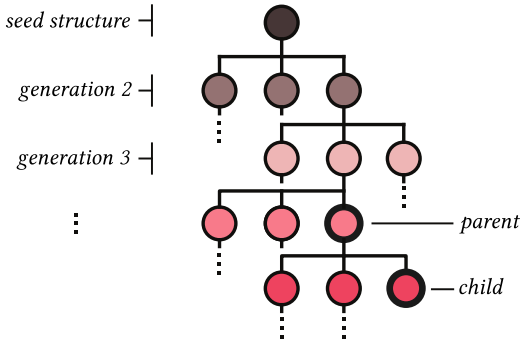
In generating a dataset of structures on which to train a student model, the ideal method has 4 key properties:

- **automated**: any technique that relies on manual steps is not scalable, and therefore not practical for large scale applications. Instead, ideal techniques should require no domain expertise or manual intervention.
- **fast**: the approach should require as few teacher model evaluations as possible to generate each structure, and ideally should be trivially parallelisable across multiple processes where available.
- **relevant**: a vast majority of possible atomic configurations (as generated by placing atoms randomly in space) are unphysical and unreachable in normal simulations, and therefore completely irrelevant. Generating such

structures is not only wasteful, but training on such can also significantly worsen the fit of MLFFs![215]

- **wide-spread**: all relevant regions of the PES should be sampled approximately uniformly, *i.e.* without regenerating similar structures many times over, or preferentially sampling the lowest energy structures at the expense of other important regions.

Inspired by random structure searching methods,[151,216,217], the approach I propose below satisfies these criteria by employing an iterative rattle-relax-repeat procedure. The protocol starts from a given seed structure, and rapidly generates a "family tree" of structures:



Generating a new child structure to add to this tree involves three steps:

1. picking an existing "parent" structure from the tree
2. (aggressively) rattling the parent, and
3. crudely relaxing it using the teacher model's force predictions

The relaxation step is constructed to rapidly generate a new, relevant, child structure that is sufficiently dissimilar to those already in the tree. Parallelisation is trivially possible by generating multiple trees, each starting from a different seed structure, in parallel. Below, I describe each of these three steps in detail. Default hyper-parameters are given at the end in Table 4.2.

Step 1: choosing a parent structure

A "parent" structure from which to generate a new "child" is chosen at random from all structures already in the tree, according to the following probability distribution:

$$\mathbb{P}_i = \beta \cdot \frac{e^{-E_i/kT}}{\sum_j e^{-E_j/kT}} + (1-\beta) \cdot \frac{G_i}{\sum_j G_j}$$

where:

- $E_i \in \mathbb{R}$ is the (teacher-labelled) energy of the i-th structure
- $G_i \in \mathbb{Z}^+$ is the "generation number" of the i-th structure
- $0 \leq \beta \leq 1$

- k is the Boltzmann constant
- T is a user-defined temperature

This selection process balances exploration and exploitation: high β values favour the selection of lower energy structures according to the Boltzmann distribution, no matter their position in the tree. Low β values favour the selection of structures from later generations in the tree: these tend to be geometrically less similar to the seed structure, and hence "explore" more of the PES (see Figure 4.4 below).

Step 2: rattling the parent structure

The selected parent structure is then rattled to generate a new child structure according to the following transformation:

$$R' \leftarrow [R(A+I)] + B, \qquad C' \leftarrow [C_0(A+I)]$$

where:

- R and $R' \in \mathbb{R}^{N\times 3}$ are the parent and child atomic positions, respectively
- C_0 and $C' \in \mathbb{R}^{3\times 3}$ are the *seed structure's* and child unit cell, respectively[i]
- $I \in \mathbb{R}^{3\times 3}$ is the identity matrix
- $A \in \mathbb{R}^{3\times 3}$ is a scaling matrix with entries sampled from $\mathcal{N}(0, \sigma_a)$
- $B \in \mathbb{R}^{N\times 3}$ is a translation vector with entries sampled from $\mathcal{N}(0, \sigma_b)$
- N is the number of atoms in the structure

In plain language, the unit cell is randomly scaled and sheared by some small perturbation, A. The fractional coordinates of the atoms are preserved under this transformation, before being rattled in Cartesian space according to some Gaussian noise vector, B. In the case of isolated structures, unit cells are not considered, and $A = 0^{3\times 3}$. User defined parameters σ_a and σ_b control the amplitude of cell perturbation and atom rattling, respectively.

Step 3: relaxing the child structure

The rattled parent structure is then relaxed to generate a new child structure. This step requires more nuance than the previous two: rather than using existing methods designed to drill down to the precise minimum energy geometry, we instead want to generate a structure with reasonable energy **in as few steps as possible**. To do this, I propose using the teacher model's force predictions to drive a Robbins-Monro[218] style relaxation, using the following update rule:

$$R_{t+1} \leftarrow R_t + \frac{\sigma_b}{t+1} \cdot \hat{F}_t$$

where:

- t is the number of steps taken so far

[i] the original seed structure, as opposed to the parent structure, is used for the unit cell augmentation to prevent vanishing or exploding unit cells in late generations

- $\hat{F}_t \in \mathbb{R}^{N\times 3}$ are the **direction vectors** of the teacher model's force predictions for the t-th structure
- $\sigma_b \in \mathbb{R}^+$ is the same user defined parameter defined above, and which here controls the step size of the relaxation

The number of steps taken is capped at some (small) maximum value M. To prevent over-sampling of low energy or highly correlated structures, I stop this relaxation process early ($t < M$) subject to the following criteria:

1. random early stopping at each step with probability:

$$\min\left(0.25, e^{-\Delta E/kT}\right)$$

   provided the maximum force magnitude is less than some threshold F_{max}, and where Δ_E is the difference in energy between the non-rattled parent and current structure.
2. immediately if the mean atom-wise separation between the current child and any structure in the tree is less than some threshold d_{min}

Criterion 1 admits, with low probability, the sampling of high energy structures, while also preventing the repeated generation of low energy structures. Criterion 2 guarantees diversity in the tree of generated structures by enforcing a strict minimum distance, in Cartesian space, between all pairs of structures.

4.4.1. augment-atoms

I have implemented this method in the open-sourced, and GPU accelerated, `augment-atoms` package.[191] Important features include:

- **speed**: the Robbins-Monro relaxation generates geometrically diverse structures very quickly, typically requiring ≈5–10 teacher model evaluations per structure, *cf.* ≈10^3 for MD simulations.
- **scalability**: it is trivial to parallelise this process over multiple machines, simply by assigning each machine its own seed structure, and generating trees in parallel.
- **widely applicable**: `augment-atoms` supports the use of any teacher model that provides an `ASE`[147] calculator object. This is a nearly universal interface, and allows the use of almost all existing PES, including empirical models, DFT codes if required, foundation models, and all MLFF architectures.
- **automated**: no manual input or domain expertise is required to generate these large datasets of synthetic data, lowering the barrier to entry for many researchers. Despite relying on many hyper-parameters, I find the method is very robust to changes in these values, and that the default values (given in Table 4.2) work well for a wide range of systems.

Table 4.2: *default hyper-parameters for* `augment-atoms`

Parameter	Value	Description
T	300 K	expected temperature of the system
β	0.5	exploitation/exploration trade-off
σ_a	$\in [0, 0.1]$ Å	range of cell rattle amplitudes
σ_b	$\in [0.01, 0.1]$ Å	range of atom rattle amplitudes
M	30	maximum relaxation steps
F_{max}	50 eV/Å	maximum force magnitude
d_{min}	0.1 Å	minimum geometric parent-child dissimilarity (RMSD)

4.4.2. Application to liquid water

To demonstrate the application of `augment-atoms`, I generate a family tree of structures that explores the PES for liquid water. Starting from a single bulk liquid structure (sourced from *Cheng et al.*, 2019[219]) I use a fine-tuned `MACE-MP0-b3`[11] model (see details below) to generate a family tree of 1100 structures using `augment-atoms` with default hyper-parameters. This took 15 minutes on a single NVIDIA A6000 GPU.

In Figure 4.4, I visualise the distribution of the population in each "generation" both as a function of the number of steps taken, and in total.

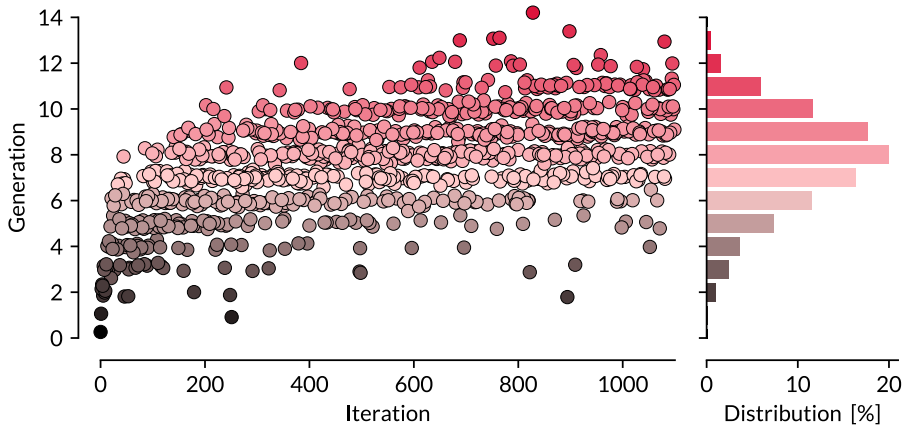


Figure 4.4: *evolution of the population of structures in each generation of the family tree*

As the exploration process proceeds, parent structures are sampled preferentially from later and later generations, leading to a wide exploration of the chemical space. The stochastic nature of the process means that low generation numbers are still sampled at late times, however. Overall, the distribution of populations in each generation tends towards a roughly normal distribution.

To demonstrate that later generations are indeed geometrically less similar to the seed structure, I calculate the root mean square displacement (RMSD) in the atomic positions of each structure relative to the initial seed structure, and visualise the distribution of these RMSDs for each generation separately in Figure 4.5.

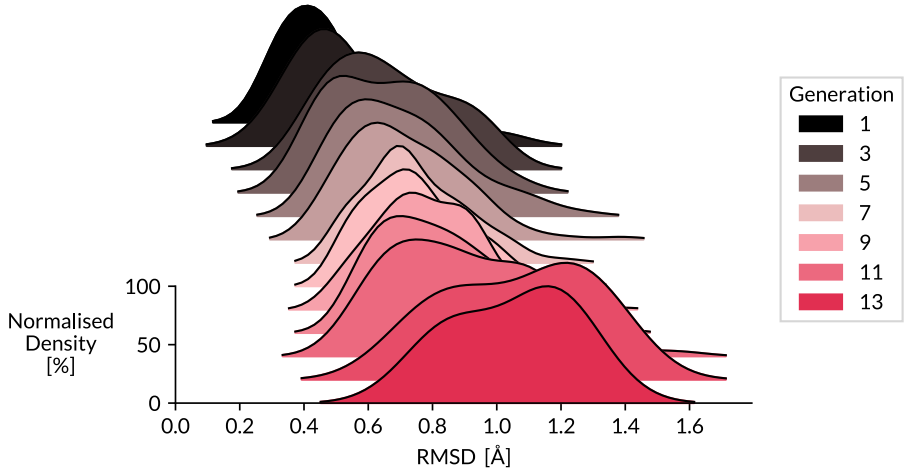


Figure 4.5: *distribution of root mean square displacement for each generation of structures in the family tree*

Finally, I explore the three-way relationship between RMSD (x-axis), energy (y-axis) and generation number (colour coded) all as relative to the seed structure in Figure 4.6.

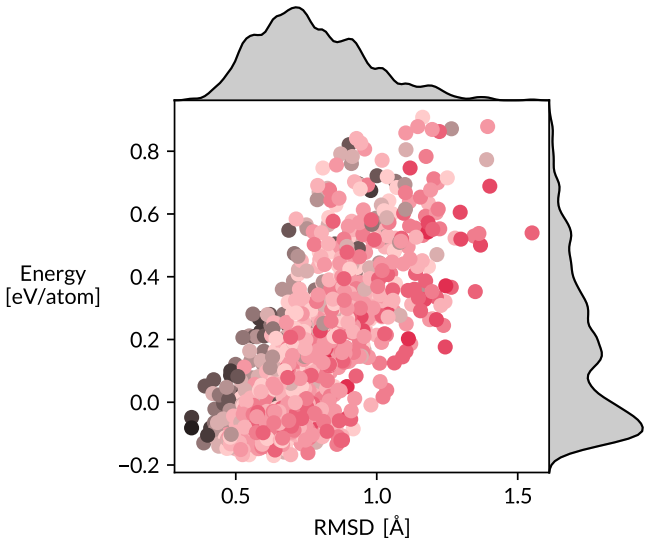


Figure 4.6: *each structure projected into energy-RMSD space, and colour coded by generation number. Marginal distributions are shown at the edges of the plot.*

The stochastic nature of the process means that the distribution of structures along these three dimensions is noisy. Nevertheless, it is clear that the lowest energy structures are found after several rattle-relax iterations, and in particular in generations 5–10. While most structures are within 0.2 eV/atom of the seed structure, structures in all generations also explore higher energy regions, sampling important, strained configurations that would not be accessible under the Boltzmann sampling of the PES using MD simulations.

4.5. Proof of concept

To demonstrate the application of `augment-atoms` to the targeted distillation of foundation MLFFs, I attempt to distil the (heavy and slow) `MACE-MP0-b3`[11] model into a (small and fast) `PaiNN`[93] model, targeting the system of bulk liquid water. This process involves three steps:

1. I use `graph-pes` to fine-tune the `MACE-MP0-b3` model on 25 structures from *Cheng et al.*'s dataset of liquid water structures.[219] This raises the level of theory from PBE[220] to hybrid revPBE0,[221–223] with Grimme D3 dispersion correction.[224] The fine-tuning process took 10 minutes on a single NVIDIA A6000 GPU.
2. I use this fine-tuned model to drive `augment-atoms`, generating a separate, 400-structure family tree for each of the 25 training structures. Generating this synthetic dataset of 10k structures took 5 hours on a single NVIDIA A6000 GPU.
3. Finally, I use `graph-pes` to train a `PaiNN` model with 50k parameters and a radial cutoff of 4.5 Å on this synthetic dataset (compared to the 5M parameter, 6 Å cutoff `MACE-MP0-b3` model). The training process took 3 hours on a single NVIDIA A6000 GPU.

I select the training structures from the first quarter of *Cheng et al.*'s dataset; on a temporally separated, hold-out test set consisting of 1000 structures, both the teacher and student models perform excellently (Figure 4.7a). Concretely, compared to the test-set DFT labels, the fine-tuned `MACE-MP0-b3` model reaches $R^2 = 0.99954$ for its per-atom energy predictions, and $R^2 = 0.99914$ for its force predictions. Despite never being explicitly trained on any DFT labels, the performance of this model is near-perfectly preserved upon distillation! The student `PaiNN` model reaches $R^2 = 0.99950$ for its per-atom energy predictions, and $R^2 = 0.99900$ for its force predictions, against the same DFT baseline.

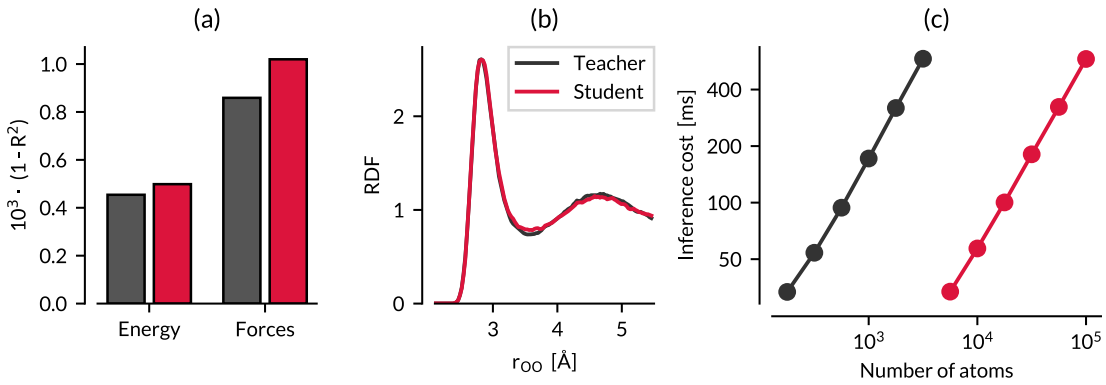


Figure 4.7: *comparing teacher (fine-tuned* `MACE-MP0-b3`*) and student (*`PaiNN`*) models performance. (a) both models display near perfect correlation with the DFT labels on a hold-out test set. (b) the student near-perfectly replicates the teacher's O-O partial radial distribution function. (c) the distilled model is approximately 25× faster for a given system size than the fine-tuned model.*

Beyond numerical validation, the student model also near-perfectly preserves the behaviour of the teacher model in MD simulations. In Figure 4.7b, I plot the O-O partial radial distribution function (RDF) as averaged over 100 ps of NVT simulation at 300 K driven by the teacher and student models separately.

Finally, due to its much smaller parameter count and reduced radial cutoff, the distilled model is approximately 25× faster for a given system size (Figure 4.7c). Alternatively, the distilled model can be used to simulate larger systems in the same amount of time as the fine-tuned model (again by a factor of 25×).

4.6. Ablation studies

Before applying this automated distillation pipeline to other foundation models and chemical systems (see Chapter 4.8), I perform a series of ablation studies to better understand the impact of each component of the distillation process.

In all cases, unless otherwise stated, I use the fine-tuned `MACE-MP0-b3` model as the teacher, and distil into `PaiNN` models with the same hyper-parametrisation as above.

4.6.1. Amount of synthetic data

The first obvious ablation axis to probe in this set-up is the amount of synthetic data used for the distillation. In Figure 4.8, I use increasing amounts of synthetic data to train the student model, and validate against both the teacher- and DFT-generated force labels on the hold-out test set.

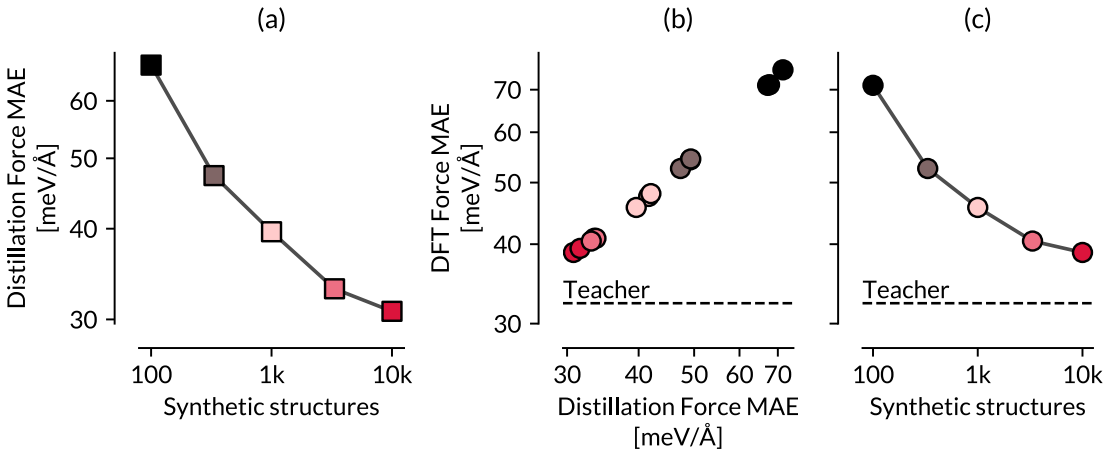


Figure 4.8: *student model performance against both teacher and DFT labels as a function of increasing amounts of synthetic data.*

As expected, training on more synthetic data leads to a reduction in the distillation error, *i.e.* reduces the difference between the student and teacher models' predictions (Figure 4.8a).

As demonstrated above, the teacher model is itself closely aligned with the ground truth, DFT labels. Hence, lowering the distillation error leads to student models that are more closely aligned with the DFT ground truth (Figure 4.8b).

Finally, therefore, increasing the amount of synthetic training data leads to a reduction in the student model's error against the DFT labels, asymptotically approaching that of the teacher model (Figure 4.8c).

4.6.2. Size of the student model

Atomistic foundation models require large parameter counts to be able to accurately model the PES for a wide range of systems. When targeting specific systems, however, MLFFs can achieve strong performances with relatively small parameter counts, due to the much lower information content of the PES in restricted regions.

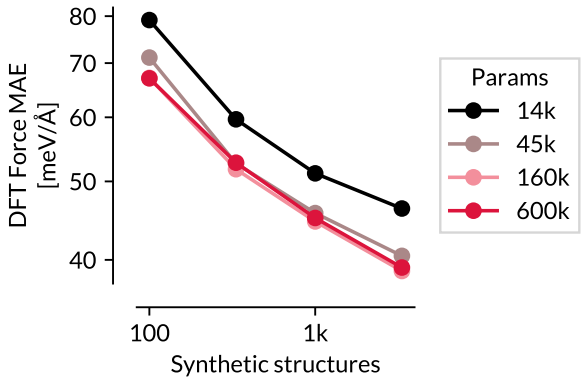


Figure 4.9: *student model performance quickly saturates with increasing parameter count.*

In Figure 4.9, I present learning curves for student models with varying parameter counts. I find that parameter counts beyond 50k lead to no discernible improvement in performance. A very small model (14k parameters) does perform worse, but accuracy continues to improve with dataset size, indicating that this is not necessarily a capacity problem.

This is a clear demonstration of the bounded information available for the model to learn from, and also provides a counter-example to common scaling law behaviour whereby accuracy increases monotonically with increasing model size.

4.6.3. Radial cutoff

I claimed above that the accurate modelling of many systems does not require the large radial cutoffs used by modern foundation models. To demonstrate this, I ablate the radial cutoff of the student model, from 2.5 to 5.5 Å, in Figure 4.10.

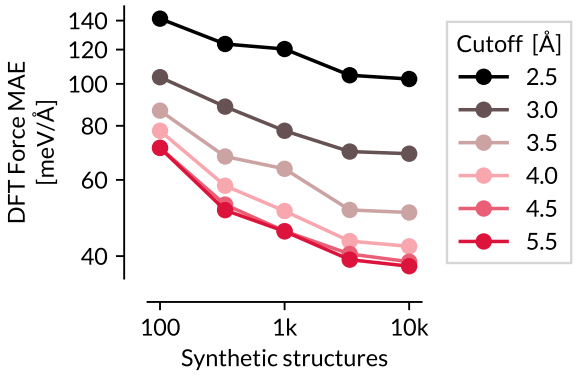


Figure 4.10: *DFT-labelled test-set force MAE for student models with increasing radial cutoffs.*

Models of all cutoffs display the same behaviour, namely a log-log linear decay in force MAE with amount of synthetic data up to ≈3,000 structures. Increasing the

cutoff from 2.5 to 4.5 Å makes the gradients of these lines steeper, and lowers them. Interpreting this, models with larger cutoffs are (a) more accurate for a given amount of synthetic data, and (b) are more efficient at learning from additional data.

Beyond 4.5 Å, I observe no further improvements in accuracy, suggesting that the interactions that the teacher model is capturing, and that the student models are learning, are significantly shorter ranged than the teacher model's cutoff of 6 Å.

4.6.4. Student model architecture

A compelling advantage of my proposed distillation pipeline is that both components of the framework (automated PES exploration, and vanilla training of the student model) are completely architecture-agnostic. I am therefore free to use any student model architecture, even beyond GNN based models.

To demonstrate, in Figure 4.11 I present `TensorNet`[95] (50k params) and `ACE`[109] (1k params) student models, trained using `graph-pes`[57] and `pacemaker`[225] respectively.

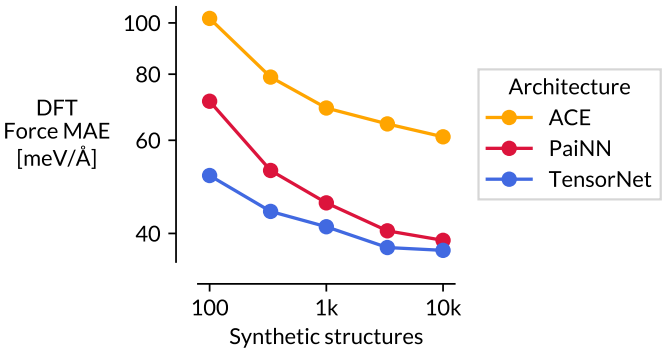


Figure 4.11: *learning curves for student models of different architectures.*

The `TensorNet` models are ≈1.5× slower than the `PaiNN` models, while the `ACE` models are ≈4× faster. Nevertheless, `PaiNN` seems the most useful architecture here: while `TensorNet` models are more accurate for a given amount of synthetic data, they appear to be plateauing to the same accuracy as the `PaiNN` models. While `ACE` models are faster than `PaiNN` models, their poor numerical accuracy makes them less suitable for driving MD simulations, in this case leading to significant over-ordering of the liquid state (Figure 4.12).

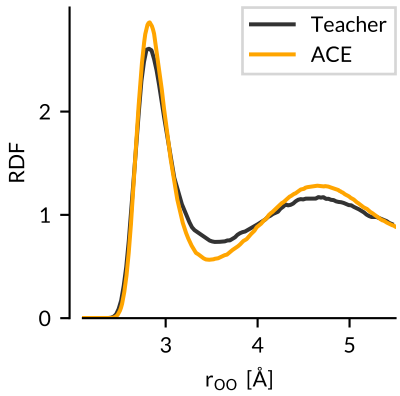


Figure 4.12: *O-O partial radial distribution function (RDF) as averaged over 100 ps of NVT simulation at 300 K driven by the teacher and student models separately. The ACE student model predicts significant over-ordering of the liquid state compared to the teacher*

4.7. Scaling laws for distillation

In Figure 4.8, I showed that increasing the amount of synthetic data reduces the distillation error, and that this in turn reduced the student model's error with respect to the ground truth DFT labels, *i.e.* the error we ultimately want to minimise. Here, I investigate this relationship more closely, and attempt to understand the scaling laws for synthetic distillation.

Consider the following three error terms:

1. **the teacher model's error**: $\varepsilon_T = \hat{F}_T - F_{\text{true}}$
   i.e. the error between the teacher model's force predictions and the ground truth forces.
2. **the student model's error**: $\varepsilon_S = \hat{F}_S - F_{\text{true}}$
   i.e. the error between the student model's force predictions and the ground truth forces. This is the error we ultimately want to minimise.
3. **the distillation error**: $\varepsilon_D = \hat{F}_S - \hat{F}_T$
   i.e. the error between the student model's force predictions and the teacher model's force predictions. This is the error we actually minimise during distillation.

These three terms are related by:

$$\varepsilon_S = \varepsilon_T + \varepsilon_D$$

which displays correct limiting behaviour: as the teacher model's error approaches zero, the student model's error approaches the distillation error. Similarly, as the distil-

lation error approaches zero, the student model's error approaches the teacher model's error.

From standard variance propagation identities, we can obtain:

$$\mathrm{Var}(\varepsilon_S) = \mathrm{Var}(\varepsilon_T) + \mathrm{Var}(\varepsilon_D) + 2 \cdot \mathrm{Cov}(\varepsilon_T, \varepsilon_D)$$

The MAE error metric is linearly related to the square root of the variance of the corresponding error term. Assuming independence of the error terms, *i.e.* that

$$\mathrm{Cov}(\varepsilon_T, \varepsilon_D) = 0$$

we would therefore expect:

$$\mathrm{MAE}(\varepsilon_S) = \sqrt{\mathrm{MAE}(\varepsilon_T)^2 + \mathrm{MAE}(\varepsilon_D)^2}$$

In Figure 4.13, I overlay this expected error scaling law (solid black line) on top of the empirical data for a range of `PaiNN`, `TensorNet` and `ACE` models trained on the synthetic dataset, with:

$$\mathrm{MAE}(\varepsilon_T) = \lambda = 32.3 \text{ meV/Å}$$

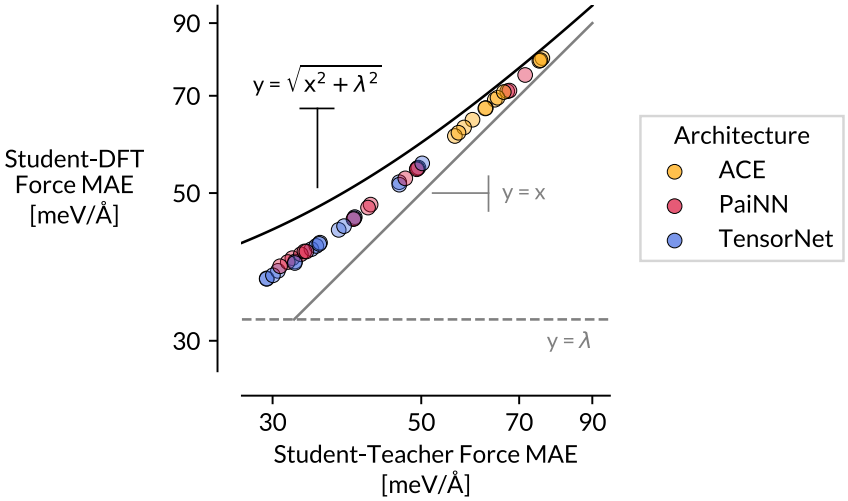


Figure 4.13: *dependence of the final student model error (vs DFT) on the intermediate distillation error (vs teacher).*

Two immediate observations are:

1. there is a clear, architecture-independent, empirical scaling law. Whatever is driving the relationship between the distillation and student error terms is not related to the model architecture.
2. the expected scaling law is incorrect! The student models universally perform better than expected on the DFT test labels for a given distillation error.

