Linköping University | Department of Physics, Chemistry and Biology
Master's thesis, 30 ECTS | Applied Physics and Electrical Engineering
2021 | LITH-IFM-A-EX–21/3923–SE

# Accelerating bulk material property prediction using machine learning potentials for molecular dynamics

– predicting physical properties of bulk Aluminium and Silicon

---

*Acceleration av materialegenskapers prediktion med hjälp av maskininlärda potentialer för molekylärdynamik*

**Nicholas Sepp Löfgren**

Supervisor : Rickard Armiento
Examiner : Ferenc Tasnádi

LINKÖPING UNIVERSITY

Linköpings universitet
SE–581 83 Linköping
+46 13 28 10 00 , www.liu.se

## Upphovsrätt



## Copyright

**Abstract**

In this project machine learning (ML) interatomic potentials are trained and used in molecular dynamics (MD) simulations to predict the physical properties of total energy, mean squared displacement (MSD) and specific heat capacity for systems of bulk Aluminium and Silicon. The interatomic potentials investigated are potentials trained using the ML models kernel ridge regression (KRR) and moment tensor potentials (MTPs). The simulations using these ML potentials are then compared with results obtained from ab-initio simulations using the gold standard method of density functional theory (DFT), as implemented in the Vienna ab-intio simulation package (VASP). The results show that the MTP simulations reach comparable accuracy compared to the DFT simulations for the properties total energy and MSD for Aluminium, with errors in the orders of magnitudes of