To understand the breakdown of the expected scaling law, I analyse the distribution of $(\varepsilon_T, \varepsilon_D)$ pairs in Figure 4.14 for the most accurate `PaiNN` model.

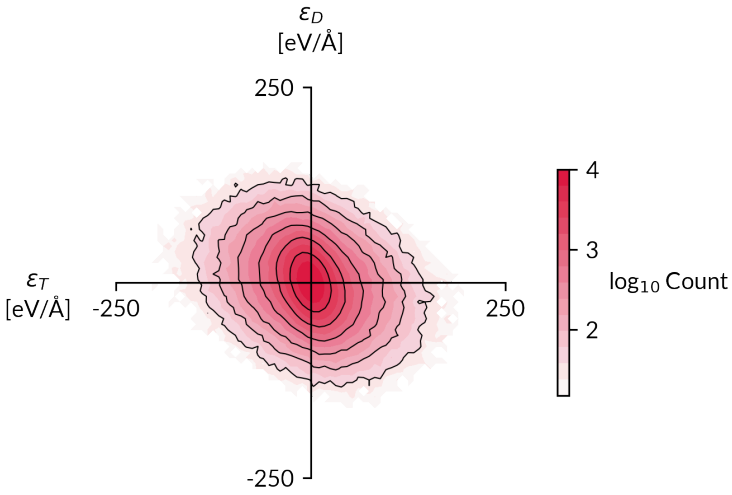


Figure 4.14: *distribution of $(\varepsilon_T, \varepsilon_D)$ pairs for the most accurate `PaiNN` model. A clear negative correlation is present.*

Intriguingly, I find that these error terms are not independent! Instead, there is a clear negative correlation (here Pearson's correlation coefficient is -0.24). Interpreting this: in cases where the teacher model over-predicts the DFT force label ($\varepsilon_T > 0$), the student model tends to under-predict the teacher model's label ($\varepsilon_D < 0$), reducing the student model's error *vs* the ground truth (ε_S).

Overall, this lack of independence acts to improve the student model's error relative to the ground truth, reducing the amount of synthetic data required to reach a given accuracy than would naively be expected. The origin of this correlation is not immediately obvious, but appears to be strongest at larger force magnitudes, and hence may be related to the often observed "over-softening" of foundation MLFFs.[226]

4.8. Applications

To demonstrate its general applicability beyond modelling bulk liquid water, I apply my proposed automated distillation pipeline to the following systems, together with assistance from the following collaborators:

- **under-dense amorphous silica**: *Louise AM Rosset* ran MD simulations, and generated averaged structure factors for analysis
- **metallic hydrogen**: *Chiheb Ben Mahmoud* ran MD simulations and fitted equations of state for analysis
- **hybrid halide perovskites**: *Laura-Bianca Paşca* ran MD simulations.

In all cases, I fine-tuned foundation models, generated synthetic data using `augment-atoms`, and distilled into student models using `graph-pes`.[57]

4.8.1. Metallic hydrogen

At extreme pressures and temperatures (>10,000 K and >400 GPa), such as those in the cores of both Jupiter and Saturn,[227] elemental hydrogen behaves as a bulk metallic liquid, with no discernible H_2 units.[228]

Studying this system experimentally is extremely challenging: in fact, while the existence of this phase was first proposed by *Eugene Wigner* in 1935,[229] it was not until 1996 that it was verified experimentally![230] Computational studies that circumvent the difficulty and expense of physical experiments are therefore of great use and interest. One important property is the "isotherm", *i.e.* the pressure–density equation of state at a given temperature. Generating these values via simulation involves running long MD simulations under the NpT ensemble, at very high temperatures and pressures, and using a very small time step (*e.g.* 0.01 fs). This makes modelling very challenging.

In one successful study of this system, *Ben Mahmoud et al.*[231] developed a hand crafted MLFF capable of modelling these isotherms, using a dataset of 28,000 DFT-labelled structures and the GAP framework.[7] More recently developed foundation models struggle to recapitulate the performance of this specialised MLFF model for two reasons:

1. **extreme conditions are often OOD**:
   To date, most large scale materials datasets are based around known, ground-state structures of common materials. Comparatively few foundational MLFFs have seen appreciable training data at the very high pressures and temperatures required by this study, and so most models generate completely unphysical predictions. Two exceptions to this are the `MatterSim`[146] and `PET-MAD`[232] models.
2. **high number densities are computationally expensive**:
   As discussed above, foundation models typically require large parameter counts and radial cutoffs. Metallic hydrogen has a very high number density, meaning abnormally large inference costs to model even modest numbers of atoms. This means MD simulations for small systems are slow, and extending to required larger systems is impossible on current hardware when using a single machine.

Metallic hydrogen is therefore a prime candidate for the distillation approach: a few existing foundation models have the required accuracy for modelling this system, but none are fast enough to be practically useful.

To reach chemical accuracy, I take the `MatterSim-v1-5M`[146] model, and fine-tune on 100 configurations from the dataset of *Ben Mahmoud et al.*,[231] with densities in the range [0.6, 1.6] g/cm³. These structures were also used as seeds to generate a total of 1,000 synthetic structures, using `augment-atoms` with default settings and a temperature of 20,000 K.

To distil, I trained a 50k parameter `PaiNN` student model with a radial cutoff of 2 Å on this dataset. Of particular interest here is the fact that the student model is more than 2 orders of magnitude faster (!) at inference than the teacher model, due to the 96% reduction in parameter count, and the 15× smaller effective cutoff radius.

In Figure 4.15, we validate the student model's performance by using NpT MD simulations to generate "isotherms" for temperatures between 10,000 and 50,000 K, and comparing to the (extremely expensive) *ab initio* data of *Ben Mahmoud et al.*[231] The teacher model was (an order of magnitude) too expensive to validate in this manner, and so I do not include it for comparison.

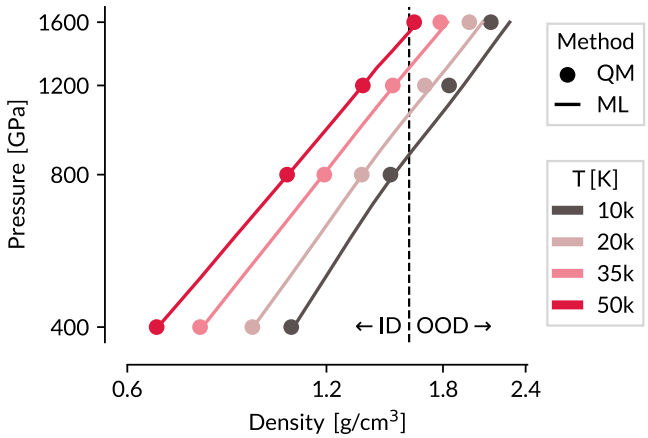


Figure 4.15: *isotherms for metallic hydrogen, generated by using* ab initio *(QM) and student models (MLP) to drive NpT MD simulations*

For the density range over which we fine-tuned and distilled, [0.6, 1.6] g/cm³, the student model near-perfectly recapitulates the *ab initio* equations of state. However, at higher densities, the student model begins to under-predict the equilibrium pressure.

This highlights an important limitation of all model distillation approaches: unlike their general purpose teachers, distilled student models are only capable of correctly modelling the PES for the domain on which they were trained.

4.8.2. Porous amorphous silica

Porous amorphous silica (a-SiO_2), when formed as an aerogel,[233] provides exceptionally low thermal conductivity while remaining mechanically stiff, lightweight, and strong.[234] At the atomic level, the structure comprises a network of vertex-sharing SiO_4 tetrahedra, with very large pores (up to 1,000 Å in diameter) leading to these desirable properties, as well as high internal surface areas for catalytic applications.[235] These pores also make modelling challenging: in order to simulate realistic meso-porous structures, MLFFs need to model both surface and bulk regions accurately.

In developing a fast and scalable model to target this system I:

1. fine-tune the `orb-v3-direct-20-omat`[143] model to reach chemical accuracy using 25 *ab initio* labelled structures from the dataset developed by *Ben Mahmoud et al.*, 2025.[236] These structures contain pores up to a maximum of 10 Å in diameter.
2. use `augment-atoms` with default settings to generate 200-structure family trees for each of the 25 training structures, and then combine these into a single, 5000-structure synthetic dataset.
3. distil into a `PaiNN` model with a radial cutoff of 4 Å, and 50k parameters.

On the established hold-out test set, the student model performs near-perfectly with respect to the *ab initio* labels (Figure 4.16). Of particular interest here is that the teacher model is neither smooth (by only considering the 20 nearest neighbours for each atom), nor conservative (by making direct force predictions). In contrast, the student model is both smooth and conservative, and yet can still learn a very accurate model of the PES!

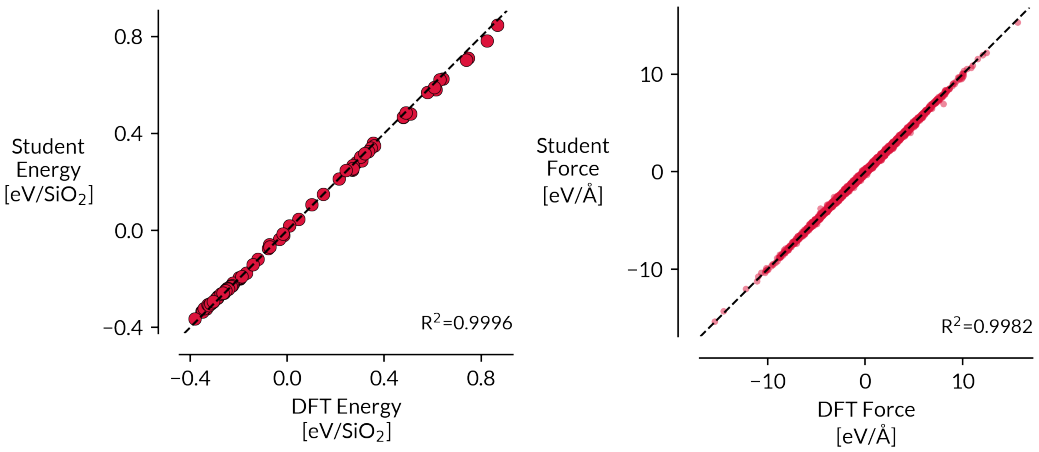


Figure 4.16: *energy and force parity plots for a `PaiNN` student of a fine-tuned `Orb` model on a hold-out test set of 84 meso-porous silica structures*

Beyond numerical validation, we ran MD in the NpT ensemble (as implemented in `ASE`[147] using a Berendsen barostat[237]) at 500 K and for 10 ps to equilibrate low-density (<1.2 g/cm³) meso-porous silica structures prepared using an existing, general purpose SiO_x model.[128] To compare these models, I plot the structure factor for the final equilibrated configurations, as averaged over 4 independent runs in Figure 4.17.

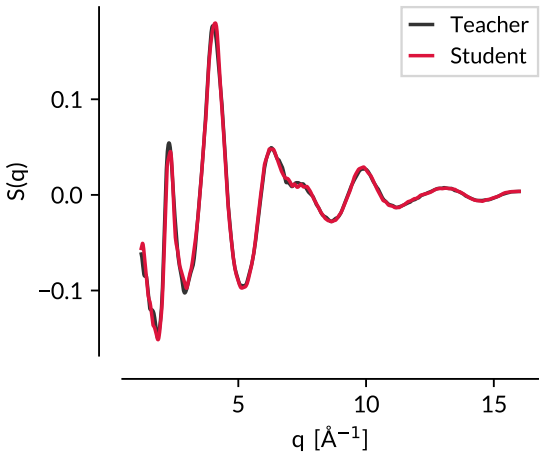


Figure 4.17: *structure factors for meso-porous silica structures equilibrated using the teacher and student models*

The structure factor for structures generated by the student `PaiNN` model shows excellent agreement with those from teacher `Orb` model, and both are in excellent agreement with equilibrations performed using an existing, general purpose SiO_x model.[153]

I note again that when pushed OOD, the student model's performance breaks down: when targeting structures with very large pores (*i.e.* at densities lower than 1 g/cm^3), the student model starts to generate unphysical structures in MD.

4.8.3. Hybrid halide perovskites

Hybrid halide perovskites emerged in the early 2010s as promising materials for thin-film photovoltaics.[238] The canonical representative, methylammonium lead iodide (MAPI), has since been the focus of many experimental[239,240] and computational[241,242] studies aimed at understanding its dynamic structure and electronic properties. While MAPI can adopt different polymorphs depending on temperature, the cubic form is the most relevant for the operation of solar-cell devices.[243]

I use my proposed pipeline to generate a model for this system by (i) fine-tuning `MACE-MP0-b3` on 25 structures taken from the dataset of *Bokdam*,[244] as labelled by *Fykouras et al.*[245] with the meta-GGA SCAN functional,[246] (ii) generating a total of 5,000 synthetic structures using `augment-atoms` with default settings, and (iii) distilling into a `PaiNN` model using `graph-pes`.

Both the teacher and student models perform well on the temporally separated, DFT-labelled test set (29 and 32 meV/Å component-wise force RMSEs, respectively). For further validation, we ran constant-pressure MD simulations (1 ns, 350 K) on a 3×3×3 cubic supercell. The equilibrium pseudocubic lattice parameters during the 350 K run ranged between 6.3 and 6.4 Å for both the teacher and student models, in agreement with experimental[247] and computational[248] studies at this temperature.

To further probe the success of the distillation, in Figure 4.18 I compare the distribution of cation orientations within the unit cell during the MD trajectories, projecting the C→N bond vector onto polar coordinates.

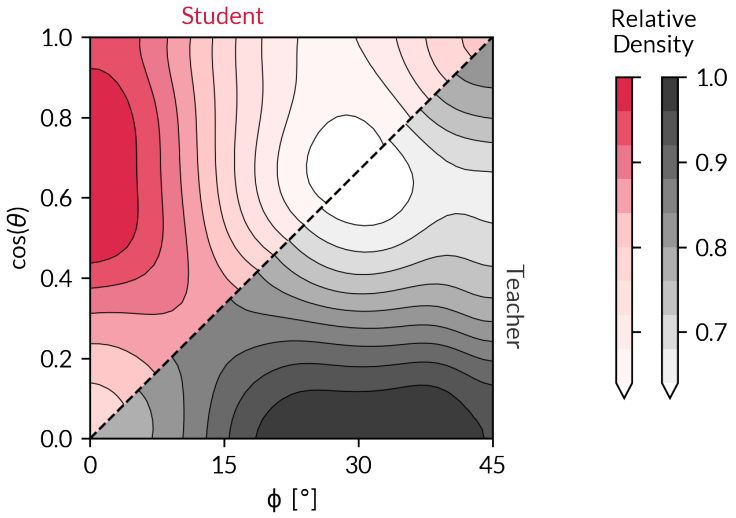


Figure 4.18: *distribution of MAPI cation orientations within the unit cell during MD driven by the teacher (black) and student (red) models*

The distribution is nearly identical for both models, expressing mild preference for alignment along one axis, while being skewed at ≈30° to the other. Again, this is in keeping with other computational studies of MAPI.[248]

4.9. Summary and outlook

Atomistic foundation models are revolutionising the field of computational materials modelling. While their zero-shot performance is impressive, and will no doubt continue to improve, a majority of their utility arises from the ease with which they can be fine-tuned to reach chemical accuracy on new systems.

Unfortunately, existing foundation models are big and slow. While much research effort is being spent on accelerating inference, current trends in parameter count and radial cutoffs suggest that foundation models and their fine-tuned counterparts will continue to become more, not less, expensive to use in practice.

A very real need therefore exists for methods that can automatically distil MLFFs into smaller, faster models for use on a targeted domain. The `augment-atoms` approach I have presented above is one such method. Among others, key properties I have demonstrated are that this approach (i) can be applied to a wide range of systems, with no need for expert domain knowledge, (ii) produces small, accurate and fast student models, (iii) is quick to perform, and hence the expense of generating a student is quickly amortised upon use in simulation, and (iv) is completely agnostic to the architecture

of both the teacher and student models, allowing researchers to pick and choose those most suitable for their application.

Looking forward, existing trends suggest that foundation models will continue to become more accurate and generally applicable, and that the speed and accuracy of smaller models will also continue to improve. The `augment-atoms` approach to distillation benefits from improvements to both of these areas, and so is future proofed.

Training details

Foundation model fine-tuning: in all cases, I used `graph-pes` to fine-tune the foundation models with a learning rate of 10^{-4}, the maximum batch size that fitted on a single NVIDIA A6000 GPU (typically 5–15 structures), equally weighted `PerAtomEnergy` and `Force` MAE components, the `Adam` optimiser, a `ReduceLROnPlateau` scheduler with factor of 0.8 and patience of 25 epochs, and an early stopping criterion of 100 epochs.

Student model training: I used `graph-pes` to train all GNN-based student models presented above. In all cases, I incorporate a ZBL core repulsion term,[141] and train on an additional 50 dimer structures for each unique pair of elements in the training set, with spacings sampled uniformly in the range $[0.6d, 3d]$ Å, where d is the sum of the van der Waals radii of the two atoms. These steps acts to ensure correct limiting behaviour of the model. Unless otherwise specified, I used a 4.5 Å radial cutoff, 50k parameters, a fresh `Adam` optimiser with a learning rate of 10^{-3}, a `ReduceLROnPlateau` scheduler with factor of 0.8 and patience of 25 epochs, and maximised the batch size to fit on a single NVIDIA A6000 GPU (typically 100–400 structures).

ACE models: *Daniel Thomas du Toit* trained all `ACE` models presented above. He used his `XPOT`[249] framework to optimise the hyper-parameters on a dataset of 333 structures, relying on `pacemaker`[225] to perform each fit. He used a fresh `AdamW` optimiser with a batch size of 32, $\kappa = 0.75$, a ZBL repulsion term, an early stopping patience of 50 epochs, and a final-stage, 50 epoch `BFGS` "upfitting" optimisation with $\kappa = 0.01$ to improve energy-prediction accuracy.

Attribution and Acknowledgements

The above chapter presents work from my joint-first-author pre-print (currently under peer review):

> Distillation of atomistic foundation models across architectures and chemical domains
> *John L.A. Gardner,* D.F. Thomas du Toit,* C. Ben Mahmoud, Z. Faure Beaulieu, V. Juraskova, L.-B. Paşca, L.A.M. Rosset, F. Duarte, F. Martelli, C.J. Pickard, V.L. Deringer*,
> arXiv preprint, 2025, arXiv:2506.10956

D.F.T.d.T and I designed the synthetic distillation strategy.
Together with V.L.D, we co-ordinated the project.
I conceived the synthetic data generation protocol, fine-tuned foundation models, generated synthetic data, and trained the graph-network models.
D.F.T.d.T trained all ACE models
Applications discussed above were contributed by C.B.M (high-pressure hydrogen), L.A.M.R (silica), and L.-B.P (MAPI), respectively.

5. End-to-end learning of symbolic force fields

Compared to MLFFs, empirical PES models have many desirable properties, including:

- **interpretability**: empirical models break down the total energy into a sum over simple terms, the form and parameters of which can be easily interpreted and understood via direct inspection.
- **limiting guarantees**: the physically inspired functional forms of empirical models ensure that they recover qualitatively correct behaviour under important limiting conditions, including at both very small and very large interatomic distances.
- **speed**: empirical models require orders of magnitude fewer FLOPs to evaluate than NN-based MLFFs, allowing them to scale to larger systems and longer time scales.

The major limitation of empirical PES models is their (greatly) reduced accuracy *vs* both MLFFs and ground-truth *ab initio* labels. In general, these high errors stem from a combination of:

1. many-bodied interactions are partially or completely missing in empirical models
2. the functional forms for the existing interactions are inaccurate, even with optimal parameters.

In many systems of practical interest, the second source of error dominates: many-bodied energy contributions are small in magnitude, and the true PES can be approximated well as a sum of simple, local terms. In these cases, improvements to the functional forms of individual terms can (dramatically) improve accuracy, yielding fast, highly useful models.

In this chapter, I present methodology to learn both the functional form and parametrisation of empirical PES models, using the representational power of MLPs to learn accurate low-body-order energy contributions, before converting these into closed form expressions via symbolic regression.

5.1. Simple, separable energy models

The total potential energy of a system can, with complete generality, be decomposed into a sum over interactions of increasing body order:

$$E(\{\vec{r}_1, ..., \vec{r}_N\}) = \sum_{i \neq j} E_{2b}(\vec{r}_i, \vec{r}_j) + \sum_{i \neq j \neq k} E_{3b}(\vec{r}_i, \vec{r}_j, \vec{r}_k) + ...$$

where the explicit dependence on the atomic numbers, $\{z_i\}$, has been dropped for simplicity of notation.

These terms become increasingly expensive to evaluate as the body order increases. Concretely, for a system composed of N atoms, the number of summands to consider for the term with body order v is:

$$\prod_{i=1}^{v}(N - i + 1) = \frac{N!}{(N - v)!} \quad : \quad O(N^v)$$

This is extremely expensive!

Linear scaling with system size can be recovered by only considering *local* interactions (see the discussion in Chapter 2). Most typically, this is achieved by the incorporation of an envelope function, $\phi_c(r)$, that (smoothly) decays to zero at and beyond some cutoff radius, c. Concretely, multiplying each energy term by an appropriate collection of envelope functions:

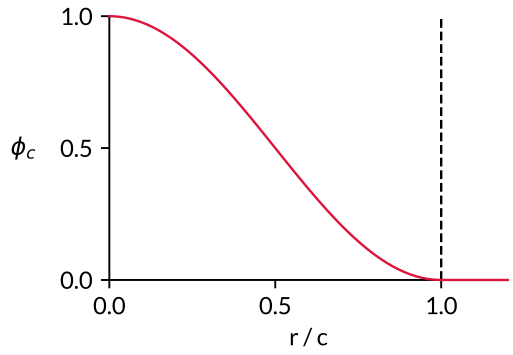
$$\begin{aligned} E = & \sum_{i \neq j} E'_{2b}(\vec{r}_i, \vec{r}_j) \cdot \phi_c(r_{ij}) + \sum_{i \neq j \neq k} E'_{3b}(\vec{r}_i, \vec{r}_j, \vec{r}_k) \cdot \phi_c(r_{ij}) \cdot \phi_c(r_{ik}) + ... \\ \equiv & \sum_{\substack{i \neq j, \\ r_{ij} < c}} E'_{2b}(\vec{r}_i, \vec{r}_j) \cdot \phi_c(r_{ij}) + \sum_{\substack{i \neq j \neq k, \\ r_{ij} < c, \\ r_{ik} < c}} E'_{3b}(\vec{r}_i, \vec{r}_j, \vec{r}_k) \cdot \phi_c(r_{ij}) \cdot \phi_c(r_{ik}) + ... \end{aligned}$$

recovers linear scaling with system size by forcing most terms to zero, ensuring that the number of non-zero summands to consider for a term with body order v reduces to:

$$N(\rho c^3)^{v-1} \quad : \quad O(N)$$

where ρ is the mean number density, c is the cutoff radius, and ρc^3 is therefore the average number of neighbours per atom. A common[6,95,119] choice of envelope function, and the one I use throughout this chapter, is the following cosine term:

$$\phi_c(r) = \begin{cases} \frac{1}{2} \cdot \left[1 + \cos\left(\frac{\pi r}{c}\right)\right] & r < c \\ 0 & r \geq c \end{cases}$$



Even when restricting the number of summands via the locality assumption, explicit consideration of $v \geq 4$ terms is prohibitively expensive. Taking inspiration from the embedded atom model (EAM),[47] we can *approximate* 4-body and higher (*i.e.* many-body) effects by considering functions that act on a *local density field*, ρ_i:

$$E_{\text{many-body}} \approx E_{\text{EAM}} = \sum_i f(\rho_i) \quad \text{where} \quad \rho_i = \sum_{j \neq i} g(\vec{r}_i, \vec{r}_j) \cdot \phi_c(r_{ij})$$

Providing that f is non-linear, this term can approximate many-bodied effects in an interpretable manner. I therefore propose a simple, interpretable, fast, and linearly scaling PES architecture as a sum of three components:

$$E = E_{\text{2B}} + E_{\text{3B}} + E_{\text{EAM}}$$

Below, I detail a construction for each of these terms, using MLPs as learnable components. Throughout, I act only on scalar terms (distances and angles) to guarantee invariance in the energy predictions with respect to translations and rotations. I use `CELU` activations which combine with the envelope function to guarantee that the model is smooth (and in fact infinitely differentiable) everywhere. Finally, I make use of expansion functions, as can be found in nearly all MLFF architectures, to convert scalar terms into collections of features as inputs to MLPs.[6,8,10,93–95]

5.1.1. Two-body term

Two-body models (*i.e.* pair potentials, such as the Lennard-Jones,[250] Buckingham,[48] and Morse[49] models) make energy predictions as a sum over pairs of atoms:

$$E_{\text{2B}} = \sum_{i \neq j} f_c(\vec{r}_i, \vec{r}_j, z_i, z_j) \cdot \phi_c(r_{ij})$$

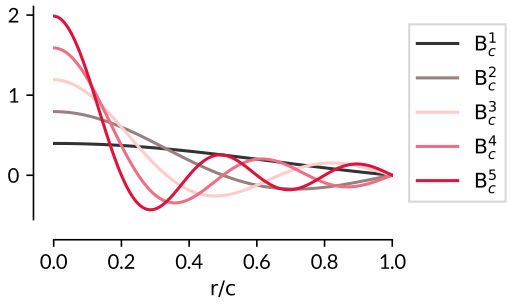
To learn such a function using MLPs, I propose the following construction for f:

$$f_c(\vec{r}_i, \vec{r}_j, z_i, z_j) = \text{MLP}^{z_i, z_j}_{[h \to \ldots \to 1;\ \texttt{CELU}]}({}^{h}B_c[r_{ij}])$$

where

- $r_{ij} = \|\vec{r}_j - \vec{r}_i\| \in \mathbb{R}^+$ is the scalar distance between atoms i and j,
- ${}^{h}B_c[r] \in \mathbb{R}^h$ is a vector of features of the scalar distance r comprising the first h terms of a Bessel expansion, with term $1 \leq n \leq h$ given as:

$$B_c^n[r] = \sqrt{\frac{2}{c}} \cdot \frac{\sin\left(n\pi \frac{r}{c}\right)}{r}$$



- $\text{MLP}^{z_a,z_b}_{[h\to\ldots\to 1;\ \texttt{CELU}]}$ is an MLP responsible for modelling interactions between atoms with atomic numbers z_a and z_b, with h inputs, a scalar output, and `CELU` activations.

5.1.2. Three-body term

Three-body terms make energy predictions as a sum over triples of atoms:

$$E_{\text{3B}} = \sum_{i\neq j\neq k} f_c(\vec{r}_i, \vec{r}_j, \vec{r}_k, z_i, z_j, z_k) \cdot \phi_c(r_{ij}) \cdot \phi_c(r_{ik})$$

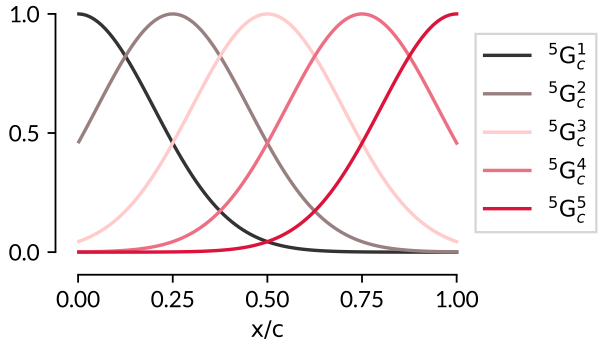
To learn such a function using MLPs, I propose the following construction for f:

$$f_c(\vec{r}_i, \vec{r}_j, \vec{r}_k, z_i, z_j, z_k) = \\ \text{MLP}^{z_i,z_j,z_k}_{[(h_m+h_d+h_\theta)\to\ldots\to 1;\ \texttt{CELU}]}\left({}^{h_m}B_c\left[m_{ij,ik}\right], {}^{h_d}G_{c^2}\left[d_{ij,ik}\right], {}^{h_\theta}G_\pi\left[\theta_{jik}\right]\right)$$

where:

- $m_{ij,ik} = \frac{r_{ij}+r_{ik}}{2} \in \mathbb{R}^+$ is the mean length of the two edges spanned by the atoms (i,j) and (i,k) respectively,
- $d_{ij,ik} = \left(r_{ij} - r_{ik}\right)^2 \in \mathbb{R}^+$ is the square of the difference of these lengths
- $0 \leq \theta_{jik} \leq \pi$ is the angle spanned by $\vec{r}_{ij}$ and $\vec{r}_{ik}$,
- ${}^hG_c[x] \in \mathbb{R}^h$ is a vector of features of the scalar value x comprising the first h terms of a Gaussian expansion, with term $0 \leq n < h$ given as:

$${}^hG_c^n[x] = \exp\left(-\frac{\left(x - \frac{n}{c\cdot(h-1)}\right)^2}{2\cdot\left(\frac{c}{h}\right)^2}\right)$$



- h_m, h_d, and h_θ are the number of features to include in the Bessel and Gaussian expansions for $m_{ij,ik}$, $d_{ij,ik}$, and θ_{jik} respectively.

The symmetrising coordinate transform: $(r_{ij}, r_{ik}) \to (m_{ij,ik}, d_{ij,ik})$ ensures permutation invariance of the non-central atoms, *i.e.* that $f(i,j,k) = f(i,k,j)$, while ensuring model outputs remain smooth as r_{ij} travels over $(r_{ik} - \delta) \to (r_{ik} + \delta)$.

5.1.3. Many-body term

EAM terms can approximate simple many-body effects:

$$E_{\text{EAM}} = \sum_i f(\rho_i, z_i) \quad \text{where} \quad \rho_i = \sum_{j \neq i} g_c(\vec{r}_i, \vec{r}_j, z_i, z_j) \cdot \phi_c(r_{ij})$$

To learn such a function using MLPs, I propose the following construction:

$$g_c(\vec{r}_i, \vec{r}_j, z_i, z_j) = \text{MLP}_{[(h_r+2h_z)\to\ldots\to h_\rho;\ \texttt{CELU}]}\left({}^{h_r}B_c[r_{ij}], {}^{h_z}\zeta[z_i], {}^{h_z}\zeta[z_j]\right)$$

$$f(\rho_i, z_i) = \text{MLP}^{z_i}_{[h_\rho\to\ldots\to 1;\ \texttt{CELU}]}(\rho_i)$$

where:

- ${}^{h}\zeta[z] \in \mathbb{R}^h$ is a vector of learned embedding features for atomic number z
- $\rho_i \in \mathbb{R}^{h_\rho}$ is some learned (scalar if $h_\rho = 1$, vector if $h_\rho > 1$) representation of atom i's local environment.

Combinations of these three terms lead to models with a partially interpretable breakdown of the total energy into two-, three-, and many-body contributions. However, the use of black-box MLPs means that the inner workings of each component are hidden. In the following section, I summarise methodology from the literature that can be used to convert black-box models into closed form expressions, allowing me to recover fast and completely interpretable symbolic PES models.

5.2. Symbolic regression

For centuries, scientists have been *manually* fitting functional forms to empirically observed data. Beyond their use as merely predictive tools, the resulting models have helped us to understand the true nature of the world around us and the physical laws that govern it. Perhaps the most impressive and important of these feats (and certainly my personal favourite) is *Johannes Kepler's* discovery of the three laws of planetary motion in 1609, relying on data collected over more than 20 years by *Tycho Brahe*, and leading directly to *Isaac Newton's* law of universal gravitation.[251–253]

Fitting non-obvious functional forms to large amounts of data is an extremely time-consuming and non-scalable process: finding a good fit requires not only searching over the combinatorially large, and non-differentiable, space of possible functional forms, but also then optimising over a loss-landscape of free-parameter values that is unique to each functional form. Despite this difficulty, a large family of "symbolic regression" methods have been developed that can successfully automate this process when searching over functions of small numbers of inputs.[254–258]

Most commonly, these approaches rely on genetic search, mimicking the process of natural selection to "evolve" an increasingly accurate "population" of candidate

models over time. Functional forms are encoded as abstract syntax trees (ASTs), with internal nodes representing mathematical operations, and leaf nodes representing variables and constants. At each "generation":

1. each equation is evaluated on the training data to assess its "fitness", a scalar value that incorporates both the loss on the training data and a term that penalises the equation's complexity (Occam's razor)
2. equations are stochastically selected for reproduction, such that the most fit equations have a higher probability of being selected.
3. new equations are created by mutating existing models (*i.e.* changing a node or sub-tree of the AST), and via crossover between two parent models, mimicking asexual and sexual reproduction respectively.

Running such a process for many generations yields a final population of candidate models, from which a Pareto front of models trading off complexity for accuracy can be extracted.

Below, I use *Miles Cranmer's* `PySR` package[255] to perform symbolic regression (see Appendix C for further details on the algorithm and configuration choices made in this work). As well as providing parallelised genetic search, this package also directly optimises parameters in a separate stage via gradient descent using the `BFGS` optimiser, dramatically accelerating convergence to highly accurate models.

Work to optimise the search process within symbolic regression is ongoing, with new approaches now using LLMs as a prior to guide the search process.[259] Despite the limitations and expense of existing symbolic regression frameworks, successful applications of `PySR` include the (re-)discovery of simple, pairwise, physical laws via the distillation of GNN components,[260] equations of motion for supermassive black hole dynamics,[261] equation recovery and simplification for quantum electrodynamics via application to data from the Large Hadron Collider,[262] discovery of simple laws governing rates and volumes of international trade,[263] and approximate solutions to arbitrary ODEs via distillation of NN-based solutions.[264]

5.3. End-to-end learning of symbolic energy models

To learn a symbolic PES in an end-to-end manner, I propose and apply the following concrete steps:

1. **identify important terms for the system**:
   The relative importance of two-, three-, and many-body interactions varies between chemical systems. Metallic bonding can be well described solely by pairwise interactions. Covalent systems often require the directionality provided by three-body terms. Many-body effects can sometimes also be important. I therefore propose constructing simple models as an *appropriate* combination of E_{2B}, E_{3B}, and E_{EAM} terms.

2. **train the MLP-model via gradient descent**:
   Unlike modern, GNN-based MLFFs, these simple MLP-based models are extremely lightweight (<5k parameters), and require very little memory footprint for the forward pass. This allows for the use of full-batch optimisation using quasi-second order optimisers (below, I use the `PyTorch`[59] implementation of `L-BFGS`[265]) on datasets containing up to 1,000 structures, leading to dramatically faster convergence than standard first order optimisers such as `Adam`.[69]
3. **distil each MLP into a symbolic form**:
   Symbolic regression frameworks are computationally expensive. Rather than using all training data to distil each MLP, I sample a small number of input-output examples (500 unless otherwise stated) from a single forward pass over the complete training set. To preferentially fit to important parts of the function, I weight the sampled examples by the appropriate envelope function evaluation/s.
4. **re-optimise the symbolic model parameters**:
   Symbolic regression frameworks rarely find fully optimal parameter values for any given functional form. To improve final accuracy, I therefore re-optimise the parameters of the fully symbolic models, again using full-batch `L-BFGS` optimisation.

5.3.1. Related work: symbolic models

Beyond the methodology proposed above, two other notable approaches exist for learning symbolic PES models. In both cases, the authors have applied these approaches to learn the PES for bulk metallic copper.

Hernandez et al. (2019)[119] propose to perform a genetic search directly over PES forms, where the loss function is evaluated (i) over the entire training set, and (ii) without the use of automatic differentiation (*i.e.* using finite differences to estimate forces and stress tensors). Using a large compute cluster, their approach took ≈2 CPU-weeks to find two Pareto-optimal models. Below, I demonstrate that when using the same loss function and dataset, my approach can find simpler, more accurate models in just a few minutes and on a personal laptop, amounting to a $\approx 10^5\times$ speed-up.

More recently, *Varughese et al.* (2025)[266] proposed an "Equation Learner Neural Network" approach, wherein traditional activation function layers are replaced with functional nodes representing mathematical basis functions like addition, multiplication, and trigonometric operations. These models act on fixed sets of pairwise features (*e.g.* r^6, e^{-r} *etc.*) and are trained via a combination of Monte Carlo Tree Search[267] and gradient descent procedures. This training process is much faster than that of *Hernandez et al.*, but restricts the final symbolic forms to be weighted summations over

[i] *Varughese et al.* choose basis functions of $e^{-\lambda r}$ and r^{-n} for $n \in \{1, 6, 7, 8, 9, 10, 11, 12, 14\}$

a limited set of basis functions.[i] In contrast, the method I propose here can recover arbitrary symbolic forms, including those that are products of several terms.

5.3.2. Related work: body-order decomposition

Decomposing the total energy into contributions from different body-order terms is a natural way to model the hierarchy of scales present in most potential energy surfaces. In particular, two-body interactions tend to be largest in magnitude, and completely dominate at small distances where the Pauli exclusion principle leads to high energies and strong repulsion.[268] Various methodologies attempt to incorporate this hierarchy of scales into the functional form of MLFF models:

In the *Gaussian Approximation Potential* (GAP) framework,[7,58] two-, and three-body terms are often modelled explicitly, in addition to the SOAP-based many-body term. Terms of all body-order are fitted simultaneously, but require hand-tuned scaling factors to generate useful and accurate fits.

In the *Ephemeral Data-Derived Potential* (EDDP) framework,[269] each atomic environment is described using features corresponding to one-, two-, and three-body terms. However, rather than regressing each of these features to separate energy components, all features for a given atom are concatenated and regressed to a single local energy: this removes the possibility of extracting separate, interpretable contributions from interactions of each body-order.

Finally, *ZBL core repulsion* terms[141] are frequently incorporated into GNN-based MLFFs, providing purely repulsive interactions at small distances: this can act to make GNN models much more robust in MD simulations.[11]

5.4. Proof of concept: copper

I demonstrate proof of concept for my methodology by learning symbolic PES models for bulk metallic copper. To ensure fair comparisons with the models discovered by *Hernandez et al.*,[119] I use the same dataset and loss function:

The dataset contains 150 bulk structures, each labelled with total energy, forces and cell stress tensors. These structures were generated using *ab initio* MD simulations, starting from a face-centred cubic structure containing 32 atoms. 50 snapshots were taken for each of the following conditions: (i) NVT at 300 K, (ii) NVT at 1,400 K, and (iii) NpT at 1,400 K. I use the same train/test split as *Hernandez et al.*, randomly partitioning this dataset into training, validation and test sets containing 75, 25 and 50 structures respectively. Since the dataset originates from an MD trajectory, I note that the structures are not independent of each other, and indeed no temporal splitting was used - this leads to an overly optimisit assessment of model generalisation below, but allows for direct comparison with the models of *Hernandez et al.*

The loss function comprises a weighted sum of total energy mean squared error (MSE), component-wise force MSE and cell stress MSE as averaged over all structures, S, in the dataset, $\mathcal{D}$:

$$\mathcal{L} = \frac{\lambda_E}{|\mathcal{D}|}\sum_{S\in\mathcal{D}}\left[\hat{E}(S) - E(S)\right]^2 + \frac{\lambda_F}{3\cdot\sum_{S\in\mathcal{D}}|S|}\sum_{S\in\mathcal{D}}\sum_{i\in S}\sum_{\alpha\in\{x,y,z\}}\left[\hat{F}_{\alpha,i}(S) - F_{\alpha,i}(S)\right]^2 + \frac{\lambda_\sigma}{|\mathcal{D}|}\sum_{S\in\mathcal{D}}\sum_{\alpha,\beta\in\{x,y,z\}}\left[\hat{\sigma}_{\alpha,\beta}(S) - \sigma_{\alpha,\beta}(S)\right]^2$$

with $\lambda_E = 5$ eV^{-2}, $\lambda_F = 4$ eV^{-2} Å^2, and $\lambda_\sigma = 50$ eV^{-2} Å^6.

5.4.1. Pair potentials

Metallic bonding dominates the PES of bulk copper, leading to a lack of directionality, and a relatively simple surface. I therefore begin by training a two-body model, using a local cutoff of 5 Å in order to directly compare against the models of *Hernandez et al.*:[119]

$$E = \sum_{i\neq j} V(r_{ij})\cdot\phi_{c=5}(r_{ij})$$

$$V(r) = \mathrm{MLP}_{[18\to80\to1;\ \texttt{CELU}]}\left({}^{h=18}B_{c=5}[r]\right)$$

Training this model to saturation on the validation set takes less than 1 minute on a M2 MacBook Pro using full-batch `L-BFGS` optimisation. The trained model performs well on the test set, achieving 2.16 meV/atom energy MAE, and 53.8 meV/Å force MAE, corresponding to R^2 values of 0.9995 and 0.9926 respectively.

Subsequent symbolic distillation takes a further 2 minutes (with the genetic search over PES forms taking 90 seconds, and parameter re-optimisation taking 30 seconds), and yields the following functional form:

$$V(r) = \left(\lambda_1 + \lambda_2 e^{-\lambda_3\cdot r}\right)\cdot(\lambda_4 - r) + \lambda_5$$

with model parameters, $\{\lambda_i\}$:

parameter:	λ_1	λ_2	λ_3	λ_4	λ_5
value:	0.463196	278.860	1.95473	2.03051	0.928622

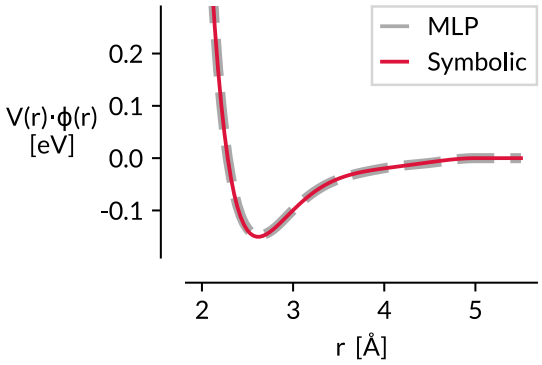


Figure 5.4: *an enveloped, two-body interaction term, $V(r) \cdot \phi_{c=5}(r)$, for bulk copper, as learned by an MLP (grey) and subsequently distilled via symbolic regression (red)*

This symbolic model near-perfectly recapitulates the MLP model (Figure 5.4), and is therefore also accurate, achieving 2.28 meV/atom energy MAE, and 56.3 meV/Å force MAE on the hold-out test set, corresponding to R^2 values of 0.9995 and 0.9921 respectively (Figure 5.5). The ≈5% increase in errors upon symbolic distillation is compensated for by the ≈4× speed-up in inference compared to the MLP-based model.

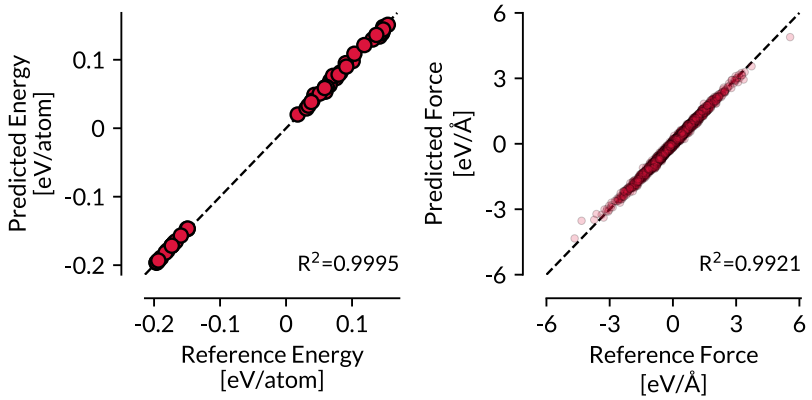


Figure 5.5: *energy and force parity plots for a symbolic pair potential for copper, evaluated on a hold-out test set of 50 structures*

Direct inspection of the symbolic model's functional form reveals qualitatively correct behaviour: the pair potential is smooth everywhere, decays to zero at large distances by construction, and has learned an exponential repulsion term at small distances. We can therefore be confident that this model will remain robust during MD simulations. Indeed, beyond robustness, I find that this model yields near perfect agreement with *ab initio* MD simulations under NVT at 1,400 K and 7.95 g/cm³, as measured by the radial distribution function of the resulting liquid (Figure 5.6).

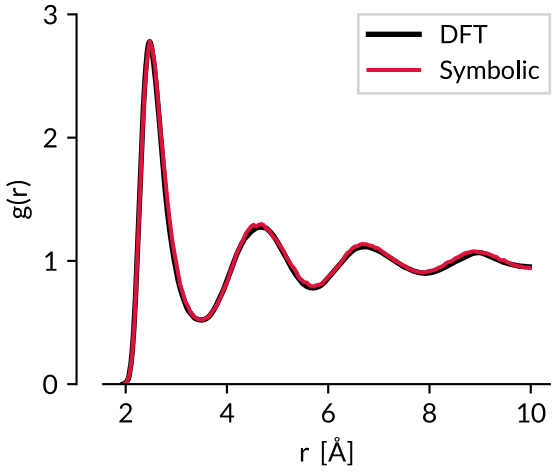


Figure 5.6: *radial distribution functions for liquid copper, averaged over 100 ps of NVT MD at 1,400 K and 7.95 g/cm³, as driven by DFT (black, data from Hernandez et al.[119]) and a symbolic pair potential (red)*

Symbolic distillation via genetic search yields a large population of candidate models. In Figure 5.7, I plot the parameter count against force error for a selection of low-loss models generated during the search process, and highlight the Pareto front.[270] Beyond 5 parameters, I see no improvement in force error - this is expected, as the model I show above already near-perfectly captures the function learned by the MLP of the original model. Models with fewer parameters are unable to fully capture an optimally accurate functional form.

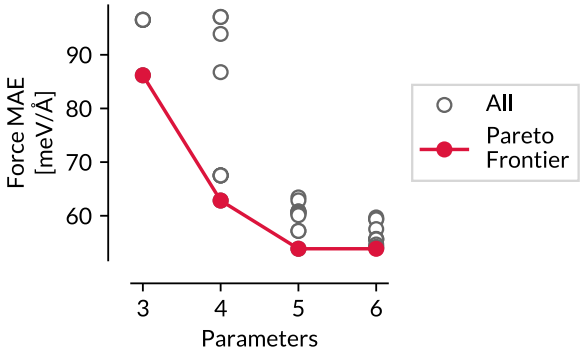


Figure 5.7: *Pareto front (red line) between the parameter count and force error of many symbolic pair potential candidates (points)*

5.4.1.1. Radial cutoff

A key hyper-parameter to tune for any local PES model is the radial cutoff. To investigate the effect of this here, I train a series of MLP models with increasing cutoffs.

In Figure 5.8, I plot the form of the learned interaction term for a selection of these models.

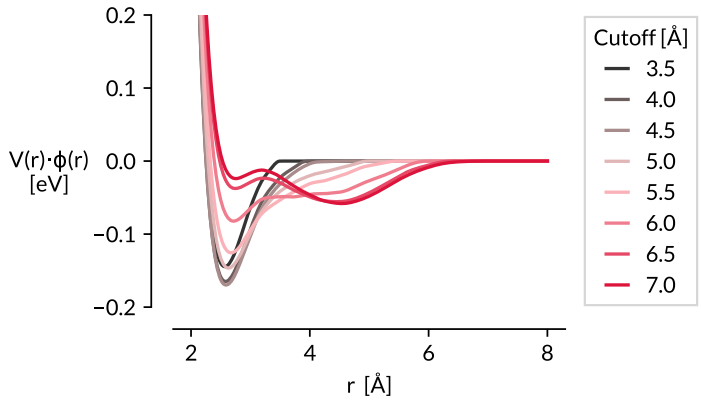


Figure 5.8: *form of the learned interaction term for a series of MLP-based copper pair potentials with increasing radial cutoffs*

At cutoffs lower than 5 Å (black lines), the model only "sees" the first coordination shell, and hence learns a simple function with a single minimum: this is qualitatively similar to the interaction term between two isolated copper atoms. However, at larger cutoffs (red lines), the model learns a function with two minima. This would be unphysical for two isolated copper atoms, but more accurately captures the energetics for bulk metallic systems, where the relative distances to second coordination shells are important, and influence the total energy due to *e.g.* Friedel oscillations.[271] Such double-dip functional forms are often applied to other chemical systems, and can be found in existing coarse-grained[272] and soft-matter[273,274] empirical potentials.

In Figure 5.9, I plot the energy, force and stress errors on the hold-out test set for MLP-based models as a function of their radial cutoff. These lines are surprisingly smooth given a lack of averaging over random seeds, indicating the effectiveness of the full-batch `L-BFGS` optimisation training protocol. Energy and force errors are non-decreasing with radial cutoff, but plateau at beyond ≈7 Å: this corresponds to the size of the unit cells of structures in the training set. Two intermediate plateaus are visible, corresponding to complete inclusion of the first coordination shell, and of interstitial atoms between the first and second shells.

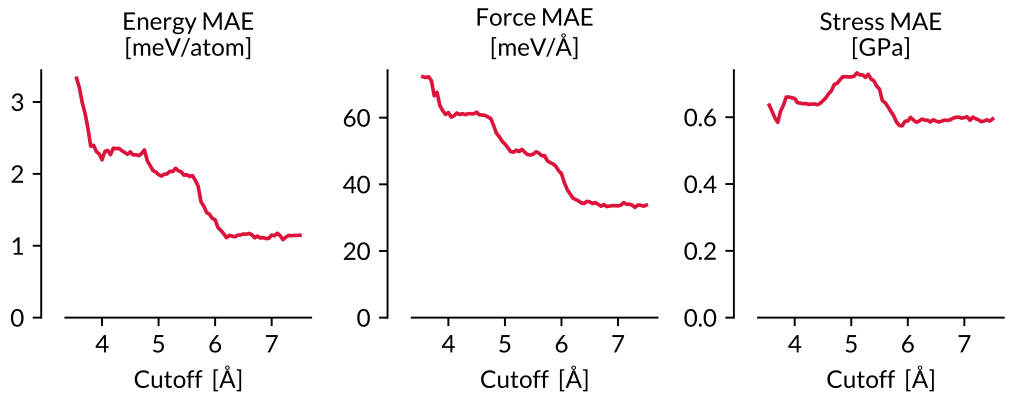


Figure 5.9: *energy, force and stress errors vs radial cutoff for a series of MLP-based copper pair potentials*

To obtain a maximally accurate symbolic pair potential, I distil the 7 Å model, and recover the following functional form:

$$V(r) = \left(\lambda_1 + \lambda_2 e^{-\lambda_3 r}\right) \cdot \left(\left[r - \lambda_4\right]^2 + \lambda_5\right) + \lambda_6$$

parameter:	λ_1	λ_2	λ_3	λ_4	λ_5	λ_6
value:	0.12471	72.0676	2.28011	5.123208	−9.532468	0.963294

The increased complexity of the functional form is reflected in an increased number of parameters relative to the 5 Å model (6 *vs* 5). Interestingly, the learned forms are highly similar, with identical short-ranged, exponential repulsion terms.

Despite improved accuracy with respect to DFT energies and forces, I find the 7 Å model generates identical radial distribution functions under NVT MD simulations as compared to the 5 Å model. However the increased number of atom pairs to consider leads to a $(7/5)^3 \approx 3\times$ slowdown in inference speed, making this model less useful for large-scale simulations.

5.4.2. Embedded atom model

Metallic bonding is not directional. Nevertheless, many-body effects are still present; EAM[47] (and similar, *e.g. Sutton-Chen*[275]) models have been shown to accurately model metallic systems. I therefore investigate a combined 2B+EAM model, again with a local cutoff of 5 Å for direct comparison with *Hernandez et al.*'s[119] existing literature benchmarks:

$$
\begin{aligned}
E &= \sum_i f\left(\sum_{j\neq i} g_{c=5}(r_{ij}) \cdot \phi_{c=5}(r_{ij})\right) \\
&\quad + \sum_{i\neq j} V(r_{ij}) \cdot \phi_{c=5}(r_{ij}) \\
V(r) &= \mathrm{MLP}_{[18\to 80\to 1;\ \mathsf{CELU}]}\left({}^{h=18}B_{c=5}[r]\right) \\
g_{c=5}(r) &= \mathrm{MLP}_{[10\to 64\to 64\to 1;\ \mathsf{CELU}]}\left({}^{h=10}B_{c=5}[r]\right) \\
f(\rho) &= \mathrm{MLP}_{[1\to 64\to 64\to 1;\ \mathsf{CELU}]}(\rho)
\end{aligned}
$$

Training takes 30 seconds on an NVIDIA A6000 GPU using full-batch `L-BFGS` optimisation. Subsequent distillation takes 5 minutes, and yields the following functional forms:

$$
\begin{aligned}
V(r) &= \lambda_1^{(r-\lambda_2)} + \frac{\lambda_3}{r^{\lambda_4}} \\
g(r) &= \lambda_5 \cdot (r-\lambda_6)^2 + \lambda_7 \\
f(\rho) &= \rho \cdot (\lambda_8 \cdot \rho + \lambda_9)
\end{aligned}
$$

with model parameters, $\{\lambda_i\}$:

parameter:	λ_1	λ_2	λ_3	λ_4	
value:	0.03072	2.23538	0.33082	1.83102	
parameter:	λ_5	λ_6	λ_7	λ_8	λ_9
value:	−0.27953	3.734115	−0.192939	0.074348	1.09796

Comparing both to the pair potential models presented above, and to the models of *Hernandez et al.*,[119] I find that this 2B+EAM model is dominantly accurate, achieving 1.39 meV/atom energy MAE, 41.1 meV/Å force MAE, and 0.279 GPa stress MAE on the hold-out test set (Figure 5.10).

I also include comparisons to the models of *Varughese et al.*[266] These models were trained on a different dataset (one that is not publicly available), and the authors do not specify the level of theory at which the data was labelled. Furthermore, these models are not perfectly smooth: no envelope functions are used in their construction, and hence the force predictions of these models are discontinuous at the cutoff radius. With these caveats in mind, I find that my EAM model also outperforms the energy and force performance of the SR1 and SR2 models of *Varughese et al.*, as measured on the hold-out test set used above.

Finally, I note that the EAM model I have learned contains 9 learnable parameters, while the most accurate models of *Hernandez et al.* and *Varughese et al.* (GP2 and SR2, respectively) contain 8 and 16 learnable parameters respectively. The increase in accuracy of my approach therefore arises not from an increased model size, but in the ability to learn a more suitable functional form.

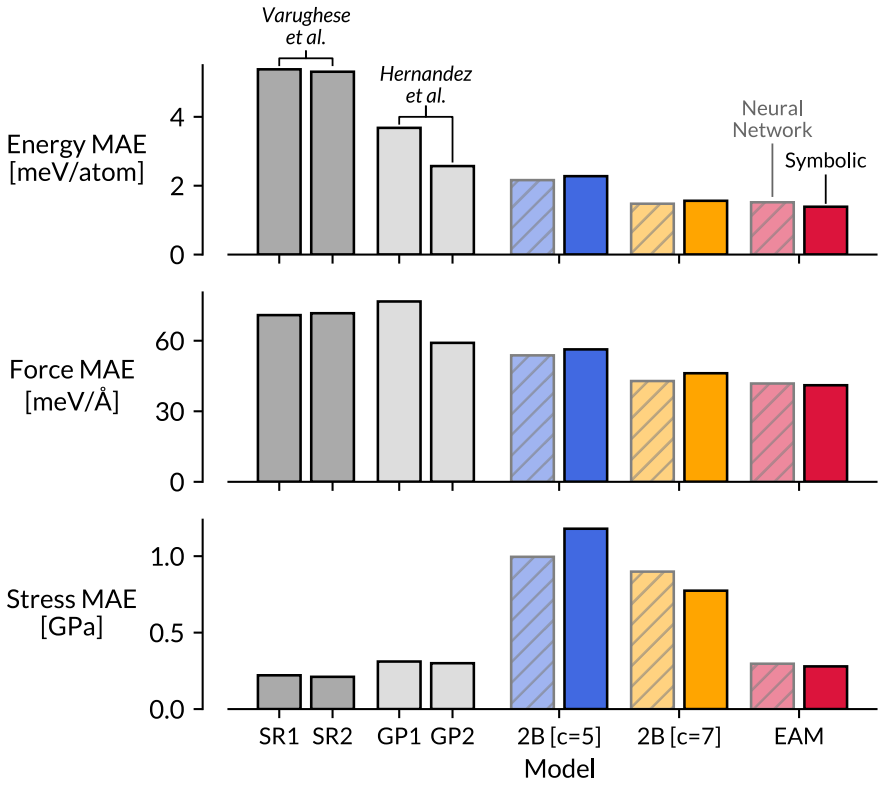


Figure 5.10: *comparing various symbolic models (solid colours) together with relevant MLP counterparts (hatched bars) on hold-out test-set errors. Literature benchmarks from Hernandez et al.[119] and Varughese et al.[266] are shown in grey.*

Beyond improved interpretability, symbolic models are also faster to evaluate than their MLP-based counterparts. Across the board, I find this speed-up to be ≈4× (Figure 5.11).

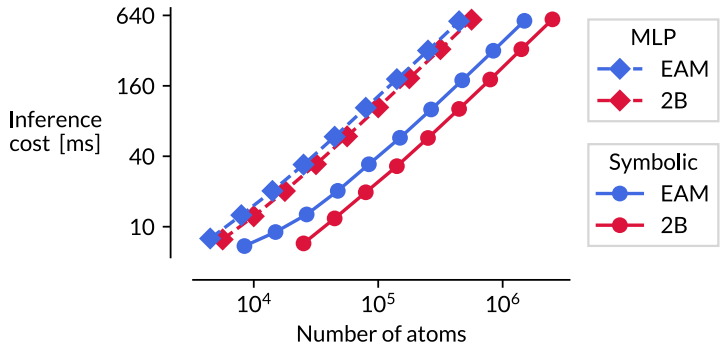


Figure 5.11: *inference cost, measured in ms per prediction (lower is better) on a NVIDIA A100 GPU, for MLP-based and symbolically distilled models*

5.4.2.1. Stress tensors

Correct modelling of cell stress tensors is crucial for performing accurate MD simulations under constant pressure conditions, as used for instance to determine the density of a system. Interestingly, strong energy and force performance does not necessarily lead to accurate stress predictions. Indeed, the simple pair potential models trained above are incapable of modelling stresses accurately and simultaneously for structures at both temperatures in the training set (300 K and 1400 K) — see the two off-diagonal clusters in Figure 5.12 LHS. Many-bodied information is required to model the stress tensors correctly for this system, as evidenced by the dramatically better performance of the EAM model — see Figure 5.12 RHS.

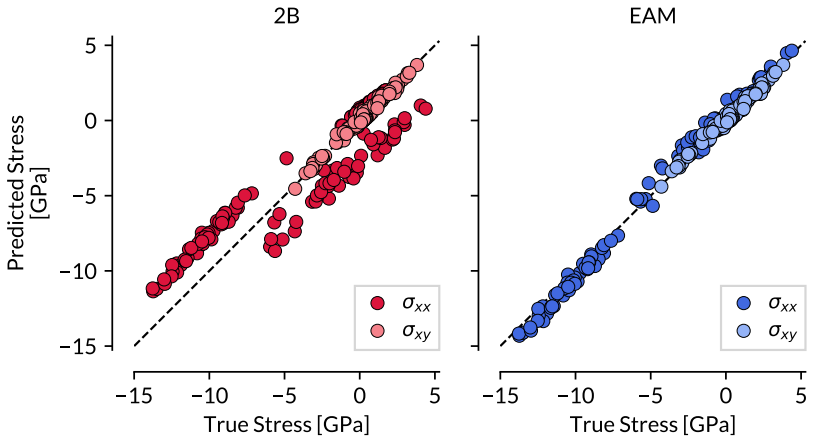


Figure 5.12: *stress-tensor component parity plots for symbolic two-body (LHS, red) and EAM (RHS, blue) models*

The breakdown of the pair potential leads to an incorrect density prediction of liquid copper at 1400 K (6.4 g/cm³ *vs* the experimental value of 8 g/cm³ from *Assael et al.*, 2010[276]) — see Figure 5.13. In contrast, the EAM model predicts a density (8.06 g/cm³) within 1 standard deviation of the experimental value. As with any PES model, there is no guarantee that the EAM will remain accurate outside the scope of its training data. Indeed, at temperatures beyond 1500 K, I observe a sharp deviation away from the experimentally observed density.

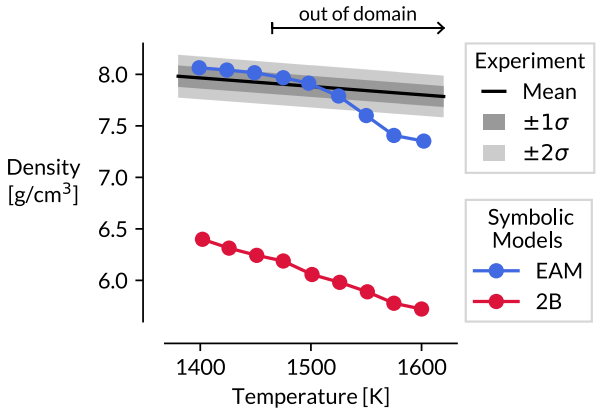


Figure 5.13: *temperature dependence of the density predictions for liquid copper for pairwise (red) and EAM (blue) models, generated by averaging over 100 ps of NpT MD*

To further understand why pair potentials are incapable of modelling stresses accurately, in Figure 5.14 I plot the (energy, force, stress) error against the stress weight, λ_σ, used in the loss function when training a collection of MLP-based models.

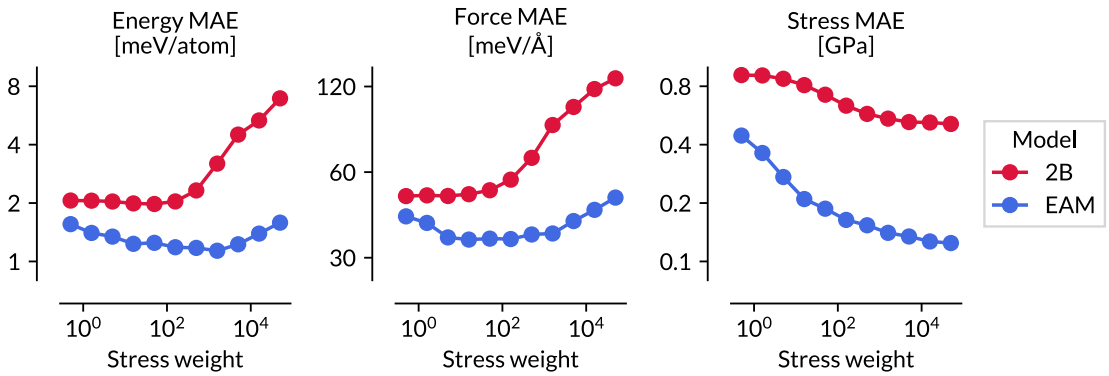


Figure 5.14: *error vs stress weight for the MLP-based pair potentials (red) and EAM models (blue)*

There is a **very clear trade-off** between the accuracy of the two-body models (red) on stresses (RHS) *vs* on energies and forces (LHS and MIDDLE). Further, the stress error plateaus at high loss even in the limit of very large stress weights, indicating that the pair potential is incapable of modelling stresses accurately even when energies and forces are, to a large extent, ignored.

The EAM models (blue) are more accurate across all metrics as compared to the pair potentials. Beyond this superiority, very different behaviour is observed as the stress weight increases. Moving from low to medium stress weights ($\lambda_\sigma = 1 \rightarrow \lambda_\sigma = 100$), **all three error terms improve** for the EAM model! In the limit of very large stress weights, the energy and force errors do start to increase, but not nearly as

markedly as for the pair potential models. These two observations clearly demonstrate that the functional form of the EAM is sufficiently expressive to model the stresses well, without needing to overly compromise on the accuracy of the energies and forces.

Beyond the simple case of metallic copper, I now apply my methodology to more challenging systems.

5.5. Application: bulk silicon

Silicon serves as a primary substrate for the modern industrial complex, with its tunable semi-conductivity being the fundamental enabler for both advanced microelectronics and large-scale photovoltaic devices. Beyond its ground-state diamond structure, bulk silicon exhibits widespread allotropy, existing in β-tin, high-density BC8, and hexagonal crystal structures, as well as polycrystalline and amorphous forms.

Existing empirical potentials, including the *Stillinger Weber*[142] model, have been optimised to study diamond-like silicon structures, but generalise poorly to other allotropes. Below I train a 2+3-body symbolic PES for silicon, and demonstrate its (perhaps surprising) ability to model a wide range of silicon's chemical diversity.

5.5.1. Dataset

To target a range of silicon allotropes, I use the `Si-GAP-18` dataset developed by *Bartok et al.* (2018).[123] I select all bulk structures, discard any that contain exotic and high-energy sp and sp² hybridised environments, and split the remaining dataset into training, validation and test sets containing 1,000, 100 and 703 structures respectively. I use stratified splits such that each configuration type (*e.g.* diamond, β-tin, liquid, amorphous) is represented in equal proportion across the training and test sets, ensuring that test-set error metrics are not dominated by any single region of chemical space.

Analysis of this dataset demonstrates the complex distribution of characteristic lengthscales adopted by silicon allotropes: in Figure 5.15, I consider all possible three-body triplets up to a 4 Å separation, and plot the distribution of (m, d) symmetrisations for each. Weighting the contribution of each triplet by $r_{ij}^{-2} r_{ik}^{-2}$ accounts for the increasing number of neighbours as r increases, in direct analogy with the normalisation term of a 1D radial distribution function.

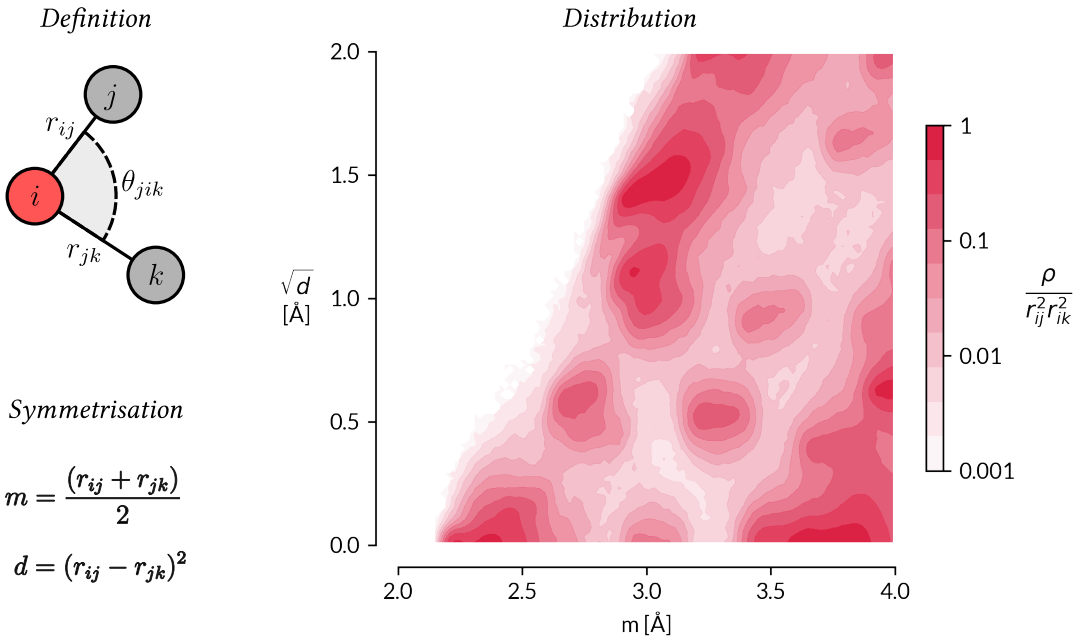


Figure 5.15: *distribution of* $\left(m, \sqrt{d}\right)$ *for all three-body triplets up to* $4\,Å$ *separation in the training set*

Many peaks exist in this distribution, corresponding to clusters of similar triplets in the dataset. Peaks along the bottom of the plot (*i.e.* $d \approx 0$) correspond to triplets where j and k are in the same coordination shell, while peaks in higher d regions correspond to j and k being increasingly separated.

Beyond rich structure in bond lengths, and despite the strong preference for tetrahedrality in "normal" silicon structures, a very wide range of characteristic angles are also present in the dataset. To demonstrate this, in Figure 5.16, I plot the distribution of (m, θ) for various ranges of d, using the same normalisation scheme as above.

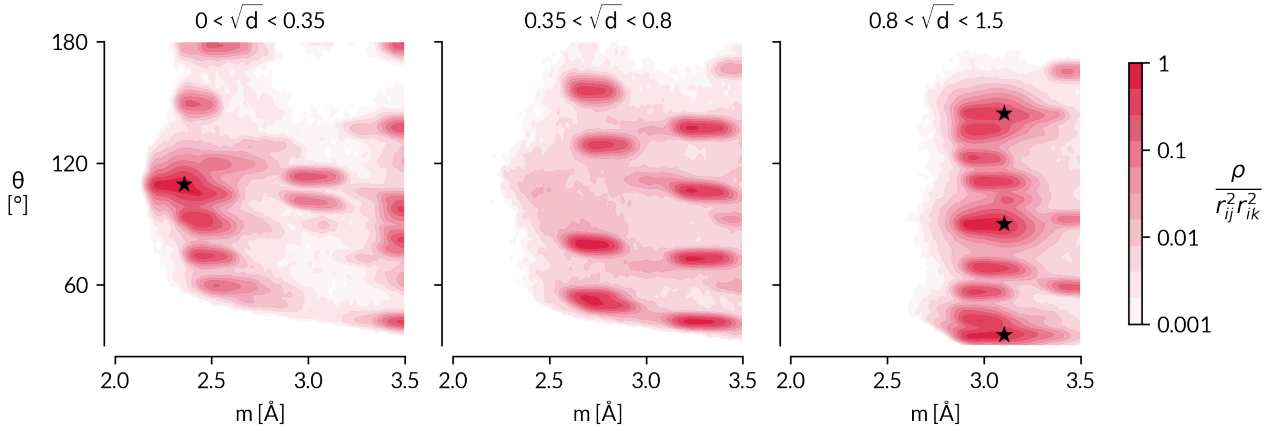


Figure 5.16: *distribution of (m, θ) for all three-body triplets up to 3.5 Å separation in the training set. Black stars mark the location of triplets found in the ground-state diamond structure.*

Again, clear clustering is present. As well as the tetrahedral environments found in the ground-state diamond structure (black stars in Figure 5.16), triplets with bond angles ranging from 60° to 180° are present in the first coordination shell alone (Figure 5.16 LHS, $2.1 \leq m \leq 2.9$ Å). The rich angular structure inherent to the structures within this dataset clearly motivates the need for a bond-angle (i.e. three-body) term in the PES.

5.5.2. Model

I use the following construction for the PES:

$$E = \sum_{i \neq j} V(r_{ij}) \cdot \phi_c(r_{ij}) + \sum_{i \neq j \neq k} f(m_{ij,ik}, d_{ij,ik}, \theta_{jik}) \cdot \phi_c(r_{ij}) \cdot \phi_c(r_{ik})$$

with each learnable component parameterised by an MLP:

$$V(r) = \text{MLP}_{[10 \to 64 \to 1;\ \text{CELU}]}\left({}^{h=10}B_{c=3.5}[r]\right)$$

$$f(m, d, \theta) = \text{MLP}_{[44 \to 16 \to 16 \to 1;\ \text{CELU}]}\left({}^{h=6}B_{c=3.5}[m],\ {}^{h=20}G_{c=(3.5)^2}[d],\ {}^{h=18}G_{\pi}[\theta]\right)$$

i.e. where both two-body and three-body terms have a 3.5 Å cutoff, and m, d and θ are expanded into 6, 20 and 18 features respectively.

Interestingly, increasing the radial cutoff, c, does not improve the accuracy of the MLP-based model. This demonstrates the short-ranged nature of the two- and three-bodied interactions in bulk silicon. Similarly, incorporating an EAM term also leads to no improvement; this suggests that the many-bodied aspects of the silicon PES are not captured well by a simple local density field, which is incapable of modelling *e.g.* important directional and polarisation effects.

Training the MLP-based model to saturation takes 10 minutes on an NVIDIA A6000 GPU using full-batch `L-BFGS` optimisation. Distilling the two-body term takes 90 seconds, and yields the following functional form:

$$V(r) = \lambda_1^{(\lambda_2 - r)} + \lambda_3 \cdot r + \lambda_4$$

Subsequent re-optimisation of the model parameters takes 30 seconds, and gives: $\lambda_1 = 13.862$, $\lambda_2 = 2.61134$, $\lambda_3 = 1.71371$, $\lambda_4 = -7.61232$. Combined with the envelope function, this yields a two-body interaction with a well-defined minimum at 2.4 Å (intermediate between the experimentally determined minimum bond lengths in the diamond,[277] 2.35 Å, and β-tin,[278] 2.44 Å, crystal structures). Direct inspection of both the functional form and the dimer-curve presented in Figure 5.17 reveal a physically-correct exponential repulsion at short distances.

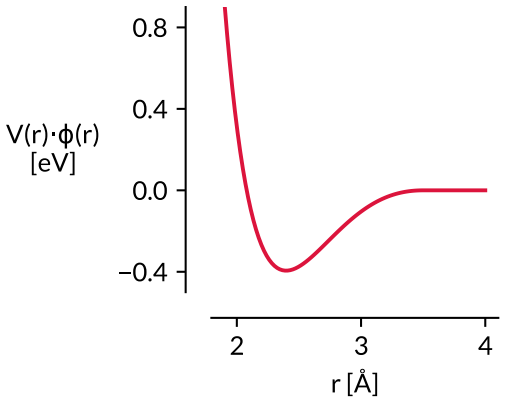


Figure 5.17: *learned symbolic two-body interaction term for bulk silicon*

Distilling the three-body term yields a more complicated functional form:

$$f(m, d, \theta) = \cos(d) \cdot [(\cos(\lambda_1 \theta + \lambda_2) + \lambda_3) \cdot \cos(\lambda_4 m + \lambda_5 \theta) \cdot \cos(m - \lambda_6) + \lambda_7]$$

with model parameters, $\{\lambda_i\}$:

parameter:	λ_1	λ_2	λ_3	λ_4
value:	3.2176	1.14046	2.3985	0.274636
parameter:	λ_5	λ_6	λ_7	
value:	2.91294	0.242292	0.547488	

Understanding this term via direct inspection is challenging. However, we can still visualise the complete energy landscape of this three-body term by plotting the PES for a range of values of m, d and θ (Figure 5.18).

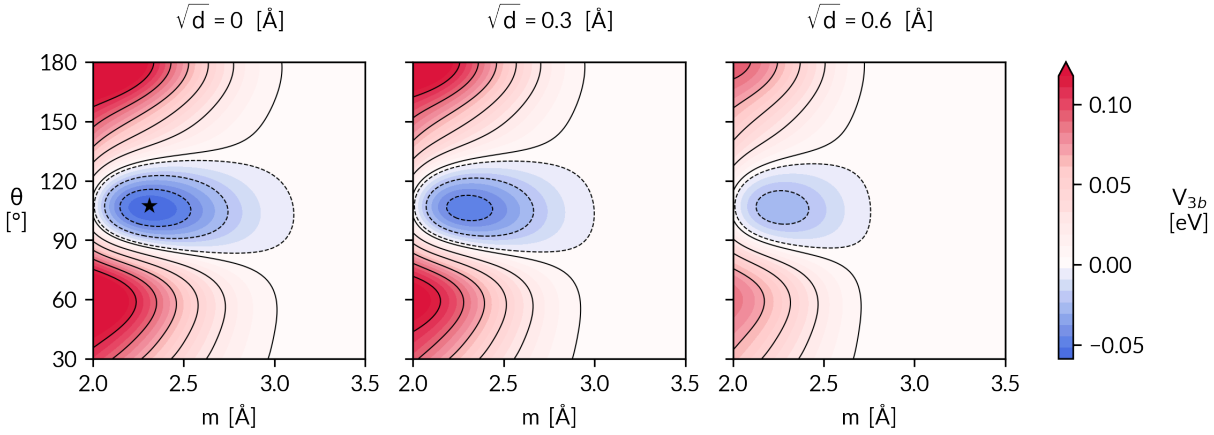


Figure 5.18: *three-body interaction term for silicon. A black star marks the global minimum at* $m = 2.32\,Å$, $d = 0\,Å$ *and* $\theta = 109.5°$.

The three-body term is smooth everywhere, by construction, and has learned a single basin of attraction about the global minimum (blue in Figure 5.18), which coincides near-perfectly with the tetrahedral geometry of the ground-state diamond structure (marked with a black star in Figure 5.18). As expected, the three-body term acts as a relatively weak perturbation on top of the strong two-body interactions: the basin depth of the two-body term is ≈8× deeper than that for three-body interactions.

5.5.3. Evaluation

To evaluate the performance of this symbolic model, I begin by comparing its energy predictions to those of the *ab initio* reference method: DFT with the PW91 functional.[279] I find close agreement ($R^2 = 0.990$, MAE = 11 meV/atom) on a hold-out test set composed of many silicon allotropes, together with liquid and defective structures (Figure 5.19 RHS).

In contrast, the *Stillinger Weber*[142] model (Figure 5.19 LHS) displays reasonable correlation for diamond-like structures (*i.e.* the chemical domain it was constructed to target, $R^2 = 0.941$), but generalises extremely poorly to both non-tetrahedral allotropes (such as β-tin type and high-density BC8 structures) and liquid structures, dramatically over-estimating the total energy in all cases.

One notable failing shared by both of these (short-ranged) models is their inability to model the "decohesion" set of structures, in which planes of atoms are separated to large distances, and yet still interact via long-range van der Waals attractions.

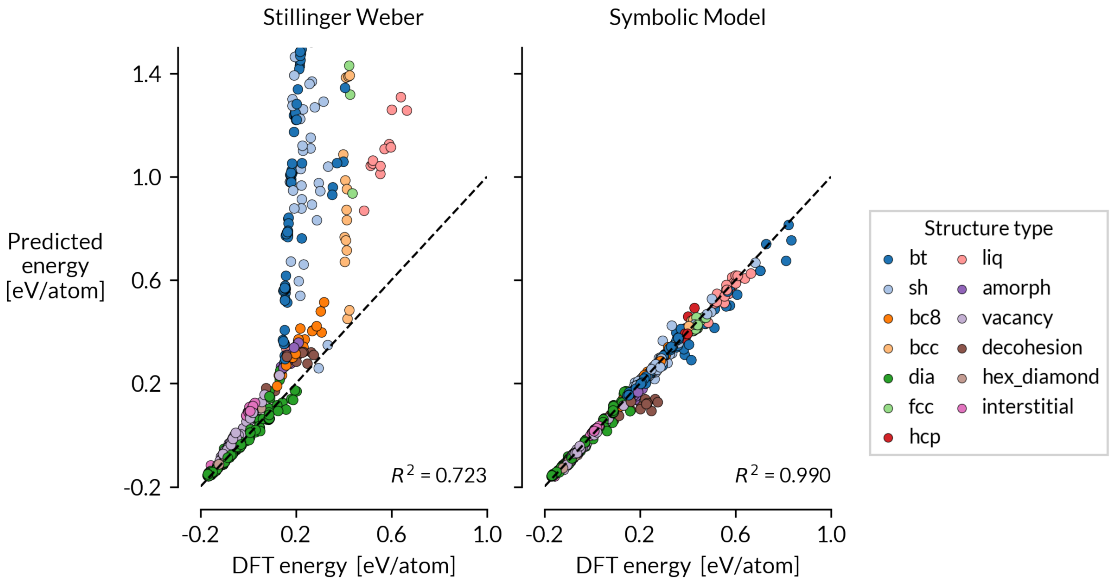


Figure 5.19: *parity plot comparing ground truth (x-axis) and predicted (y-axis) energies for a range of silicon allotropes and structure types (colours)*

The symbolic model accurately approximates the *ab initio* stress tensors on the hold-out test set (Figure 5.20 LHS). Combined with the accurate energy predictions demonstrated above, this leads to strong qualitative agreement of (i) the energy ordering of various crystal structures, (ii) their respective densities, and (iii) the response of their energies to isotropic expansion and compression of the unit cell (Figure 5.20 RHS).

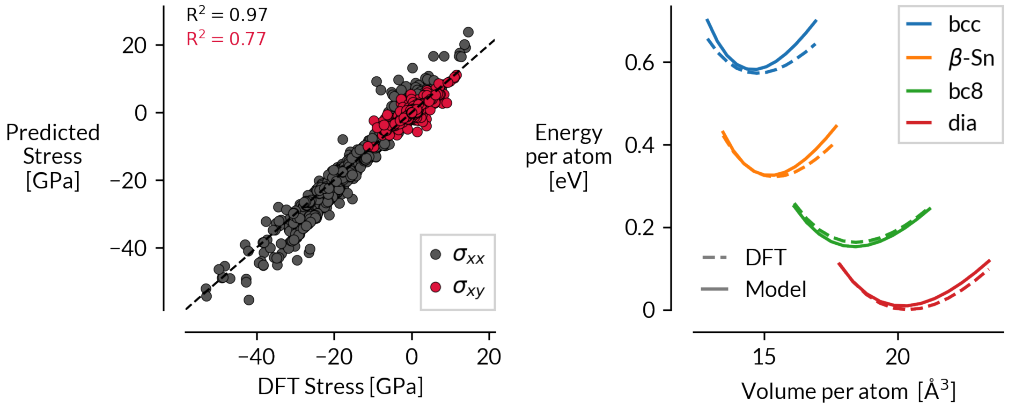


Figure 5.20: *comparing stress predictions between the 2+3-body symbolic model and DFT labels.*

Finally, beyond static evaluations, I analyse the ability of the symbolic PES to model the behaviour of liquid silicon. In Figure 5.21, I plot the radial distribution function as averaged over 10 ps of MD simulation under NpT conditions at 2,000 K and

0 external pressure, and compare to trajectories driven by DFT and reference empirical models (data taken from *Bartok et al.*, 2018).[123]

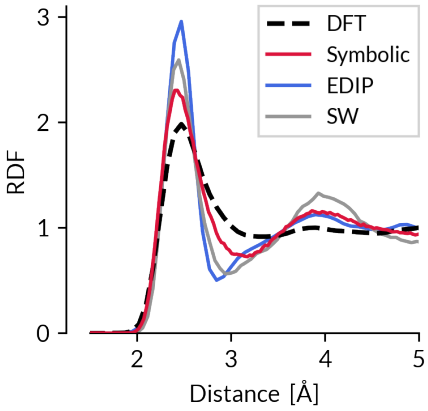


Figure 5.21: *comparison of radial distribution functions for liquid silicon (at 2,000 K) as driven by DFT,[123] a learned symbolic PES, and the Stillinger-Weber[142] and* `EDIP`*[280] empirical potentials*

The symbolic model yields a radial distribution function for liquid silicon that is significantly less ordered than existing potentials (as indicated by the lower value of the first peak, and higher population in interstitial regions), and that is in closer (although not perfect) agreement with *ab initio* MD simulations (Figure 5.21).

In summary, I have demonstrated that a simple 2+3-body symbolic PES is able to model a wide range of silicon's chemical diversity, in contrast to existing empirical potentials. I attribute this improvement not to the energy decomposition (the *Stillinger Weber* model also makes predictions as a sum of two- and three-body terms, and `EDIP` is even more expressive), but rather to the learning of a more suitable functional form from data (rather than via hand-crafting), and to training on a wide range of structures.

5.6. Application: amorphous silica

Amorphous silica, a-SiO_2, forms a bulk solid, composed primarily of vertex-sharing SiO_4 tetrahedra, and displaying both covalent and ionic bonding characteristics. Various existing empirical models have been proposed for this system. In a recent study, *Erhard et al.*[73] developed a GAP-based MLFF for the SiO_2 system (`SiO2-GAP-22`) and compared its performance with several empirical pair potentials, including the `Vashishta`[281], `BKS`[282], and `CHIK`[283] models. Below, I train a symbolic EAM model for this system, and compare its performance to both `SiO2-GAP-22` and empirical models.

5.6.1. Dataset

To target bulk amorphous silica, I use the `SiOx-ACE-24` dataset developed by *Erhard et al.*, 2024.[128] I filter this dataset to include only bulk, non-crystalline structures of Si_nO_{2n} composition. This includes liquid, rapidly quenched, annealed and vacancy-containing structures, generated via a range of protocols including MD and active learning. I split this dataset into training, validation and test sets containing 1,000, 100 and 400 structures respectively, using stratified splits to ensure that each configuration type (liquid, quenched, annealed and vacancy-containing) is represented in equal proportion across all three sets.

5.6.2. Loss function

The resulting training set contains structures of a range of different sizes (32–240 atoms), and is labelled with energies and forces using the SCAN DFT functional.[246] I use a per-atom weighting scheme to prevent biasing towards larger structures, such that the loss function is:

$$\mathcal{L} = \lambda_E \cdot \frac{1}{|\mathcal{D}|} \sum_{S \in \mathcal{D}} \frac{\left|\hat{E}(S) - E(S)\right|}{|S|} + \frac{1}{3 \cdot \sum_{S \in \mathcal{D}} |S|} \sum_{S \in \mathcal{D}} \sum_{i \in S} \sum_{\alpha \in \{x,y,z\}} \left|\hat{F}_{\alpha,i}(S) - F_{\alpha,i}(S)\right|$$

I found $\lambda_E = 5$ to minimise both force and energy errors on the validation set.

5.6.3. Model

I use a combination of two-body and EAM terms to model the PES:

$$E = \sum_{i \neq j} V(r_{ij}, z_i, z_j) \cdot \phi_{c=4}(r_{ij}) + \sum_i f\left(\sum_{j \neq i} g(r_{ij}, z_i, z_j) \cdot \phi_{c=3}(r_{ij}), z_i\right)$$

where each learnable component is parameterised by an MLP:

$$\begin{aligned} V(r, z_i, z_j) &= \mathrm{MLP}^{z_i,z_j}_{[10 \to 64 \to 1;\ \mathtt{CELU}]}\left({}^{h=10}B_{c=4}[r]\right) \\ g(r, z_i, z_j) &= \mathrm{MLP}_{[12 \to 64 \to 64 \to 2;\ \mathtt{CELU}]}\left({}^{h_r=10}B_{c=3}[r_{ij}], {}^{h_z=1}\zeta[z_i], {}^{h_z=1}\zeta[z_j]\right) \\ f(\rho, z_i) &= \mathrm{MLP}^{z_i}_{[2 \to 64 \to 64 \to 1;\ \mathtt{CELU}]}(\rho) \end{aligned}$$

i.e. where the two-body terms (one for each distinct pair of atomic numbers) have a 4 Å cutoff, the EAM term has a 3 Å cutoff, and ${}^{h_z=1}\zeta[z_i]$ is a one-dimensional embedding of the atomic number z_i.

I found that using an embedding dimension of 2 for ρ dramatically improved model accuracy compared to using a single scalar, while leading to very small increases in inference cost.

Training the MLP-based model to saturation takes 5 minutes on an NVIDIA A100. Subsequent distillation of the two body terms yields the following functional forms for each distinct pair of atomic numbers:

$$V_{[\mathrm{Si,O}]}(r) = (\lambda_1 - \lambda_2 r)e^{-\lambda_3 r} + \lambda_4$$
$$V_{[\mathrm{Si,Si}]}(r) = (r - \lambda_5)\left(r^{\lambda_6} - \lambda_7\right)$$
$$V_{[\mathrm{O,O}]}(r) = (r - \lambda_8)(r[\lambda_9 - \lambda_{10} r] - \lambda_{11})$$

with model parameters, $\{\lambda_i\}$:

parameter:	λ_1	λ_2	λ_3	λ_4
value:	6231.8	4781.4	3.25697	2.26695
parameter:	λ_5	λ_6	λ_7	
value:	5.21588	1.3270	3.59614	
parameter:	λ_8	λ_9	λ_{10}	λ_{11}
value:	2.48828	17.1135	2.4306	31.0578

These functional forms display qualitatively correct behaviour (Figure 5.22): oppositely charged Si-O atoms have a basin of strong attraction, with the equilibrium bond length of 1.62 Å matching the experimental value in α-quartz[284] All three terms display strong repulsions at short distances. Both Si-Si and O-O interactions are dominantly repulsive, with very small amounts of attraction at intermediate distances: these features are also observed in the `CHIK` pair potential, and help to stabilise the relative arrangements of SiO_4 tetrahedra (see *e.g.* the Si-Si distance in α-quartz marked in Figure 5.22).

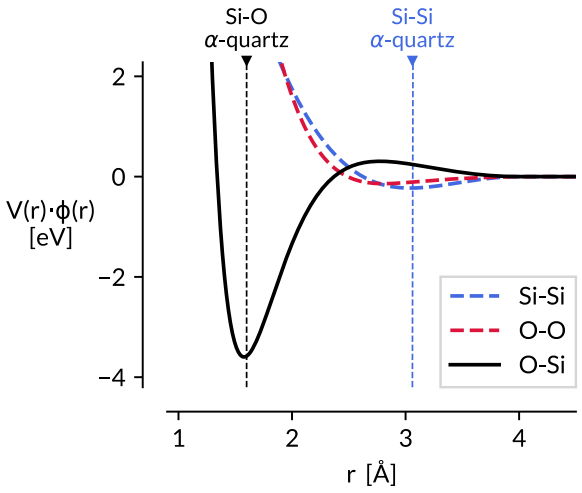


Figure 5.22: *two-body interaction terms learned by a symbolic model for amorphous silica*

Distilling the EAM term yields the following functional form:

$$
\begin{aligned}
g^{(1)}(r, z_i, z_j) &= \left(\lambda_1 \cdot \zeta[z_i] + \lambda_2 e^{\zeta[z_j] r}\right)\left(\zeta\left[z_j\right] + r + \lambda_3\right) + \lambda_4 \\
g^{(2)}(r, z_i, z_j) &= \lambda_5 r + (\lambda_6 \zeta[z_i] - \lambda_7)\left(\zeta\left[z_j\right] + \lambda_8\right) \\
f_{[z_i = \mathrm{Si}]}(\rho) &= -\left(\lambda_9 \rho^{(1)} + \lambda_{10}\right)\left(\cos\left(\lambda_{11} \rho^{(2)}\right) + \lambda_{12}\right) \\
f_{[z_i = \mathrm{O}]}(\rho) &= \lambda_{13} \cos\left(\lambda_{14} \rho^{(1)} + \lambda_{15} \rho^{(2)}\right) + \lambda_{16}
\end{aligned}
$$

with model parameters, $\{\lambda_i\}$:

parameter:	λ_1	λ_2	λ_3	λ_4
value:	8.49611	1.70391	2.08362	2.20532
parameter:	λ_5	λ_6	λ_7	λ_8
value:	4.39389	1.05896	4.55489	1.17515
parameter:	λ_9	λ_{10}	λ_{11}	λ_{12}
value:	0.161485	0.6691	0.470756	0.638173
parameter:	λ_{13}	λ_{14}	λ_{15}	λ_{16}
value:	2.02264	0.648047	0.524761	2.13605

and atomic number embeddings $\zeta_{\mathrm{O}} = -0.365547$ and $\zeta_{\mathrm{Si}} = 0.876027$.

Direct inspection shows that the learned embedding function, g, probes two different quantities, scaling with r and r^2 respectively. We can visualise the information that this embedding function encodes by plotting $\left(\rho^{(1)}, \rho^{(2)}\right)$ for a representative sample of 1,000 atomic environments from the training set, and colour coding by properties

of interest, namely (i) the **EAM's local energy prediction**, $f_{[z_i]}(\rho_i)$, and (ii) a **soft neighbour count**, defined as:

$$N_i^{\text{soft}} = \sum_{j \neq i} \frac{\phi_{c=3}(r_{ij})}{\phi_{c=3}(1.62)}$$

where the normalisation factor is the envelope evaluated at the Si-O equilibrium bond length, such that counts are comparable to those generated by a neighbour count with a hard cutoff. I present separate plots for the Si and O environments, as shown in Figure 5.23 and Figure 5.24 respectively.

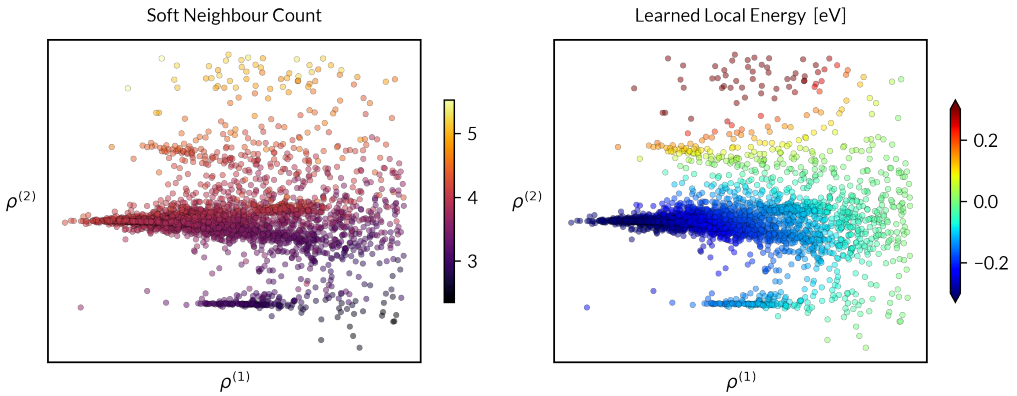


Figure 5.23: *learned embedding space for the symbolic EAM potential. Each point corresponds to a unique Si atomic environment in the training set.*

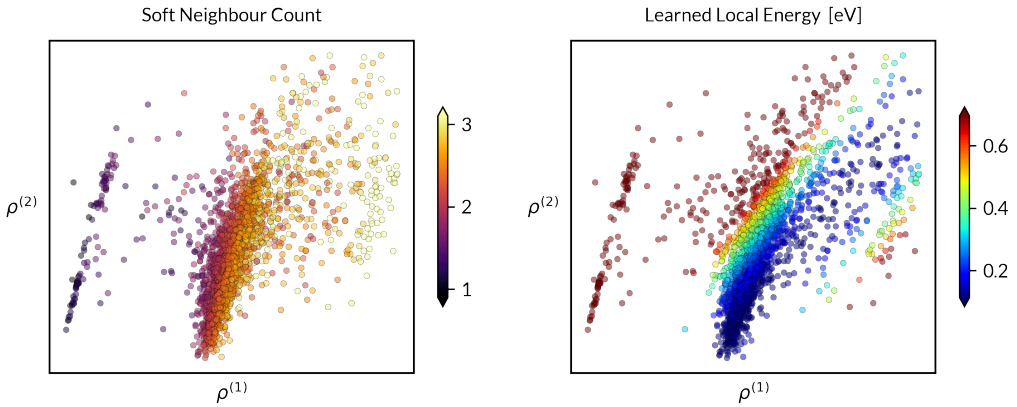


Figure 5.24: *learned embedding space for the symbolic EAM potential. Each point corresponds to a unique O atomic environment in the training set.*

For both kinds of environment, the model has clearly learned to separate environments with distinct soft neighbour counts using one dimension, and then to generate a gradient of local energy predictions for each cluster using the other dimension. I posit

that this hierarchical clustering, something that would not be possible to achieve with a scalar embedding, is the reason why an embedding dimension of 2 leads to a much more accurate EAM model than when using a scalar embedding.

5.6.4. Evaluation

I find that this (relatively) simple symbolic model is able to perform well on a hold-out test set. Indeed, it outperforms both `CHIK` and `SiO2-GAP-22` models in predicting DFT energy labels (Figure 5.25), with an energy MAE of 21.2 meV/atom. Beating `SiO2-GAP-22` here is especially impressive considering some of this data comprises the `SiO2-GAP-22` training set, and that `SiO2-GAP-22` contains 4,090 learnable coefficients (compared to the 29 parameters of the symbolic model). Having only been fit to reproduce experimental radial distribution functions, it is not surprising that the `CHIK` model performs poorly in quantitatively reproducing DFT energies.

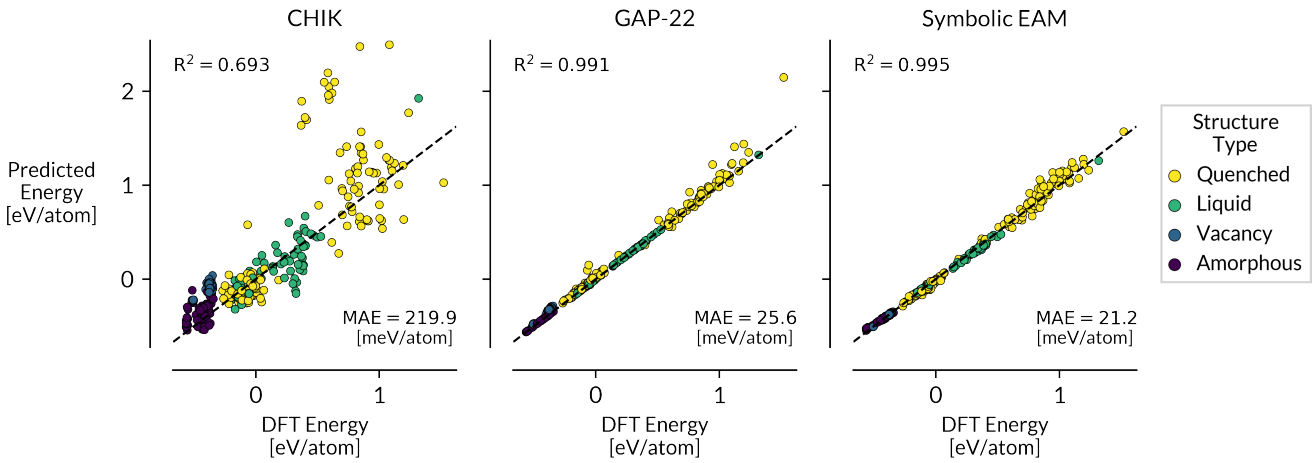


Figure 5.25: *per-atom energy parity plots for `CHIK` (LHS), `SiO2-GAP-22` (MID), and the symbolic EAM model (RHS), as evaluated on a hold-out test set composed of several kinds of structure*

Force accuracy for the symbolic EAM model is also competitive, with R^2 values of 0.874 and 0.895, and RMSEs of 0.76 eV/Å and 0.70 eV/Å, for the force component predictions of the symbolic EAM model and `SiO2-GAP-22` respectively.

Interestingly, despite similar numerical errors, the local energy partition learned by the symbolic EAM model departs significantly from that learned by `SiO2-GAP-22`, as shown in Figure 5.26.

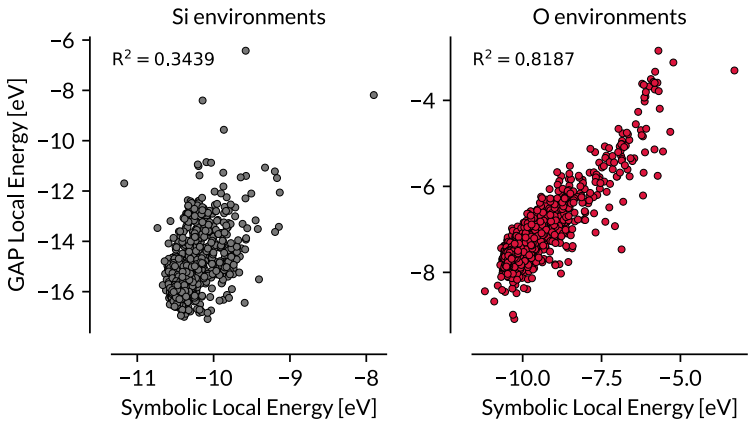


Figure 5.26: *discrepancy between the learned local energy partitions of the symbolic EAM model (x-axis) and* `SiO2-GAP-22` *(y-axis)*

Beyond numerical validation on the hold-out test set, I adopt the MD-based validation protocols proposed and carried out by *Erhard et al.* (2022).[73] This entails running a five phase protocol, wherein a random hard-sphere configuration is (i) randomised at 6,000 K for 10 ps under NVT, (ii) melted at 4,000 K for 100 ps, (iii) quenched to 300 K at a rate of 10^{13} K/s, and (iv) held at 300 K for 10 ps, with phases (ii-iv) all under NpT at 0 external pressure. Finally, (v) the structure is relaxed to a maximum force magnitude of 0.01 eV/Å.

When driven by an accurate PES model, the quenching in this protocol is sufficiently gradual to produce an amorphous silica structure with (i) a low energy relative to the α-quartz ground state, and (ii) minimal coordination defects (*i.e.* low numbers of atoms not associated with perfect SiO_4 tetrahedra). As model quality increases, the proportion of defects should decrease, as should the final energy of the amorphous structure.

I use the symbolic EAM model to run several of these simulations, using different initial configurations and random seeds, and compare to the trajectories of *Erhard et al.* (2022).[73] In Table 5.7, I present the proportion of both Si and O defects at 300 K for a structure composed of 1,728 SiO_2 units.

Table 5.7: *proportion of defects (measured in %) at 300 K for a quenched structure generated by a range of models, and composed of 1,728 SiO_2 units*

	Si 3-fold	Si 5-fold	O 1-fold	O 3-fold	Total
Vashishta	0.81	0.00	1.07	0.67	1.43
BKS	0.00	1.79	0.09	0.98	1.31
CHIK	0.00	0.06	0.58	0.61	0.81
EAM	0.34	0.46	0.11	0.17	0.46
GAP	0.12	0.17	0.03	0.06	0.16

The symbolic EAM model performs well, generating a structure with 0.46% defective (and hence energetically unfavourable) atomic environments at 300 K. This is almost twice as good as the next best empirical model (`CHIK`, 0.81%). However, the many-bodied `SiO2-GAP-22` model still performs substantially better, with only a 0.16% defect concentration.

Alongside coordination defects, I also compare the energetic favourability of the final, relaxed structures generated by each model. To do this, in Figure 5.27 I plot the mean (and standard deviation) of the energy of the final relaxed structures relative to the α-quartz ground state, all as labelled with `SiO2-GAP-22` and averaged over 5 repeats of the MD protocol for structures composed of 64 SiO_2 units.

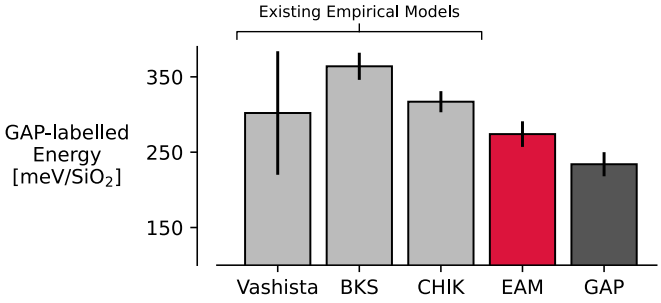


Figure 5.27: *energy parity comparison for the various models. Error bars indicate plus/minus one standard deviation.*

Again, I find that the EAM model (red) outperforms all other empirical models (grey), with a mean energy of +274 meV/SiO_2 relative to the α-quartz ground state (next best is `Vashishta` at +302 meV/SiO_2). Again, there is still a significant gap to the many-bodied `SiO2-GAP-22` model (black), which generates structures with a mean energy gap of +234 meV/SiO_2.

In summary, across both numerical and physical validations, I find that my symbolic EAM model, learned in an end-to-end fashion, is able to outperform established empirical potentials. I attribute this improvement to the more expressive functional form of the EAM term, which allows the model to capture many-body effects that are not present in the simple two-body interactions of the other empirical models. Crucially, whereas existing empirical models have been fitted to reproduce experimental observables, and hence perform poorly in approximating *ab initio* PES labels, the symbolic EAM model outperforms the existing models **both** in its ability to produce realistic structures, and in approximating ground-truth energies and forces.

5.6.5. Improved baselines for many-body force fields?

A common approach to train a MLFF that is both (i) highly accurate and (ii) robust is to make energy (and therefore force and stress) predictions as a sum of some simple "baseline" term, together with a more expressive model:

$$E = E_{\text{baseline}} + E_{\text{expressive}}$$

When using state-of-the-art GNN-MLFFs, this baseline model is typically a repulsive *ZBL* pairwise term[11,141,153] — this provides qualitatively correct behaviour at small interatomic distances, and allows the many-bodied MLFF to "focus" on learning the lower-amplitude, many-bodied interactions.

Here, I briefly investigate the use of my symbolic EAM model as an alternative baseline. I train on the same dataset as above (*i.e* 1k SiO_2 structures), using `TensorNet` models with 10^5 parameters, two message-passing layers with 5 Å cutoffs, a batch size of 32 structures, the `Adam` optimiser with an initial learning rate of 0.002, and a `ReduceLROnPlateau` scheduler with a factor of 0.8 and patience of 25 epochs.

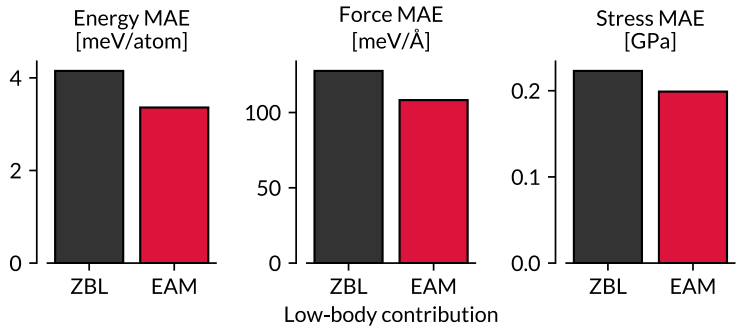


Figure 5.28: *hold-out test-set energy, force and stress MAEs for* `TensorNet` *models trained with* ZBL *(black) and symbolic EAM (red) baselines*

In Figure 5.28, I compare the energy, force and stress MAEs for models trained with *ZBL* (black) and symbolic EAM (red) baselines. I find that using the symbolic EAM model as a baseline improves all of these error metrics by ≈20%. A possible

explanation for this improvement is that, beyond the exponential repulsion of the *ZBL* term, the symbolic EAM model captures other important low-body-order interactions, imparting more useful inductive biases, and hence allowing the GNN model to only model explicitly many-bodied interactions. More work is needed to fully investigate this hypothesis, and to extend this analysis to other systems.

5.7. Summary and outlook

Above, I presented a state-of-the-art method for learning symbolic PES models directly from data, and in an end-to-end manner. The main technical innovation lies in using a two-phase training process: in the first phase, I leverage the representational power of lightweight MLPs to learn accurate functional forms for low-bodied energy contributions, using the `L-BFGS` optimiser to ensure training is fast and stable. In the second phase, I symbolically distil each MLP component in isolation and on representative input-output pairs: symbolic regression is expensive, so this dramatically cuts cost relative to a genetic search over complete PES forms (as used by *Hernandez et al.*[119]). I use a general purpose symbolic regression framework (`PySR`[255]) to avoid being limited to sums over hand-crafted basis functions (as used by *Varughese et al.*[266]).

I started by applying my symbolic distillation procedure to learn PES models for bulk copper, and showed that my approach outperforms existing methodologies from the literature, generating more accurate models while using (substantially) fewer computational resources. Beyond metallic bonding, I also targeted two systems (silicon and silica) dominated by (more complex) covalent interactions. My models improve over established, hand-crafted empirical potentials in both quantitative and qualitative manners. Finally, I showed proof-of-concept that simple, symbolic PES models improve over the popular *ZBL* core repulsion baseline for generating robust and accurate GNN-MLFFs.

While simple symbolic models will never be able to compete with the generality and quantitative accuracy of many-bodied MLFFs, empirical potentials continue to find use in a range of applications,[285,286] and in particular when simulation over extreme time- and length-scales are required,[287,288] or when the functional form of the PES is itself an object of analysis.[272] Learning more suitable functional forms improves the accuracy of these simple models for no increase in computational cost, and therefore boosts the utility of their application.

Looking to the future, natural extensions of this work include:

- **transferability**: above, I learned a new form of low-body-order PES for each new chemical system. Future work could investigate the transferability of these forms: perhaps a small number of terms can be found that are able to model a wide range of chemical systems, up to changes in their exact parametrisation.

- **automation**: above, I relied on domain knowledge to identify important terms for each chemical system. Future work could attempt to automate this process, allowing for rapid application to arbitrary new systems.
- **many-bodied interpretability**: using symbolic, low-body-order models as baselines forces GNN-MLFFs to exclusively learn many-bodied energy contributions. Future work could analyse the many-bodied local energy predictions of these models, and therefore improve our understanding of the kinds of interactions existing architectures struggle to capture.

Attribution and Acknowledgements

The above chapter presents new research, performed solely by me, John Gardner. A publication describing this work is currently in preparation.

6. Outlook

> *"Prediction is very difficult, especially about the future."*
>
> Niels Bohr

> *"By far the greatest danger of Artificial Intelligence is that people conclude too early that they understand it."* [289]
>
> Eliezer Yudkowsky

Predicting the future is hard.

Nevertheless, while history may never repeat itself, it often rhymes. Below, I offer speculation about the future trajectory of MLFF research and application, attempting to ground my thoughts in observable trends and recent developments in adjacent fields.

6.1. A high level perspective

Richard Sutton's bitter lesson[71] appears to be holding true: in nearly all domains of machine learning, *general methods that leverage large amounts of data and computation continue to dominate over specialised and hand-crafted approaches.* Projecting into the future, and in the specific context of MLFFs, this suggests that increasingly large models will continue to be trained on increasingly large datasets, with commensurate expansions in their scope and accuracy. The importance of automated systems for exploring chemical space and generating training data, such as the `augment-atoms` approach I presented in Chapter 4, will therefore only continue to grow.

In recent years, the march of progress in the successful design and application of machine learning systems has been relentless. One particularly striking observation is that steady and dependable improvements in loss metrics can lead to sudden and massive changes in qualitative model performance[290] (*e.g.* GPT2[111] → GPT3[291] → GPT4[292]). Despite the many challenges the field still faces (see below), it seems plausible that MLFFs will also experience such a phase change, becoming sufficiently accurate and general to be suddenly applicable to nearly all chemical domains with high confidence. If this comes to pass, I can speculatively imagine two possible outcomes, among others:

- **unlocking biology**: biological systems are "just" collections of atoms interacting with each other. Sufficiently fast and accurate MLFFs could unlock routine simulations of arbitrary proteins, enzymes, and other biological structures and processes at *ab initio* accuracy, dramatically surpassing

the applicability of existing empirical models.[46,293] Coupled perhaps with improved sampling methods,[294–296] this could revolutionise our ability to understand and target biological processes, and hence to design new drugs and therapies at an unprecedented rate and efficacy.

- **a materials discovery cascade**: the space of stable crystal structures is vast, encompassing a combinatorially large range of possible compositions and geometries. MLFFs are increasingly well positioned to search through this space in a high-throughput manner, particularly when combined with (rapidly improving) conditionally generative models.[297,298] Beyond confirming their stability, MLFFs are also uniquely poised to investigate important, and hard to predict, properties of these newly proposed materials, including *e.g.* dopant effects,[299] thermal and mechanical properties,[300,301] and the existence of amorphous phases.[302–305]

While physical experiment will of course remain the ultimate arbiter of truth in both of these settings, accurate and reliable MLFFs may be able to dramatically accelerate exploration and discovery cycles. Indeed, the potential here is sufficiently great that many startups have recently emerged to bet on both MLFF-driven materials design *(e.g. Cusp, Entalpic, Periodic Labs, Orbital Materials)* and drug discovery *(e.g. Isomorphic Labs, Axiom Therapeutics, Ångström AI)*.

As a final high-level remark, we can see that MLFFs are highly accurate, and yet we do not fully understand exactly how they learn to make predictions, or why certain architectural changes aid or hinder their performance. As in other fields of machine learning,[306–308] fundamental research studying simple and interpretable models, such as the symbolic PESs I presented in Chapter 5, will continue to be important in understanding and improving MLFF architectures and training approaches.

6.2. The nuts-and-bolts view

If high-level commentary about the future is difficult, then predicting exact mechanisms by which outcomes might occur is even more so. Nevertheless, several clear and concrete challenges remain, and will need to be addressed to unlock the full potential of MLFFs:

- **long range interactions**: long range interactions are important in a wide variety of chemical systems, and remain a critical challenge for MLFFs. Simply scaling to large cutoffs is computationally prohibitive, while the phenomenon of over-squashing appears to prevent the effective addition of more message-passing layers.[114] New approaches are therefore required. Early works attempting to address these challenges include virtual nodes,[309–311] particle-mesh Ewald summations,[312] learnable Coulombic contributions,[313] and most recently, global modelling via application of vanilla

transformers.[192] None of these approaches have yet been incorporated into a popular foundation model, suggesting improvements are still needed.

- **reactivity**: existing foundation models excel at modelling equilibrium geometries, but are frequently observed to perform poorly on reactive systems,[144,226,314,315] tending to underestimate reaction barriers and to struggle with transition states in particular. This appears to be more of a data challenge than an architectural one; reactive configurations are inherently rarer in chemical space (as explored by MD), and are thus under-represented in most training datasets. Systematic generation of high-quality training data for reactive processes will be essential for progress in this area.[232]
- **automated validation**: careful (and costly) *physical validation* is currently required in order to trust MLFF predictions in new application domains.[54] The expert domain knowledge and manual construction required to do this is not scalable: automated validation methods will therefore become increasingly important in the age of general purpose MLFFs. Among existing approaches, uncertainty quantification currently appears most promising: if a model can reliably indicate when its own predictions are trustworthy (*i.e.* is well calibrated), we can gain a degree of confidence in its predictions out-of-the-box. While full model ensembles are too expensive to be practical at large scales (both to train and to deploy), shallow ensembles[316] and last-layer prediction rigidity[317] provide promising initial alternatives.

Despite these challenges, it seems impossible to be anything but optimistic about the future of MLFFs, and the opportunities that they will bring.

Humanity is entering an era of unprecedented computational power: MLFFs are poised to leverage this and unlock a deluge of new discoveries in fields as diverse as drug discovery, materials science, catalysis, energy storage, and fundamental biology.

References

1. Gardner, C. J. & Downs, A. J. Some Reactions of Uranium Chloride Pentafluoride. *Journal of the Chemical Society, Dalton Transactions* 1289 (1986).
2. Gardner, L. *Balthasar at the End of Modernity.* (T&T Clark, Edinburgh, 1999).
3. Davidson née Gardner, D. Pastoral Imagery as a Homiletic Device in Alcuin's Poems. (2023).
4. Newton, I. *Opticks: Or, A Treatise of the Reflections, Refractions, Inflexions and Colours of Light : Also Two Treatises of the Species and Magnitude of Curvilinear Figures.* (London, 1704).
5. Feynman, R. P. Simulating Physics with Computers. *International Journal of Theoretical Physics* **21,** 467–488 (1982).
6. Behler, J. & Parrinello, M. Generalized Neural-Network Representation of High-Dimensional Potential-Energy Surfaces. *Physical Review Letters* **98,** 146401 (2007).
7. Bartók, A. P., Payne, M. C., Kondor, R. & Csányi, G. Gaussian Approximation Potentials: The Accuracy of Quantum Mechanics, without the Electrons. *Physical Review Letters* **104,** 136403 (2010).
8. Schütt, K. T., Sauceda, H. E., Kindermans, P.-J., Tkatchenko, A. & Müller, K.-R. SchNet – A Deep Learning Architecture for Molecules and Materials. *The Journal of Chemical Physics* **148,** 241722 (2018).
9. Unke, O. T., Chmiela, S., Sauceda, H. E., Gastegger, M., Poltavsky, I., Schütt, K. T., Tkatchenko, A. & Müller, K.-R. Machine Learning Force Fields. *Chemical Reviews* **121,** 10142–10186 (2021).
10. Batatia, I., Kovács, D. P., Simm, G. N. C., Ortner, C. & Csányi, G. MACE: Higher Order Equivariant Message Passing Neural Networks for Fast and Accurate Force Fields. (2023) doi:10.48550/arXiv.2206.07697.
11. Batatia, I. *et al.* A Foundation Model for Atomistic Materials Chemistry. *The Journal of Chemical Physics* **163,** 184110 (2025).
12. Wood, B. M. *et al.* UMA: A Family of Universal Models for Atoms. (2025) doi:10.48550/arXiv.2506.23971.
13. Moore, G. E. Cramming more components onto integrated circuits. *Electronics* **38,** 114–117 (1965).
14. Deringer, V. L., Bartók, A. P., Bernstein, N., Wilkins, D. M., Ceriotti, M. & Csányi, G. Gaussian Process Regression for Materials and Molecules. *Chemical Reviews* **121,** 10073–10141 (2021).
15. Zhou, Y., Zhang, W., Ma, E. & Deringer, V. L. Device-Scale Atomistic Modelling of Phase-Change Memory Materials. *Nature Electronics* **6,** 746–754 (2023).
16. Merchant, A., Batzner, S., Schoenholz, S. S., Aykol, M., Cheon, G. & Cubuk, E. D. Scaling Deep Learning for Materials Discovery. *Nature* **624,** 80–85 (2023).
17. Chen, M., Jiang, X., Zhang, L., Chen, X., Wen, Y., Gu, Z., Li, X. & Zheng, M. The Emergence of Machine Learning Force Fields in Drug Design. *Medicinal Research Reviews* **44,** 1147–1182 (2024).
18. Morrow, J. D., Ugwumadu, C., Drabold, D. A., Elliott, S. R., Goodwin, A. L. & Deringer, V. L. Understanding Defects in Amorphous Silicon with Million-Atom Simulations and Machine Learning. *Angewandte Chemie* **136,** e202403842 (2024).
19. El-Machachi, Z., Cheng, B. & L.~Deringer, V. Mechanical Properties of Graphene Oxide from Machine-Learning-Driven Simulations. *Chemical Communications* **61,** 11405–11408 (2025).
20. Jinnouchi, R. & Minami, S. Machine Learning Force Fields in Electrochemistry: From Fundamentals to Applications. *ACS Nano* **19,** 22600–22644 (2025).

21. Dirac, P. A. M. Quantum Mechanics of Many-Electron Systems. *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character* **123,** 714–733 (1927).
22. Heisenberg, W. Über quantentheoretische Umdeutung kinematischer und mechanischer Beziehungen. *Zeitschrift für Physik* **33,** 879–893 (1925).
23. Schrödinger, E. Quantisierung Als Eigenwertproblem. *Annalen der Physik* **384,** 361–376 (1926).
24. Bohr, N. On the Constitution of Atoms and Molecules. *Philosophical Magazine* **26,** 1–25 (1913).
25. Planck, M. Zur Theorie des Gesetzes der Energieverteilung im Normalspectrum. *Verhandlungen der Deutschen Physikalischen Gesellschaft* **2,** 237–245 (1900).
26. Lord Rayleigh, J. W. S. Remarks upon the Law of Complete Radiation. *Philosophical Magazine* **49,** 539–540 (1900).
27. Born, M. & Oppenheimer, R. Zur Quantentheorie Der Molekeln. *Annalen der Physik* **389,** 457–484 (1927).
28. Broglie, L. de. A tentative theory of light quanta. *Philosophical Magazine* **47,** 446–458 (1924).
29. Heisenberg, W. Über den anschaulichen Inhalt der quantentheoretischen Kinematik und Mechanik. *Zeitschrift für Physik* **43,** 172–198 (1927).
30. Chandler, D. & Wolynes, P. G. Exploiting the Isomorphism between Quantum Theory and Classical Statistical Mechanics of Polyatomic Fluids. *The Journal of Chemical Physics* **74,** 4078–4095 (1981).
31. Marx, D. & Parrinello, M. Ab Initio Path-Integral Molecular Dynamics. *Zeitschrift für Physik B Condensed Matter* **95,** 143–144 (1994).
32. Hohenberg, P. & Kohn, W. Inhomogeneous Electron Gas. *Physical Review* **136,** B864–B871 (1964).
33. Kohn, W. & Sham, L. J. Self-Consistent Equations Including Exchange and Correlation Effects. *Physical Review* **140,** A1133–A1138 (1965).
34. Coester, F. Bound States of a Many-Particle System. *Nuclear Physics* **7,** 421–424 (1958).
35. Coester, F. & Kümmel, H. Short-Range Correlations in Nuclear Wave Functions. *Nuclear Physics* **17,** 477–485 (1960).
36. Čížek, J. On the Correlation Problem in Atomic and Molecular Systems. Calculation of Wavefunction Components in Ursell-Type Expansion Using Quantum-Field Theoretical Methods. *The Journal of Chemical Physics* **45,** 4256–4266 (1966).
37. Purvis, G. D., III & Bartlett, R. J. A Full Coupled-cluster Singles and Doubles Model: The Inclusion of Disconnected Triples. *The Journal of Chemical Physics* **76,** 1910–1918 (1982).
38. Raghavachari, K., Trucks, G. W., Pople, J. A. & Head-Gordon, M. A Fifth-Order Perturbation Comparison of Electron Correlation Theories. *Chemical Physics Letters* **157,** 479–483 (1989).
39. Clark, S. J., Segall, M. D., Pickard, C. J., Hasnip, P. J., Probert, M. I. J., Refson, K. & Payne, M. C. First Principles Methods Using CASTEP. *Zeitschrift für Kristallographie - Crystalline Materials* **220,** 567–570 (2005).
40. Kresse, G. & Furthmüller, J. Efficient Iterative Schemes for Ab Initio Total-Energy Calculations Using a Plane-Wave Basis Set. *Physical Review B* **54,** 11169–11186 (1996).
41. Jones, J. E. On the Determination of Molecular Fields. I. From the Variation of the Viscosity of a Gas with Temperature. *Proceedings of the Royal Society of London. Series A, Containing Papers of a Mathematical and Physical Character* **106,** 441–462 (1924).

42. Jones, J. E. On the Determination of Molecular Fields. 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* **106,** 463–477 (1924).

43. Weiner, S. J., Kollman, P. A., Case, D. A., Singh, U. C., Ghio, C., Alagona, G., Profeta, S. & Weiner, P. A New Force Field for Molecular Mechanical Simulation of Nucleic Acids and Proteins. *Journal of the American Chemical Society* **106,** 765–784 (1984).

44. Weiner, S. J., Kollman, P. A., Nguyen, D. T. & Case, D. A. An All Atom Force Field for Simulations of Proteins and Nucleic Acids. *Journal of Computational Chemistry* **7,** 230–252 (1986).

45. MacKerell, A. D. *et al.* All-Atom Empirical Potential for Molecular Modeling and Dynamics Studies of Proteins. *The Journal of Physical Chemistry B* **102,** 3586–3616 (1998).

46. Vanommeslaeghe, K. & MacKerell, A. D. CHARMM Additive and Polarizable Force Fields for Biophysics and Computer-Aided Drug Design. *Biochimica et Biophysica Acta (BBA) - General Subjects* **1850,** 861–871 (2015).

47. Daw, M. S. & Baskes, M. I. Embedded-Atom Method: Derivation and Application to Impurities, Surfaces, and Other Defects in Metals. *Physical Review B* **29,** 6443–6453 (1984).

48. Buckingham, R. A. The Classical Equation of State of Gaseous Helium, Neon and Argon. *Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences* **168,** 264–283 (1938).

49. Morse, P. M. Diatomic Molecules According to the Wave Mechanics. II. Vibrational Levels. *Physical Review* **34,** 57–64 (1929).

50. Shaw, D. E. *et al.* Anton 2: Raising the Bar for Performance and Programmability in a Special-Purpose Molecular Dynamics Supercomputer. in *SC '14: Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis* 41–53 (2014). doi:10.1109/SC.2014.9.

51. Schmidt, J., Cerqueira, T. F. T., Romero, A. H., Loew, A., Jäger, F., Wang, H.-C., Botti, S. & Marques, M. A. L. Improving Machine-Learning Models in Materials Science through Large Datasets. *Materials Today Physics* **48,** 101560 (2024).

52. Deng, B., Zhong, P., Jun, K., Riebesell, J., Han, K., Bartel, C. J. & Ceder, G. CHGNet as a Pretrained Universal Neural Network Potential for Charge-Informed Atomistic Modelling. *Nature Machine Intelligence* **5,** 1031–1041 (2023).

53. Levine, D. S. *et al.* The Open Molecules 2025 (OMol25) Dataset, Evaluations, and Models. (2025) doi:10.48550/arXiv.2505.08762.

54. Morrow, J. D., Gardner, J. L. A. & Deringer, V. L. How to Validate Machine-Learned Interatomic Potentials. *The Journal of Chemical Physics* **158,** 121501 (2023).

55. Wang, Z., Wu, H., Sun, L., He, X., Liu, Z., Shao, B., Wang, T. & Liu, T.-Y. Improving Machine Learning Force Fields for Molecular Dynamics Simulations with Fine-Grained Force Metrics. *The Journal of Chemical Physics* **159,** 35101 (2023).

56. Kozinsky, B., Musaelian, A., Johansson, A. & Batzner, S. Scaling the Leading Accuracy of Deep Equivariant Models to Biomolecular Simulations of Realistic Size. in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis* 1–12 (New York, NY, USA, 2023). doi:10.1145/3581784.3627041.

57. Gardner, J. graph-pes: train and use graph-based ML models of potential energy surfaces. https://github.com/jla-gardner/graph-pes (2024) doi:10.5281/zenodo.14956210.

58. Deringer, V. L. & Csányi, G. Machine Learning Based Interatomic Potential for Amorphous Carbon. *Physical Review B* **95,** 94203 (2017).

59. Paszke, A. *et al.* PyTorch: An Imperative Style, High-Performance Deep Learning Library. in *Advances in Neural Information Processing Systems* vol. 32 (2019).

60. Bradbury, J. *et al.* JAX: composable transformations of Python+NumPy programs. (2018).

61. Griewank, A. On Automatic Differentiation. in *Mathematical Programming: Recent Developments and Applications* (eds. Iri, M. & Tanabe, K.) 83–107 (Kluwer Academic Publishers, Dordrecht, 1989).

62. Choromanska, A., Henaff, M., Mathieu, M., Arous, G. B. & LeCun, Y. The Loss Surfaces of Multilayer Networks. (2015) doi:10.48550/arXiv.1412.0233.

63. Li, H., Xu, Z., Taylor, G., Studer, C. & Goldstein, T. Visualizing the Loss Landscape of Neural Nets. in *Advances in Neural Information Processing Systems* vol. 31 (2018).

64. He, K., Zhang, X., Ren, S. & Sun, J. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. (2015).

65. Ba, J. L., Kiros, J. R. & Hinton, G. E. Layer Normalization. (2016) doi:10.48550/arXiv.1607.06450.

66. Ioffe, S. & Szegedy, C. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. in *Proceedings of the 32nd International Conference on Machine Learning* 448–456 (2015).

67. He, K., Zhang, X., Ren, S. & Sun, J. Deep Residual Learning for Image Recognition. (2015) doi:10.48550/arXiv.1512.03385.

68. Izmailov, P., Podoprikhin, D., Garipov, T., Vetrov, D. & Wilson, A. G. Averaging Weights Leads to Wider Optima and Better Generalization. (2019) doi:10.48550/arXiv.1803.05407.

69. Kingma, D. P. & Ba, J. Adam: A Method for Stochastic Optimization. (2017) doi:10.48550/arXiv.1412.6980.

70. Dyson, F. A Meeting with Enrico Fermi. *Nature* **427,** 297 (2004).

71. Sutton, R. The Bitter Lesson. (2019).

72. Rowe, P., Deringer, V. L., Gasparotto, P., Csányi, G. & Michaelides, A. An Accurate and Transferable Machine Learning Potential for Carbon. *The Journal of Chemical Physics* **153,** 34702 (2020).

73. Erhard, L. C., Rohrer, J., Albe, K. & Deringer, V. L. A Machine-Learned Interatomic Potential for Silica and Its Relation to Empirical Models. *npj Computational Materials* **8,** 1–12 (2022).

74. Stone, M. H. The Generalized Weierstrass Approximation Theorem. *Mathematics Magazine* **21,** 167–184 (1948).

75. Park, J. & Sandberg, I. W. Universal Approximation Using Radial-Basis-Function Networks. *Neural Computation* **3,** 246–257 (1991).

76. Steinwart, I. On the Influence of the Kernel on the Consistency of Support Vector Machines. *J. Mach. Learn. Res.* **2,** 67–93 (2002).

77. Rasmussen, C. E. & Williams, C. K. I. *Gaussian Processes for Machine Learning*. (Cambridge, Mass., 2008).

78. Hornik, K., Stinchcombe, M. & White, H. Multilayer Feedforward Networks Are Universal Approximators. *Neural Networks* **2,** 359–366 (1989).

79. Hooker, S. The Hardware Lottery. (2020) doi:10.48550/arXiv.2009.06489.

80. Loshchilov, I. & Hutter, F. SGDR: Stochastic Gradient Descent with Warm Restarts. (2017) doi:10.48550/arXiv.1608.03983.

81. Huss, J. E. & Pennline, J. A. *A Comparison of Five Benchmarks*. (1987).

82. Zhang, Y., Tiňo, P., Leonardis, A. & Tang, K. A Survey on Neural Network Interpretability. *IEEE Transactions on Emerging Topics in Computational Intelligence* **5,** 726–742 (2021).

83. Bigi, F., Langer, M. & Ceriotti, M. The Dark Side of the Forces: Assessing Non-Conservative Force Models for Atomistic Machine Learning. (2025) doi:10.48550/arXiv.2412.11569.

84. Fu, X., Wood, B. M., Barroso-Luque, L., Levine, D. S., Gao, M., Dzamba, M. & Zitnick, C. L. Learning Smooth and Expressive Interatomic Potentials for Physical Property Prediction. (2025) doi:10.48550/arXiv.2502.12147.

85. Wood, B. M., Fu, X., Levine, D. S., Gao, M., Dzamba, M. & Zitnick, C. L. eSEN: Efficient, Scalable, Equivariant Neural Networks for Atomic Systems. (2025) doi:10.48550/arXiv.2503.05635.

86. Neumann, M., Gin, J., Rhodes, B., Bennett, S., Li, Z., Choubisa, H., Hussey, A. & Godwin, J. Orb: A Fast, Scalable Neural Network Potential. (2024) doi:10.48550/arXiv.2410.22570.

87. Darden, T., York, D. & Pedersen, L. Particle Mesh Ewald: An NlogN Method for Ewald Sums in Large Systems. *The Journal of Chemical Physics* **98,** 10089–10092 (1993).

88. Grisafi, A. & Ceriotti, M. Incorporating Long-Range Physics in Atomic-Scale Machine Learning. *The Journal of Chemical Physics* **151,** 204105 (2019).

89. Pozdnyakov, S. N., Willatt, M. J., Bartók, A. P., Ortner, C., Csányi, G. & Ceriotti, M. Incompleteness of Atomic Structure Representations. *Physical Review Letters* **125,** 166001 (2020).

90. Bell, J. S. On the Einstein Podolsky Rosen Paradox. *Physics Physique Fizika* **1,** 195–200 (1964).

91. Chmiela, S., Vassilev-Galindo, V., Unke, O. T., Kabylda, A., Sauceda, H. E., Tkatchenko, A. & Müller, K.-R. Accurate Global Machine Learning Force Fields for Molecules with Hundreds of Atoms. (2022) doi:10.48550/arXiv.2209.14865.

92. Gasteiger, J., Groß, J. & Günnemann, S. Directional Message Passing for Molecular Graphs. (2022) doi:10.48550/arXiv.2003.03123.

93. Schütt, K. T., Unke, O. T. & Gastegger, M. Equivariant Message Passing for the Prediction of Tensorial Properties and Molecular Spectra. (2021) doi:10.48550/arXiv.2102.03150.

94. Batzner, S., Musaelian, A., Sun, L., Geiger, M., Mailoa, J. P., Kornbluth, M., Molinari, N., Smidt, T. E. & Kozinsky, B. E(3)-Equivariant Graph Neural Networks for Data-Efficient and Accurate Interatomic Potentials. *Nature Communications* **13,** 2453 (2022).

95. Simeon, G. & de Fabritiis, G. TensorNet: Cartesian Tensor Representations for Efficient Learning of Molecular Potentials. (2023).

96. Wang, Y., Wang, T., Li, S., He, X., Li, M., Wang, Z., Zheng, N., Shao, B. & Liu, T.-Y. Enhancing Geometric Representations for Molecules with Equivariant Vector-Scalar Interactive Message Passing. *Nature Communications* **15,** 313 (2024).

97. Ramakrishnan, R., Dral, P. O., Rupp, M. & von Lilienfeld, O. A. Quantum Chemistry Structures and Properties of 134 Kilo Molecules. *Scientific Data* **1,** 140022 (2014).

98. Chmiela, S., Tkatchenko, A., Sauceda, H. E., Poltavsky, I., Schütt, K. T. & Müller, K.-R. Machine Learning of Accurate Energy-Conserving Molecular Force Fields. *Science Advances* **3,** e1603015 (2017).

99. Pozdnyakov, S. N. & Ceriotti, M. Incompleteness of Graph Neural Networks for Points Clouds in Three Dimensions. *Machine Learning: Science and Technology* **3,** 45020 (2022).

100. Batatia, I., Batzner, S., Kovács, D. P., Musaelian, A., Simm, G. N. C., Drautz, R., Ortner, C., Kozinsky, B. & Csányi, G. The Design Space of E(3)-Equivariant Atom-Centred Interatomic Potentials. *Nature Machine Intelligence* **7,** 56–67 (2025).

101. Duval, A. *et al.* A Hitchhiker's Guide to Geometric GNNs for 3D Atomic Systems. (2023) doi:10.48550/arXiv.2312.07511.

102. Wang, Z., Liu, G., Zhou, Y., Wang, T. & Shao, B. Efficiently Incorporating Quintuple Interactions into Geometric Deep Learning Force Fields. *Advances in Neural Information Processing Systems* **36,** 77043–77055 (2023).

103. Musaelian, A., Batzner, S., Johansson, A., Sun, L., Owen, C. J., Kornbluth, M. & Kozinsky, B. Learning Local Equivariant Representations for Large-Scale Atomistic Dynamics. *Nature Communications* **14,** 579 (2023).

104. Loose, T. D., Sahrmann, P. G., Qu, T. S. & Voth, G. A. Coarse-Graining with Equivariant Neural Networks: A Path Towards Accurate and Data-Efficient Models. (2023) doi:10.48550/arXiv.2309.09290.

105. Wu, Y. & He, K. Group Normalization. (2018) doi:10.48550/arXiv.1803.08494.

106. Zhang, B. & Sennrich, R. Root Mean Square Layer Normalization. (2019) doi:10.48550/arXiv.1910.07467.

107. Xiong, R. *et al.* On Layer Normalization in the Transformer Architecture. (2020) doi:10.48550/arXiv.2002.04745.

108. Chen, C. & Ong, S. P. A Universal Graph Deep Learning Interatomic Potential for the Periodic Table. *Nature Computational Science* **2,** 718–728 (2022).

109. Drautz, R. Atomic Cluster Expansion for Accurate and Transferable Interatomic Potentials. *Physical Review B* **99,** 14104 (2019).

110. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L. & Polosukhin, I. Attention Is All You Need. (2023) doi:10.48550/arXiv.1706.03762.

111. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D. & Sutskever, I. Language Models Are Unsupervised Multitask Learners. *OpenAI blog* **1,** 9 (2019).

112. Thölke, P. & Fabritiis, G. D. TorchMD-NET: Equivariant Transformers for Neural Network Based Molecular Potentials. (2022) doi:10.48550/arXiv.2202.02541.

113. Liao, Y.-L. & Smidt, T. Equiformer: Equivariant Graph Attention Transformer for 3D Atomistic Graphs. (2023) doi:10.48550/arXiv.2206.11990.

114. Topping, J., Giovanni, F. D., Chamberlain, B. P., Dong, X. & Bronstein, M. M. Understanding Over-Squashing and Bottlenecks on Graphs via Curvature. (2022) doi:10.48550/arXiv.2111.14522.

115. Zhang, C., Bengio, S., Hardt, M., Recht, B. & Vinyals, O. Understanding Deep Learning Requires Rethinking Generalization. (2017) doi:10.48550/arXiv.1611.03530.

116. Kaplan, J. *et al.* Scaling Laws for Neural Language Models. (2020) doi:10.48550/arXiv.2001.08361.

117. Hoffmann, J. *et al.* Training Compute-Optimal Large Language Models. (2022) doi:10.48550/arXiv.2203.15556.

118. Li, C., Ye, Z., Pasini, M. L., Choi, J. Y., Wan, C., Lin, Y. C. & Balaprakash, P. Scaling Laws of Graph Neural Networks for Atomistic Materials Modeling. (2025) doi:10.48550/arXiv.2504.08112.

119. Hernandez, A., Balasubramanian, A., Yuan, F., Mason, S. A. M. & Mueller, T. Fast, Accurate, and Transferable Many-Body Interatomic Potentials by Symbolic Regression. *npj Computational Materials* **5,** 1–11 (2019).

120. Hom, T., Kiszenik, W. & Post, B. Accurate Lattice Constants from Multiple Reflection Measurements. II. Lattice Constants of Germanium Silicon, and Diamond. *Journal of Applied Crystallography* **8,** 457–458 (1975).

121. Baskin, Y. & Meyer, L. Lattice Constants of Graphite at Low Temperatures. *Physical Review* **100,** 544 (1955).

122. George, J., Hautier, G., Bartók, A. P., Csányi, G. & Deringer, V. L. Combining Phonon Accuracy with High Transferability in Gaussian Approximation Potential Models. *The Journal of Chemical Physics* **153,** 44104 (2020).

123. Bartók, A. P., Kermode, J., Bernstein, N. & Csányi, G. Machine Learning a General-Purpose Interatomic Potential for Silicon. *Physical Review X* **8,** 41048 (2018).

124. Hwang, J., Jin, Y. & Lee, J. Machine Learning Force Field Based Phonon Dispersion Prediction. *Current Applied Physics* **66,** 76–80 (2024).

125. Landau, L. D. & Lifshitz, E. M. *Theory of Elasticity*. vol. 7 (New York, 1986).

126. Zuo, Y. *et al.* Performance and Cost Assessment of Machine Learning Interatomic Potentials. *The Journal of Physical Chemistry A* **124,** 731–745 (2020).

127. Wines, D. & Choudhary, K. CHIPS-FF: Evaluating Universal Machine Learning Force Fields for Material Properties. (2025) doi:10.48550/arXiv.2412.10516.

128. Erhard, L. C., Rohrer, J., Albe, K. & Deringer, V. L. Modelling Atomic and Nanoscale Structure in the Silicon–Oxygen System through Active Machine Learning. *Nature Communications* **15,** 1927 (2024).

129. Chmiela, S., Sauceda, H. E., Müller, K.-R. & Tkatchenko, A. Towards Exact Molecular Dynamics Simulations with Machine-Learned Force Fields. *Nature Communications* **9,** 3887 (2018).

130. Duangdangchote, S., Seferos, D. S. & Voznyy, O. Stability and Transferability of Machine Learning Force Fields for Molecular Dynamics Applications. *Digital Discovery* **3,** 2177–2182 (2024).

131. Alder, B. J. & Wainwright, T. E. Phase Transition for a Hard Sphere System. *The Journal of Chemical Physics* **27,** 1208–1209 (1957).

132. Alder, B. J. & Wainwright, T. E. Studies in Molecular Dynamics. I. General Method. *The Journal of Chemical Physics* **31,** 459–466 (1959).

133. Verlet, L. Computer "Experiments" on Classical Fluids. I. Thermodynamical Properties of Lennard-Jones Molecules. *Physical Review* **159,** 98–103 (1967).

134. Beaulieu, Z. F., Deringer, V. L. & Martelli, F. Topological Origin of Peak Splitting in the Structure Factor of Liquid Water. (2026) doi:10.48550/arXiv.2601.05891.

135. Stukowski, A. Visualization and Analysis of Atomistic Simulation Data with OVITO–the Open Visualization Tool. *Modelling and Simulation in Materials Science and Engineering* **18,** 15012 (2009).

136. Schütt, K. T., Kessel, P., Gastegger, M., Nicoli, K. A., Tkatchenko, A. & Müller, K.-R. SchNetPack: A Deep Learning Toolbox For Atomistic Systems. *Journal of Chemical Theory and Computation* **15,** 448–455 (2019).

137. Schütt, K. T., Hessmann, S. S. P., Gebauer, N. W. A., Lederer, J. & Gastegger, M. SchNetPack 2.0: A neural network toolbox for atomistic machine learning. *The Journal of Chemical Physics* **158,** 144801 (2023).

138. Tan, C. W. *et al.* High-Performance Training and Inference for Deep Equivariant Interatomic Potentials. (2025) doi:10.48550/arXiv.2504.16068.

139. Shuaibi, M. *et al.* FAIRChem. https://github.com/facebookresearch/fairchem (2025) doi:10.5281/zenodo.15587498.

140. Pelaez, R. P., Simeon, G., Galvelis, R., Mirarchi, A., Eastman, P., Doerr, S., Thölke, P., Markland, T. E. & De Fabritiis, G. TorchMD-Net 2.0: Fast Neural Network Potentials for Molecular Simulations. *Journal of Chemical Theory and Computation* **20,** 4076–4087 (2024).

141. Ziegler, J. F. & Biersack, J. P. The Stopping and Range of Ions in Matter. *Treatise on Heavy-Ion Science: Volume 6: Astrophysics, Chemistry, and Condensed Matter* 93–129 (1985) doi:10.1007/978-1-4615-8103-1_3.

142. Stillinger, F. H. & Weber, T. A. Computer Simulation of Local Order in Condensed Phases of Silicon. *Physical Review B* **31,** 5262–5271 (1985).

143. Rhodes, B., Vandenhaute, S., Šimkus, V., Gin, J., Godwin, J., Duignan, T. & Neumann, M. Orb-v3: Atomistic Simulation at Scale. (2025) doi:10.48550/arXiv.2504.06231.

144. Kovács, D. P. *et al.* MACE-OFF: Short-Range Transferable Machine Learning Force Fields for Organic Molecules. *Journal of the American Chemical Society* **147,** 17598–17611 (2025).

145. El-Machachi, Z., Frantzov, D., Nijamudheen, A., Zarrouk, T., Caro, M. A. & Deringer, V. L. Accelerated First-Principles Exploration of Structure and Reactivity in Graphene Oxide. *Angewandte Chemie International Edition* **63,** e202410088 (2024).

146. Yang, H. *et al.* MatterSim: A Deep Learning Atomistic Model Across Elements, Temperatures and Pressures. (2024) doi:10.48550/arXiv.2405.04967.

147. Larsen, A. H. *et al.* The Atomic Simulation Environment—a Python Library for Working with Atoms. *Journal of Physics: Condensed Matter* **29,** 273002 (2017).

148. Thompson, A. P. *et al.* LAMMPS - a Flexible Simulation Tool for Particle-Based Materials Modeling at the Atomic, Meso, and Continuum Scales. *Computer Physics Communications* **271,** 108171 (2022).

149. Cohen, O., Riebesell, J., Goodall, R., Kolluru, A., Falletta, S., Krause, J., Colindres, J., Ceder, G. & Gangan, A. S. TorchSim: An Efficient Atomistic Simulation Engine in PyTorch. *AI for Science* **1,** 25003 (2025).

150. Radova, M., Stark, W. G., Allen, C. S., Maurer, R. J. & Bartók, A. P. Fine-Tuning Foundation Models of Materials Interatomic Potentials with Frozen Transfer Learning. *npj Computational Materials* **11,** 237 (2025).

151. Liu, Y., Morrow, J. D., Ertural, C., Fragapane, N. L., Gardner, J. L. A., Naik, A. A., Zhou, Y., George, J. & Deringer, V. L. An Automated Framework for Exploring and Learning Potential-Energy Surfaces. *Nature Communications* **16,** 7666 (2025).

152. Mahmoud, C. B., El-Machachi, Z., Gierczak, K. A., Gardner, J. L. A. & Deringer, V. L. Assessing Zero-Shot Generalisation Behaviour in Graph-Neural-Network Interatomic Potentials. *Digital Discovery* **4,** 3389–3399 (2025).

153. Gardner, J. L. A. *et al.* Distillation of Atomistic Foundation Models across Architectures and Chemical Domains. (2025) doi:10.48550/arXiv.2506.10956.

154. Ouyang, L. *et al.* Training Language Models to Follow Instructions with Human Feedback. (2022) doi:10.48550/arXiv.2203.02155.

155. Wen, X. *et al.* Reinforcement Learning with Verifiable Rewards Implicitly Incentivizes Correct Reasoning in Base LLMs. (2025) doi:10.48550/arXiv.2506.14245.

156. DeepSeek-AI *et al.* DeepSeek-R1: Incentivizing Reasoning Capability in LLMs via Reinforcement Learning. *Nature* **645,** 633–638 (2025).

157. Song, Z. *et al.* Synthetic Datasets for Autonomous Driving: A Survey. *IEEE Transactions on Intelligent Vehicles* **9,** 1847–1864 (2024).

158. Qamar, M., Mrovec, M., Lysogorskiy, Y., Bochkarev, A. & Drautz, R. Atomic Cluster Expansion for Quantum-Accurate Large-Scale Simulations of Carbon. *Journal of Chemical Theory and Computation* https://doi.org/10.1021/acs.jctc.2c01149 (2023) doi:10.1021/acs.jctc.2c01149.

159. Powles, R. C., Marks, N. A. & Lau, D. W. M. Self-Assembly of sp^2-Bonded Carbon Nanostructures from Amorphous Precursors. *Physical Review B* **79,** 75430 (2009).

160. de Tomas, C., Suarez-Martinez, I., Vallejos-Burgos, F., López, M. J., Kaneko, K. & Marks, N. A. Structural Prediction of Graphitization and Porosity in Carbide-Derived Carbons. *Carbon* **119,** 1–9 (2017).

161. Deringer, V. L., Merlet, C., Hu, Y., Lee, T. H., Kattirtzi, J. A., Pecher, O., Csányi, G., Elliott, S. R. & Grey, C. P. Towards an Atomistic Understanding of Disordered Carbon Electrode Materials. *Chemical Communications* **54,** 5988–5991 (2018).

162. Wang, Y., Fan, Z., Qian, P., Ala-Nissila, T. & Caro, M. A. Structure and Pore Size Distribution in Nanoporous Carbon. *Chemistry of Materials* **34,** 617–628 (2022).

163. Bartók, A. P., Kondor, R. & Csányi, G. On Representing Chemical Environments. *Physical Review B* **87,** 184115 (2013).

164. Csányi, G., Winfield, S., Kermode, J. R., De Vita, A., Comisso, A., Bernstein, N. & Payne, M. C. Expressive Programming for Computational Physics in Fortran 95+. *IoP Comput. Phys. Newsletter* Spring2007 (2007).

165. Kermode, J. R. f90wrap: an automated tool for constructing deep Python interfaces to modern Fortran codes. *J. Phys. Condens. Matter* https://doi.org/10.1088/1361-648X/ab82d2 (2020) doi:10.1088/1361-648X/ab82d2.

166. Gardner, J. Synthetic-Data-Experiments. (2023) doi:10.5281/ZENODO.7688015.

167. Harris, C. R. *et al.* Array programming with NumPy. *Nature* **585,** 357–362 (2020).

168. Hinton, G. E., Srivastava, N., Krizhevsky, A., Sutskever, I. & Salakhutdinov, R. R. Improving Neural Networks by Preventing Co-Adaptation of Feature Detectors. (2012) doi:10.48550/arXiv.1207.0580.

169. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I. & Salakhutdinov, R. Dropout: A Simple Way to Prevent Neural Networks from Overfitting. *Journal of Machine Learning Research* **15,** 1929–1958 (2014).

170. Krogh, A. & Hertz, J. A. A simple weight decay can improve generalization. in *Proceedings of the 5th International Conference on Neural Information Processing Systems* 950–957 (Morgan Kaufmann Publishers Inc., Denver, Colorado, 1991).

171. Prechelt, L. Early Stopping - But When?. *Neural Networks: Tricks of the Trade* vol. 1524 55–69 (1998).

172. Le, Q. V., Jaitly, N. & Hinton, G. E. A Simple Way to Initialize Recurrent Networks of Rectified Linear Units. (2015).

173. Pinheiro, J. M. H., Oliveira, S. V. B. de, Silva, T. H. S., Saraiva, P. A. R., Souza, E. F. de, Godoy, R. V., Ambrosio, L. A. & Becker, M. The impact of feature scaling in Machine Learning: Effects on regression and classification tasks. (2025).

174. Wilson, A. G., Hu, Z., Salakhutdinov, R. & Xing, E. P. Deep Kernel Learning. (2015) doi:10.48550/arXiv.1511.02222.

175. Wilson, A. G., Hu, Z., Salakhutdinov, R. & Xing, E. P. Stochastic Variational Deep Kernel Learning. (2016) doi:10.48550/arXiv.1611.00336.

176. Singh, S. & Hernández-Lobato, J. M. Deep Kernel Learning for Reaction Outcome Prediction and Optimization. *Communications Chemistry* **7,** 136 (2024).

177. Rios, M. & Abu-Hanna, A. Deep Kernel Learning for Mortality Prediction in the Face of Temporal Shift. *Artificial Intelligence in Medicine* vol. 12721 199–208 (2021).

178. Gardner, J., Pleiss, G., Weinberger, K. Q., Bindel, D. & Wilson, A. G. GPyTorch: Blackbox Matrix-Matrix Gaussian Process Inference with GPU Acceleration. in *Advances in Neural Information Processing Systems* vol. 31 (2018).

179. Wong, T.-T., Luk, W.-S. & Heng, P.-A. Sampling with Hammersley and Halton Points. *Journal of Graphics Tools* **2,** 9–24 (1997).

180. Barron, J. T. Continuously Differentiable Exponential Linear Units. (2017) doi:10.48550/arXiv.1704.07483.

181. Maaten, L. van der & Hinton, G. Visualizing Data Using T-SNE. *Journal of Machine Learning Research* **9,** 2579–2605 (2008).

182. Pearson, K. LIII. On Lines and Planes of Closest Fit to Systems of Points in Space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* **2,** 559–572 (1901).

183. Gardner, J. L. A., Schulz, H., Helie, J., Sun, L. & Simm, G. N. C. Understanding Multi-Fidelity Training of Machine-Learned Force-Fields. (2025) doi:10.48550/arXiv.2506.14963.

184. Yoo, D., Lee, K., Jeong, W., Lee, D., Watanabe, S. & Han, S. Atomic Energy Mapping of Neural Network Potential. *Physical Review Materials* **3,** 93802 (2019).

185. Chong, S., Grasselli, F., Ben Mahmoud, C., Morrow, J. D., Deringer, V. L. & Ceriotti, M. Robustness of Local Predictions in Atomistic Machine Learning Models. *Journal of Chemical Theory and Computation* **19,** 8020–8031 (2023).

186. Los, J. H. & Fasolino, A. Intrinsic Long-Range Bond-Order Potential for Carbon: Performance in Monte Carlo Simulations of Graphitization. *Physical Review B* **68,** 24107 (2003).

187. Bazant, M. Z. & Kaxiras, E. Modeling of Covalent Bonding in Solids by Inversion of Cohesive Energy Curves. *Physical Review Letters* **77,** 4370–4373 (1996).

188. Marks, N. A. Generalizing the Environment-Dependent Interaction Potential for Carbon. *Physical Review B* **63,** 35401 (2000).

189. Gardner, J. L. A., Beaulieu, Z. F. & Deringer, V. L. Synthetic Data Enable Experiments in Atomistic Machine Learning. *Digital Discovery* https://doi.org/10.1039/D2DD00137C (2023) doi:10.1039/D2DD00137C.

190. Batatia, I., Lin, C., Hart, J., Kasoar, E., Elena, A. M., Norwood, S. W., Wolf, T. & Csányi, G. Cross Learning between Electronic Structure Theories for Unifying Molecular, Surface, and Inorganic Crystal Foundation Force Fields. (2025) doi:10.48550/arXiv.2510.25380.

191. Gardner, J. augment-atoms. https://github.com/jla-gardner/augment-atoms (2025).

192. Kreiman, T., Bai, Y., Atieh, F., Weaver, E., Qu, E. & Krishnapriyan, A. S. Transformers Discover Molecular Structure Without Graph Priors. (2025) doi:10.48550/arXiv.2510.02259.

193. Shapeev, A. V. Moment Tensor Potentials: A Class of Systematically Improvable Interatomic Potentials. *Multiscale Modeling & Simulation* **14,** 1153–1173 (2016).

194. Dao, T., Fu, D. Y., Ermon, S., Rudra, A. & Ré, C. FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness. (2022) doi:10.48550/arXiv.2205.14135.

195. Bharadwaj, V., Glover, A., Buluc, A. & Demmel, J. An Efficient Sparse Kernel Generator for O(3)-Equivariant Deep Networks. (2025) doi:10.48550/arXiv.2501.13986.

196. Geiger, M., Kucukbenli, E., Zandstein, B., Tretina, K. & Corporation, N. cuEquivariance: A CUDA-X Library for Equivariant Neural Network Kernels. https://docs.nvidia.com/cuda/cuequivariance/ (2024).

197. Lee, S. Y., Kim, H., Park, Y., Jeong, D., Han, S., Park, Y. & Lee, J. W. FlashTP: Fused, Sparsity-Aware Tensor Product for Machine Learning Interatomic Potentials. in *Forty-Second International Conference on Machine Learning* (2025).

198. Simm, G. N. C. *et al.* SimPoly: Simulation of Polymers with Machine Learning Force Fields Derived from First Principles. (2025) doi:10.48550/arXiv.2510.13696.

199. Jacobs, R. A., Jordan, M. I., Nowlan, S. J. & Hinton, G. E. Adaptive Mixtures of Local Experts. *Neural Computation* **3,** 79–87 (1991).

200. Shazeer, N., Mirhoseini, A., Maziarz, K., Davis, A., Le, Q., Hinton, G. & Dean, J. Outrageously Large Neural Networks: The Sparsely-Gated Mixture-of-Experts Layer. (2017) doi:10.48550/arXiv.1701.06538.

201. Touvron, H. *et al.* LLaMA: Open and Efficient Foundation Language Models. (2023) doi:10.48550/arXiv.2302.13971.

202. Hinton, G., Vinyals, O. & Dean, J. Distilling the Knowledge in a Neural Network. *arXiv:1503.02531 [cs, stat]* (2015).

203. Cheng, X., Rao, Z., Chen, Y. & Zhang, Q. Explaining Knowledge Distillation by Quantifying the Knowledge. (2020) doi:10.48550/arXiv.2003.03622.

204. Peng, B., Li, C., He, P., Galley, M. & Gao, J. Instruction Tuning with GPT-4. (2023) doi:10.48550/arXiv.2304.03277.

205. Liu, Z., Liu, Q., Li, Y., Liu, L., Shrivastava, A., Bi, S., Hong, L., Chi, E. H. & Zhao, Z. Wisdom of Committee: Distilling from Foundation Model to Specialized Application Model. (2024) doi:10.48550/arXiv.2402.14035.

206. Vemulapalli, R., Pouransari, H., Faghri, F., Mehta, S., Farajtabar, M., Rastegari, M. & Tuzel, O. Knowledge Transfer from Vision Foundation Models for Efficient Training of Small Task-specific Models. in *Proceedings of the 41st International Conference on Machine Learning* 49345–49367 (2024).

207. Morrow, J. D. & Deringer, V. L. Indirect Learning and Physically Guided Validation of Interatomic Potential Models. *The Journal of Chemical Physics* **157,** 104105 (2022).

208. Amin, I., Raja, S. & Krishnapriyan, A. Towards Fast, Specialized Machine Learning Force Fields: Distilling Foundation Models via Energy Hessians. (2025) doi:10.48550/arXiv.2501.09009.

209. Kelvinius, F. E., Georgiev, D., Toshev, A. P. & Gasteiger, J. Accelerating Molecular Graph Neural Networks via Knowledge Distillation. (2023) doi:10.48550/arXiv.2306.14818.

210. Gardner, J. L. A., Baker, K. T. & Deringer, V. L. Synthetic Pre-Training for Neural-Network Interatomic Potentials. *Machine Learning: Science and Technology* **5,** 15003 (2024).

211. Wang, R., Gao, Y., Wu, H. & Zhong, Z. Pre-Training, Fine-Tuning, and Distillation (PFD): Automatically Generating Machine Learning Force Fields from Universal Models. *Physical Review Materials* **9,** 113802 (2025).

212. Vandermause, J., Torrisi, S. B., Batzner, S., Xie, Y., Sun, L., Kolpak, A. M. & Kozinsky, B. On-the-Fly Active Learning of Interpretable Bayesian Force Fields for Atomistic Rare Events. *npj Computational Materials* **6,** 20 (2020).

213. Zhang, L., Lin, D.-Y., Wang, H., Car, R. & E, W. Active Learning of Uniformly Accurate Interatomic Potentials for Materials Simulation. *Physical Review Materials* **3,** 23804 (2019).

214. Rosset, L. A. M., Drabold, D. A. & Deringer, V. L. Signatures of Paracrystallinity in Amorphous Silicon from Machine-Learning-Driven Molecular Dynamics. *Nature Communications* **16,** 2360 (2025).

215. Kovács, D. P., Oord, C. van der, Kucera, J., Allen, A. E. A., Cole, D. J., Ortner, C. & Csányi, G. Linear Atomic Cluster Expansion Force Fields for Organic Molecules: Beyond RMSE. *Journal of Chemical Theory and Computation* **17,** 7696–7711 (2021).

216. Pickard, C. J. & Needs, R. J. Ab Initio Random Structure Searching. *Journal of Physics: Condensed Matter* **23,** 53201 (2011).

217. Deringer, V. L., Proserpio, D. M., Csányi, G. & Pickard, C. J. Data-Driven Learning and Prediction of Inorganic Crystal Structures. *Faraday Discussions* **211,** 45–59 (2018).

218. Robbins, H. & Monro, S. A Stochastic Approximation Method. *The Annals of Mathematical Statistics* **22,** 400–407 (1951).

219. Cheng, B., Engel, E. A., Behler, J., Dellago, C. & Ceriotti, M. Ab Initio Thermodynamics of Liquid and Solid Water. *Proceedings of the National Academy of Sciences of the United States of America* **116,** 1110–1115 (2019).

220. Perdew, J. P., Burke, K. & Ernzerhof, M. Generalized Gradient Approximation Made Simple. *Physical Review Letters* **77,** 3865–3868 (1996).

221. Zhang, Y. & Yang, W. Comment on ``Generalized Gradient Approximation Made Simple''. *Physical Review Letters* **80,** 890 (1998).

222. Adamo, C. & Barone, V. Toward Reliable Density Functional Methods without Adjustable Parameters: The PBE0 Model. *The Journal of Chemical Physics* **110,** 6158–6170 (1999).

223. Goerigk, L. & Grimme, S. A Thorough Benchmark of Density Functional Methods for General Main Group Thermochemistry, Kinetics, and Noncovalent Interactions. *Physical Chemistry Chemical Physics* **13,** 6670–6688 (2011).

224. Grimme, S., Antony, J., Ehrlich, S. & Krieg, H. A Consistent and Accurate Ab Initio Parametrization of Density Functional Dispersion Correction (DFT-D) for the 94 Elements H-Pu. *The Journal of Chemical Physics* **132,** 154104 (2010).

225. Lysogorskiy, Y. *et al.* Performant Implementation of the Atomic Cluster Expansion (PACE) and Application to Copper and Silicon. *npj Computational Materials* **7,** 1–12 (2021).

226. Deng, B., Choi, Y., Zhong, P., Riebesell, J., Anand, S., Li, Z., Jun, K., Persson, K. A. & Ceder, G. Systematic Softening in Universal Machine Learning Interatomic Potentials. *npj Computational Materials* **11,** 9 (2025).

227. Guillot, T. The Interiors of Giant Planets: Models and Outstanding Questions. *Annual Review of Earth and Planetary Sciences* **33,** 493–530 (2005).

228. Helled, R., Mazzola, G. & Redmer, R. Understanding Dense Hydrogen at Planetary Conditions. *Nature Reviews Physics* **2,** 562–574 (2020).

229. Wigner, E. & Huntington, H. B. On the Possibility of a Metallic Modification of Hydrogen. *The Journal of Chemical Physics* **3,** 764–770 (1935).

230. Weir, S. T., Mitchell, A. C. & Nellis, W. J. Metallization of Fluid Molecular Hydrogen at 140 GPa (1.4 Mbar). *Physical Review Letters* **76,** 1860–1863 (1996).

231. Ben Mahmoud, C., Grasselli, F. & Ceriotti, M. Predicting Hot-Electron Free Energies from Ground-State Data. *Physical Review B* **106,** L121116 (2022).

232. Mazitov, A., Bigi, F., Kellner, M., Pegolo, P., Tisi, D., Fraux, G., Pozdnyakov, S., Loche, P. & Ceriotti, M. PET-MAD as a Lightweight Universal Interatomic Potential for Advanced Materials Modeling. *Nature Communications* **16,** 10653 (2025).

233. Pajonk, G. M. Aerogel Catalysts. *Applied Catalysis* **72,** 217–266 (1991).

234. Iswar, S. *et al.* Dense and Strong, but Superinsulating Silica Aerogel. *Acta Materialia* **213,** 116959 (2021).

235. Dunn, B. C., Cole, P., Covington, D., Webster, M. C., Pugmire, R. J., Ernst, R. D., Eyring, E. M., Shah, N. & Huffman, G. P. Silica Aerogel Supported Catalysts for Fischer–Tropsch Synthesis. *Applied Catalysis A: General* **278,** 233–238 (2005).

236. Ben Mahmoud, C., Rosset, L. A., Yates, J. R. & Deringer, V. L. Graph-Neural-Network Predictions of Solid-State NMR Parameters in Silica from Spherical Tensor Decomposition. *The Journal of Chemical Physics* **163,** (2025).

237. Berendsen, H. J. C., Postma, J. P. M., van Gunsteren, W. F., DiNola, A. & Haak, J. R. Molecular Dynamics with Coupling to an External Bath. *The Journal of Chemical Physics* **81,** 3684–3690 (1984).

238. Lee, M. M., Teuscher, J., Miyasaka, T., Murakami, T. N. & Snaith, H. J. Efficient Hybrid Solar Cells Based on Meso-Superstructured Organometal Halide Perovskites. *Science* **338,** 643–647 (2012).

239. Whitfield, P. S., Herron, N., Guise, W. E., Page, K., Cheng, Y. Q., Milas, I. & Crawford, M. K. Structures, Phase Transitions and Tricritical Behavior of the Hybrid Perovskite Methyl Ammonium Lead Iodide. *Scientific Reports* **6,** 35685 (2016).

240. López, C. A. *et al.* Enhanced Stability in CH3NH3PbI3 Hybrid Perovskite from Mechano-Chemical Synthesis: Structural, Microstructural and Optoelectronic Characterization. *Scientific Reports* **10,** 11228 (2020).

241. Eames, C., Frost, J. M., Barnes, P. R. F., O'Regan, B. C., Walsh, A. & Islam, M. S. Ionic Transport in Hybrid Lead Iodide Perovskite Solar Cells. *Nature Communications* **6,** 7497 (2015).

242. Wang, S., Qiao, J., Xu, Y., Guo, X. & Nan, C. First-Principles Study on CH3NH3PbI3 Materials for Perovskite Solar Cells. *Physica B: Condensed Matter* **550,** 347–353 (2018).

243. Ong, K. P., Goh, T. W., Xu, Q. & Huan, A. Structural Evolution in Methylammonium Lead Iodide CH3NH3PbI3. *The Journal of Physical Chemistry A* **119,** 11033–11038 (2015).

244. Bokdam, M. Mixed MAPbI(3-x)Brx Database for Force Field Training. (2023) doi:10.4121/21878661.V2.

245. Fykouras, K. *et al.* Disorder to Order: How Halide Mixing in MAPbI3-xBrx Perovskites Restricts MA Dynamics. *Journal of Materials Chemistry A* **11,** 4587–4597 (2023).

246. Sun, J., Ruzsinszky, A. & Perdew, J. P. Strongly Constrained and Appropriately Normed Semilocal Density Functional. *Physical Review Letters* **115,** 36402 (2015).

247. Nakamura, Y., Shibayama, N., Hayashida, H., Sugimoto, K. & Miyasaka, T. Thermal Expansion Behavior of Halide Perovskite Single Crystals Across a Broad Temperature Range. *European Journal of Inorganic Chemistry* **28,** e202400682 (2025).

248. Jinnouchi, R., Lahnsteiner, J., Karsai, F., Kresse, G. & Bokdam, M. Phase Transitions of Hybrid Perovskites Simulated by Machine-Learning Force Fields Trained on the Fly with Bayesian Inference. *Physical Review Letters* **122,** 225701 (2019).

249. Thomas du Toit, D. F. & Deringer, V. L. Cross-Platform Hyperparameter Optimization for Machine Learning Interatomic Potentials. *The Journal of Chemical Physics* **159,** 24803 (2023).

250. Lennard-Jones, J. E. On the Determination of Molecular Fields. —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* **106,** 463–477 (1924).

251. Kepler, J. *Astronomia Nova Aitiologetos Seu Physica Coelestis, Tradita Commentariis De Motibus Stellae Martis Ex Observationibus G. V. Tychonis Brahe.* (G. Voegelinus, Prague, 1609).

252. Newton, I. *Philosophiæ Naturalis Principia Mathematica.* (Royal Society, London, 1687).

253. Hawking, S. *On the Shoulders of Giants : the Great Works of Physics and Astronomy.* (2002).

254. Mundhenk, T. N., Landajuela, M., Glatt, R., Santiago, C. P., Faissol, D. M. & Petersen, B. K. Symbolic Regression via Neural-Guided Genetic Programming Population Seeding. (2021) doi:10.48550/arXiv.2111.00053.

255. Cranmer, M. Interpretable Machine Learning for Science with PySR and SymbolicRegression.Jl. (2023) doi:10.48550/arXiv.2305.01582.

256. Kahlmeyer, P., Giesen, J., Habeck, M. & Voigt, H. Scaling Up Unbiased Search-based Symbolic Regression. in *Proceedings of the Thirty-ThirdInternational Joint Conference on Artificial Intelligence* 4227–4235 (2024). doi:10.24963/ijcai.2024/471.

257. Reissmann, M., Fang, Y., Ooi, A. S. H. & Sandberg, R. D. Constraining Genetic Symbolic Regression via Semantic Backpropagation. *Genetic Programming and Evolvable Machines* **26,** 12 (2025).

258. Bühler, M. A. & Guillén-Gosálbez, G. KAN-SR: A Kolmogorov-Arnold Network Guided Symbolic Regression Framework. (2025) doi:10.48550/arXiv.2509.10089.

259. Merler, M., Haitsiukevich, K., Dainese, N. & Marttinen, P. In-Context Symbolic Regression: Leveraging Large Language Models for Function Discovery. in *Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (Volume 4: Student Research Workshop)* (eds. Fu, X. & Fleisig, E.) 427–444 (Bangkok, Thailand, 2024). doi:10.18653/v1/2024.acl-srw.49.

260. Cranmer, M., Sanchez Gonzalez, A., Battaglia, P., Xu, R., Cranmer, K., Spergel, D. & Ho, S. Discovering Symbolic Models from Deep Learning with Inductive Biases. in *Advances in Neural Information Processing Systems* vol. 33 17429–17442 (2020).

261. Davis, B. L. & Jin, Z. Discovery of a Planar Black Hole Mass Scaling Relation for Spiral Galaxies. *The Astrophysical Journal Letters* **956,** L22 (2023).

262. Morales-Alvarado, M., Conde, D., Bendavid, J., Sanz, V. & Ubiali, M. Symbolic Regression for Precision LHC Physics. in *Postponed: Machine Learning and the Physical Sciences: Workshop at NeurIPS 2024* (2024).

263. Verstyuk, S. & Douglas, M. R. Machine Learning the Gravity Equation for International Trade. *SSRN Electronic Journal* https://doi.org/10.2139/ssrn.4053795 (2022) doi:10.2139/ssrn.4053795.

264. Kidger, P. On Neural Differential Equations. (2022) doi:10.48550/arXiv.2202.02435.

265. Nocedal, J. Updating Quasi-Newton Matrices with Limited Storage. *Mathematics of Computation* **35,** 773–782 (1980).

266. Varughese, B. *et al.* Physically Interpretable Interatomic Potentials via Symbolic Regression and Reinforcement Learning. (2025) doi:10.48550/arXiv.2511.20506.

267. Coulom, R. Efficient Selectivity and Backup Operators in Monte-Carlo Tree Search. in *Computers and Games* (eds. Hutchison, D. et al.) vol. 4630 72–83 (Berlin, Heidelberg, 2007).

268. Pauli, W. Über den Zusammenhang des Abschlusses der Elektronengruppen im Atom mit der Komplexstruktur der Spektren. *Zeitschrift für Physik* **31,** 765–783 (1925).

269. Pickard, C. J. Ephemeral Data Derived Potentials for Random Structure Search. *Physical Review B* **106,** 14102 (2022).

270. Pareto, V. *Cours D'économie Politique.* (F. Rouge, Lausanne, 1896).

271. Friedel, J. The Distribution of Electrons Round Impurities in Monovalent Metals. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science* **43,** 153–189 (1952).

272. Nicholas, T. C., Stones, A. E., Patel, A., Michel, F. M., Reeder, R. J., Aarts, D. G. A. L., Deringer, V. L. & Goodwin, A. L. Geometrically Frustrated Interactions Drive Structural Complexity in Amorphous Calcium Carbonate. *Nature Chemistry* **16,** 36–41 (2024).

273. Rechtsman, M., Stillinger, F. & Torquato, S. Designed Interaction Potentials via Inverse Methods for Self-Assembly. *Physical Review E* **73,** 11406 (2006).

274. Engel, M. & Trebin, H.-R. Self-Assembly of Monatomic Complex Crystals and Quasicrystals with a Double-Well Interaction Potential. *Physical Review Letters* **98,** 225505 (2007).

275. Sutton, A. P. & Chen, J. Long-Range Finnis–Sinclair Potentials. *Philosophical Magazine Letters* **61,** 139–146 (1990).

276. Assael, M. J., Kalyva, A. E., Antoniadis, K. D., Michael Banish, R., Egry, I., Wu, J., Kaschnitz, E. & Wakeham, W. A. Reference Data for the Density and Viscosity of Liquid Copper and Liquid Tin. *Journal of Physical and Chemical Reference Data* **39,** 33105 (2010).

277. Batchelder, D. N. & Simmons, R. O. Lattice Constants and Thermal Expansivities of Silicon and of Calcium Fluoride between 6 and 322 K. *The Journal of Chemical Physics* **41,** 2324–2329 (1964).

278. McMahon, M. I., Nelmes, R. J., Wright, N. G. & Allan, D. R. Pressure Dependence of the Imma Phase of Silicon. *Physical Review B* **50,** 739–743 (1994).

279. Perdew, J. P., Chevary, J. A., Vosko, S. H., Jackson, K. A., Pederson, M. R., Singh, D. J. & Fiolhais, C. Atoms, Molecules, Solids, and Surfaces: Applications of the Generalized Gradient Approximation for Exchange and Correlation. *Physical Review B* **46,** 6671–6687 (1992).

280. Justo, J. F., Bazant, M. Z., Kaxiras, E., Bulatov, V. V. & Yip, S. Interatomic Potential for Silicon Defects and Disordered Phases. *Physical Review B* **58,** 2539–2550 (1998).

281. Vashishta, P., Kalia, R. K., Rino, J. P. & Ebbsjö, I. Interaction Potential for SiO_2: A Molecular-Dynamics Study of Structural Correlations. *Physical Review B* **41,** 12197–12209 (1990).

282. van Beest, B. W. H., Kramer, G. J. & van Santen, R. A. Force Fields for Silicas and Aluminophosphates Based on Ab Initio Calculations. *Physical Review Letters* **64,** 1955–1958 (1990).

283. Carré, A., Horbach, J., Ispas, S. & Kob, W. New Fitting Scheme to Obtain Effective Potential from Car-Parrinello Molecular-Dynamics Simulations: Application to Silica. *Europhysics Letters* **82,** 17001 (2008).

284. Antao, S., Hassan, I., Wang, J., Lee, P. & Toby, B. State-of-the-Art High-Resolution Powder X-ray Diffraction (HRPXRD) Illustrated with Rietveld Structure Refinement of Quartz, Sodalite, Tremolite, and Meionite. *Canadian Mineralogist* **46,** 1501–1509 (2008).

285. Whitaker, O., Ormrod Morley, D. & Wilson, M. Structural Effects of the Insertion of Large Rings in Two-Dimensional Networks. *The Journal of Chemical Physics* **162,** 114507 (2025).

286. Phan, T. M., Mohanty, P. & Mittal, J. Optimized Protein-Water Interactions and Torsional Refinements Yield Balanced Atomistic Protein Force Fields. *Nature Communications* **16,** 10562 (2025).

287. Li, Y.-F. & Zhang, L.-C. Role of Interatomic Potentials in Molecular Dynamics Simulations of Silicon Nanomachining. *Advances in Manufacturing* **13,** 265–283 (2025).

288. Milinski, A., Dejaegere, A. & Stote, R. Impact of Force Field Polarization on Correlated Motions of Proteins. (2025) doi:10.48550/arXiv.2510.04897.

289. Yudkowsky, E. Artificial Intelligence as a Positive and Negative Factor in Global Risk. *Global Catastrophic Risks* 0 (2008) doi:10.1093/oso/9780198570509.003.0021.

290. Schaeffer, R., Miranda, B. & Koyejo, S. Are Emergent Abilities of Large Language Models a Mirage?. (2023) doi:10.48550/arXiv.2304.15004.

291. Brown, T. *et al.* Language Models Are Few-Shot Learners. in *Advances in Neural Information Processing Systems* vol. 33 1877–1901 (2020).

292. OpenAI *et al.* GPT-4 Technical Report. (2024) doi:10.48550/arXiv.2303.08774.

293. Weiner, P. K. & Kollman, P. A. AMBER: Assisted Model Building with Energy Refinement. A General Program for Modeling Molecules and Their Interactions. *Journal of Computational Chemistry* **2,** 287–303 (1981).

294. Hsu, C., Fannjiang, C. & Listgarten, J. Generative Models for Protein Structures and Sequences. *Nature Biotechnology* **42,** 196–199 (2024).

295. Lewis, S. *et al.* Scalable Emulation of Protein Equilibrium Ensembles with Generative Deep Learning. *Science* **389,** eadv9817 (2025).

296. Wang, Y., Lu, J., Jaitly, N., Susskind, J. & Bautista, M. A. SimpleFold: Folding Proteins Is Simpler than You Think. (2025) doi:10.48550/arXiv.2509.18480.

297. Zeni, C. *et al.* A Generative Model for Inorganic Materials Design. *Nature* **639,** 624–632 (2025).

298. Bone, C., Walker, M., Leng, K., Antunes, L. M., Grau-Crespo, R., Aligayev, A., Dominguez, J. & Butler, K. T. Discovery and Recovery of Crystalline Materials with Property-Conditioned Transformers. (2025) doi:10.48550/arXiv.2511.21299.

299. Klarbring, J. & Walsh, A. Na Vacancy-Driven Phase Transformation and Fast Ion Conduction in W-Doped Na3SbS4 from Machine Learning Force Fields. *Chemistry of Materials* **36,** 9406–9413 (2024).

300. Feng, W. *et al.* Accelerated Machine Learning Force Field for Predicting Thermal Conductivity of Organic Liquids. *Materials Today Energy* **56,** 102178 (2026).

301. Hudson, B. G., Lee, Y.-L., Abernathy, H. W. & Saidi, W. Machine Learning-Accelerated First-Principles Molecular Dynamics Explains Anomalous Lattice Thermal Expansion in BaZr0.78Y0.22O3-\delta. *The Journal of Physical Chemistry Letters* **16,** 8833–8840 (2025).

302. Liu, Y., Madanchi, A., Anker, A. S., Simine, L. & Deringer, V. L. The Amorphous State as a Frontier in Computational Materials Design. *Nature Reviews Materials* **10,** 228–241 (2025).

303. Liu, Y., Zhou, Y., Ademuwagun, R., Walterbos, L., George, J., Elliott, S. R. & Deringer, V. L. Medium-Range Structural Order in Amorphous Arsenic. (2025) doi:10.48550/arXiv.2509.02484.

304. Wu, L. & Deringer, V. L. The Zintl–Klemm Concept in the Amorphous State: A Case Study of Na–P Battery Anodes. *Angewandte Chemie International Edition* e8305 (2025) doi:10.1002/anie.202508305.

305. Morrow, J. D. Data-driven understanding of disordered materials. (University of Oxford, 2024).

306. Zeiler, M. D. & Fergus, R. Visualizing and Understanding Convolutional Networks. (2013) doi:10.48550/arXiv.1311.2901.

307. Ameisen, E. *et al.* Circuit Tracing: Revealing Computational Graphs in Language Models. *Transformer Circuits Thread* https://transformer-circuits.pub/2025/attribution-graphs/methods.html (2025).

308. Lindsey, J. *et al.* On the Biology of a Large Language Model. *Transformer Circuits Thread* https://transformer-circuits.pub/2025/attribution-graphs/biology.html (2025).

309. Okabe, R. *et al.* Virtual Node Graph Neural Network for Full Phonon Prediction. *Nature Computational Science* **4,** 522–531 (2024).

310. Sestak, F., Schneckenreiter, L., Brandstetter, J., Hochreiter, S., Mayr, A. & Klambauer, G. VN-EGNN: E(3)- and SE(3)-Equivariant Graph Neural Networks with Virtual Nodes Enhance Protein Binding Site Identification. (2024).

311. Zhang, Y. & Guo, H. Node-Equivariant Message Passing for Efficient and Accurate Machine Learning Interatomic Potentials. *Chemical Science* https://doi.org/10.1039/D5SC07248D (2026) doi:10.1039/D5SC07248D.

312. Loche, P., Huguenin-Dumittan, K. K., Honarmand, M., Xu, Q., Rumiantsev, E., How, W. B., Langer, M. F. & Ceriotti, M. Fast and Flexible Long-Range Models for Atomistic Machine Learning. *The Journal of Chemical Physics* **162,** 142501 (2025).

313. Kim, D. & Cheng, B. Long-Range Electrostatics for Machine Learning Interatomic Potentials Is Easier than We Thought. (2025) doi:10.48550/arXiv.2512.18029.

314. Kreiman, T. & Krishnapriyan, A. S. Understanding and Mitigating Distribution Shifts For Machine Learning Force Fields. (2025) doi:10.48550/arXiv.2503.08674.

315. Zhao, Q. *et al.* Harnessing Machine Learning to Enhance Transition State Search with Interatomic Potentials and Generative Models. *Advanced Science* **12,** e6240 (2025).

316. Kellner, M. & Ceriotti, M. Uncertainty Quantification by Direct Propagation of Shallow Ensembles. *Machine Learning: Science and Technology* **5,** 35006 (2024).

317. Bigi, F., Chong, S., Ceriotti, M. & Grasselli, F. A Prediction Rigidity Formalism for Low-Cost Uncertainties in Trained Neural Networks. *Machine Learning: Science and Technology* **5,** 45018 (2024).

318. Behler, J. Atom-Centered Symmetry Functions for Constructing High-Dimensional Neural Network Potentials. *The Journal of Chemical Physics* **134,** 74106 (2011).

319. Bochkarev, A., Lysogorskiy, Y. & Drautz, R. Graph Atomic Cluster Expansion for Semilocal Interactions beyond Equivariant Message Passing. *Physical Review X* **14,** 21036 (2024).

320. Goldberg, D. E. & Deb, K. A Comparative Analysis of Selection Schemes Used in Genetic Algorithms. *Foundations of Genetic Algorithms* 69–93 (1991).

321. Fletcher, R. A New Approach to Variable Metric Algorithms. *The Computer Journal* **13,** 317–322 (1970).

322. Odrzywołek, A. All Elementary Functions from a Single Operator. (2026) doi:10.48550/arXiv.2603.21852.

323. Pedregosa, F. *et al.* Scikit-learn: Machine Learning in Python. *Journal of Machine Learning Research* **12,** 2825–2830 (2011).

Appendix

A Smooth Overlap of Atomic Positions (SOAP)

The local environment of an atom is most typically defined as the set of atoms within some cutoff radius. The raw cartesian positions of these atoms do not make for a useful representation of this environment: they are not invariant to rigid rotations or to permutations of the atoms, and their length grows and shrinks as atoms enter and leave the cutoff radius.

"Atomic descriptors" act on sets of atom locations and identities to generate a fixed-length vector of features that "describe" the local environment: similar atomic environments generate similar descriptors, while (very) different atomic environments generate (very) different descriptors. Careful construction ensures that these representations are translationally and rotationally invariant, and smooth with respect to the atomic positions.

Many such descriptors have been proposed in the literature, including *Atom Centred Symmetry Functions* (ACSF, 2007)[318] — used to construct the pioneering MLFF models of *Behler and Parrinello*[6] — the *Smooth Overlap of Atomic Positions* (SOAP, 2010)[163] descriptor — used by *Bartok et al.* to construct Gaussian Approximation Potentials[7] — and most recently the *Atomic Cluster Expansion* (ACE, 2019) descriptor that forms the base for *Ralf Drautz*'s fast and accurate `ACE`[109] models, as well as the `MACE`[10] and `GRACE`[319] message passing architectures.

Below, I sketch the construction of the SOAP descriptor for a single element system; the reader is referred to *Bartok et al.* (2013)[163] for a more detailed, multi-component treatment.

A.1 Definition

As a first step, the (variably sized) local environment is converted into a density field, $\rho(\vec{r}) : \mathbb{R}^3 \rightarrow \mathbb{R}$, by placing a Gaussian of width σ at each atom's position:

$$\rho(\vec{r}) = \sum_{j \in \mathcal{N}_i} \exp\left[-\frac{|\vec{r} - \vec{r}_j|^2}{\sigma^2}\right] \phi(r_{ij})$$

where $\vec{r}_j$ is the position of atom j and $\phi(r_{ij})$ is some smooth envelope function that decays to zero at the cutoff radius.

This function can be expanded exactly as a weighted sum over a product basis spanned by orthogonal radial functions, $R_n(r)$, and the spherical harmonics, $Y_l^m(\hat{\boldsymbol{r}})$:

$$\rho(\boldsymbol{r}) = \sum_{nlm} c_{nlm} R_n(r) Y_l^m(\hat{\boldsymbol{r}})$$

where the expansion coefficients, c_{nlm}, form a permutationally invariant description of the environment. Rotational invariance is achieved by summing over m to yield the "power spectrum", p, with:

$$p_{n,n',l} = \frac{1}{\sqrt{2l+1}} \sum_m c_{nlm} c^{\dagger}_{n'lm}$$

Finally, truncating at some maximum radial and angular orders (n_{max} and l_{max} respectively) yields a fixed-length SOAP vector describing the local environment.

B Gaussian Process Regression

Gaussian Process Regression (GPR) is a non-parametric Bayesian regression technique. The reader is referred to the work of *Rasmussen and Williams* for a thorough treatment.[77] Below, I provide brief qualitative and quantitative overviews.

B.1 Qualitative overview

At the heart of each GPR model is a kernel function, $k(x, x') \in \mathbb{R}$, that quantifies the similarity between two data *locations*, x and x'. By assuming that output values at similar locations are themselves similar, the kernel function induces a prior distribution over possible functions, $f(x)$ — see Figure 1 top for samples drawn from the prior distribution induced by three different kernel functions.

The fitting procedure involves preferentially favouring functions that pass close to the training data, and hence induces a posterior distribution (Figure 1 middle). Model predictions and uncertainties can then be computed as the mean and variance of this posterior distribution at some location of interest (Figure 1 bottom).

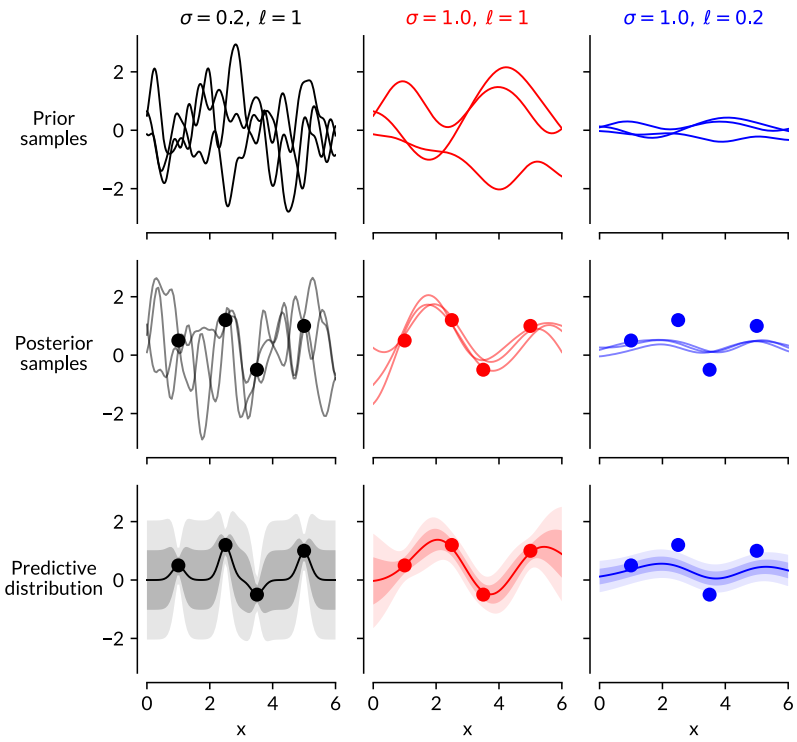


Figure B.1: *GPR models with RBF kernel functions:*

$$k(x, x') = \sigma^2 \exp\left[-\frac{|x - x'|^2}{2l^2}\right]$$

Different hyper-parametrisations (columns, colours) lead to different prior (top) and posterior (middle) distributions over functions, and hence different predictive means and variances (bottom) for the same training data (points)

Choosing the kernel function, and its hyper-parametrisation, is the most important step in GPR modelling. As can be seen in Figure 1, different kernel functions lead to different prior and posterior distributions over functions, and hence different predictive means.

For small datasets, optimal kernel hyper-parameters can be found by maximising the marginal likelihood of the training data. This becomes prohibitively expensive for larger datasets; instead, a combination of heuristics, domain knowledge, and validation-set accuracy are typically used to choose hyper-parameter values in this setting.

B.2 Mathematical details

GPR model generate predictive means as a sum over weighted kernel evaluations:

$$f(x) = \sum_{x' \in \mathcal{M}} \alpha_{x'} \cdot k(x, x')$$

where $\alpha_{x'}$ are (learned) coefficients associated with each location, x', in a set of reference locations, $\mathcal{M}$. Inference cost for these models is therefore linear in the number of reference points considered.

"Training" such GPR models involves finding the optimal coefficients, $\alpha_{x'}$, *i.e.* those that minimise the mean squared loss between predicted and true values on some training set, $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{D}$. "Full" GPR models use every point in the training set as an inducing point, such that $\mathcal{M} = \mathcal{D}$. In this case, the optimal coefficients are found by solving the linear system:

$$\boldsymbol{\alpha} = (\boldsymbol{K}_{\text{DD}} + \boldsymbol{\Sigma})^{-1} \boldsymbol{y}$$

where

- $\boldsymbol{K}_{\text{DD}}$ is the matrix of kernel function evaluations between all pairs of data points in the training set, such that $\boldsymbol{K}_{\text{DD}}[i, j] = k(x_i, x_j)$,
- $\boldsymbol{y}$ is the vector of target values
- $\boldsymbol{\Sigma}$ is a regularisation term

Solving this system typically costs $O(D^3)$ time, and $O(D^2)$ memory, making it infeasible for large datasets.

The Subset of Regressors (SoR) approximation limits the number of reference points to some fixed value, M. These data locations are typically chosen to be the most "representative" of the training set, *e.g.* using uniform, random, or furthest point sampling, and do not necessarily need to be part of the training set. Fitting involves solving the linear system:

$$\boldsymbol{\alpha} = \left[\boldsymbol{K}_{\text{MM}} + \boldsymbol{K}_{\text{MD}} \boldsymbol{\Sigma}^{-1} \boldsymbol{K}_{\text{DM}}\right]^{-1} \boldsymbol{K}_{\text{MD}} \boldsymbol{\Sigma}^{-1} \boldsymbol{y}$$

where

- $\boldsymbol{K}_{\text{MM}}$ is the matrix of kernel function evaluations between all pairs of points in the inducing set,
- $\boldsymbol{K}_{\text{MD}}$ is the matrix of kernel function evaluations between the inducing and training sets ($\mathcal{M}$ and $\mathcal{D}$ respectively),

Solving this system is typically much cheaper (when $M \ll D$), and costs $O(DM^2 + M^3)$ time, and $O(DM + M^2)$ memory, allowing SoR-GPR models to be trained on (much) larger datasets than their full-rank counterparts.

C Symbolic distillation via PySR

In Chapter 5, I use the PySR package[255] to distil trained MLP components into closed-form symbolic expressions. Below, I provide further details on the PySR algorithm and the specific configuration choices made in this work.

C.1 Algorithm overview

PySR is a Python frontend to the `SymbolicRegression.jl` Julia library. It frames the search for symbolic expressions as a multi-population evolutionary optimisation problem, operating on candidate models represented as *abstract syntax trees* (ASTs).

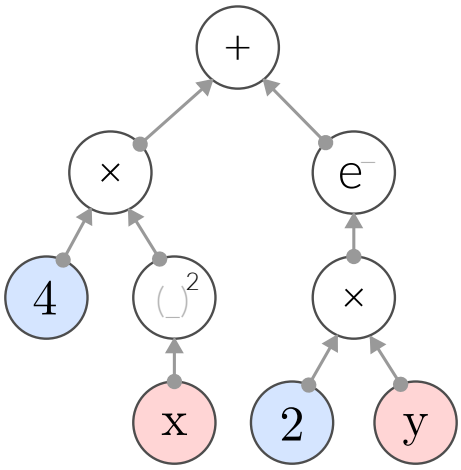


Figure C.2: *an* abstract syntax tree *(AST) representation of the symbolic expression* $4x^2 + e^{2y}$*. Blue leaf nodes correspond to free parameters (constants), while red leaf nodes correspond to input variables. Internal nodes (white) correspond to unary and binary operators that act on their child nodes. The tree is evaluated recursively from the leaf nodes upwards.*

Any symbolic expression can be equivalently represented as an AST, and vice versa. Leaf nodes of the tree are either input variables or free parameters (numerical constants); internal nodes are mathematical operators. These operators are classified as either *unary*, functions of a single argument, such as `exp`, `sin` and `square`, which correspond to internal nodes with one child, or *binary*, functions of two arguments, such as `+`, `-`, `*` and `/`, which correspond to internal nodes with two (ordered) children. Operations involving more than two arguments are represented by nesting: for example, a product of three terms, $a \cdot b \cdot c$, can be encoded as $\times(a, \times(b, c))$. The tree is evaluated recursively from the leaves upwards.

PySR's evolutionary search operates directly on these tree structures as follows:

1. **Initialisation**: multiple independent populations ("islands") of random expression trees are created. Each island evolves in parallel, maintaining diversity and reducing the risk of premature convergence.

2. **Fitness evaluation**: at each generation, every candidate expression is evaluated on the training data. Fitness combines the prediction loss (below, mean squared error) with a complexity penalty that favours simpler expressions, implementing a form of Occam's razor.
3. **Selection**: parents are chosen via *tournament selection*.[320] At each selection event, n individuals are sampled at random from the population (by default, $n = 15$) and ranked by fitness. The best individual is selected with probability p (by default, $p = 0.982$); if not selected, the second-best is chosen with probability $p(1-p)$, and so on, with the selection probability decaying geometrically as $p(1-p)^k$ for the k-th ranked individual. This biases selection strongly towards the fittest candidates while retaining a small chance of selecting less fit individuals, maintaining diversity in the population.
4. **Variation**: offspring are generated through a combination of structural mutations, each selected with a configurable relative weight:
    - *operator mutation*: swapping one operator for another (*e.g.* + → *),
    - *node insertion / deletion*: adding or removing operators and sub-trees, changing the size and shape of the expression,
    - *tree rotation*: rearranging the branching structure of the tree at a random node,
    - *subtree crossover*: exchanging sub-trees between two parent expressions
5. **Constant optimisation**: for each candidate expression with a fixed symbolic structure, the numerical constants are locally optimised via the `BFGS` quasi-Newton method.[321]
6. **Migration**: periodically, the best-performing individuals from one island migrate to another, introducing genetic diversity and helping escape local optima.
7. **Output**: after many generations, a *Hall of Fame* of Pareto-optimal expressions is returned, representing the best trade-off between model complexity (number of nodes in the AST) and prediction accuracy.

C.2 Configuration for this work

The default `PySR` optimisation hyperparameters have been tuned across a wide range of regression problems.[255] I find that they work well for the symbolic distillation tasks in Chapter 5 and therefore leave them unchanged.

The main choice required of the user is the *operator set*. In principle, a single binary operator suffices to represent all elementary functions: the recently proposed `EML` operator, $\mathrm{eml}(x, y) = e^x - \ln(y)$, generates all standard arithmetic, transcendental, and algebraic functions via repeated application.[322] However, the trees (and hence expressions) built from this operator grow rapidly when fitting to non-trivial functions, making its use impractical and undesirable here. Instead, a small but expressive set of operators is therefore preferred.

Since all smooth functions admit a polynomial approximation, addition (`+`), multiplication (`*`) and exponentiation (`pow`) provide a natural minimal basis. To this I add two further operators

motivated by the physics of interatomic potentials: `exp`, which appears extensively in quantum-mechanically inspired models of short-range repulsion (*e.g.* the Buckingham[48] potential), and `cos`, which provides a natural way to express angular dependencies in three-body terms. This gives a set of three binary operators (`+`, `*`, `pow`) and two unary operators (`exp`, `cos`).

Limiting the operator set is important: for an expression tree with n internal nodes, the number of distinct operator labelings is k^n, where k is the total number of available operators. The search space therefore grows exponentially with k, and using a small number of expressive operators dramatically reduces the space of candidate expressions that `PySR` must explore.

C.3 Usage

In practice, `PySR` exposes a `scikit-learn`-compatible[323] interface. The user prepares a tabular dataset: a matrix of input features, $\boldsymbol{X} \in \mathbb{R}^{N \times d}$, and a vector of target outputs, $\boldsymbol{y} \in \mathbb{R}^{N}$, where N is the number of data points and d is the number of input features. For example, when distilling a two-body MLP, $d = 1$ (the interatomic distance, r), and $\boldsymbol{y}$ contains the corresponding MLP predictions; when distilling a three-body MLP, $d = 3$ (two distances and an angle). Search is configured and run via the `PySRRegressor` class:

```python
from pysr import PySRRegressor

model = PySRRegressor(
    binary_operators=["+", "*", "pow"],
    unary_operators=["exp", "cos"],
)
model.fit(X, y)
```

After fitting, `model.equations_` contains the Hall of Fame: a table of candidate expressions ranked by complexity and loss, from which the user can select a final model. I subsequently use `symtorch`,[i] a lightweight library I developed during my DPil, to convert a given symbolic expression into a corresponding `torch.nn.Module` that can be evaluated and optimised as a native `PyTorch` module, facilitating downstream use within `graph-pes`, as well as further optimisation via gradient descent.

```python
from symtorch import symtorchify

f = symtorchify("exp(-1.95*r) * (0.46 + 278.9*exp(-1.95*r))")
f({"r": torch.tensor(2.0)})
# tensor([..], grad_fn=<MulBackward0>)

optimizer = torch.optim.LBFGS(f.parameters())
```

[i] `symtorch` is available at https://github.com/jla-gardner/symtorch.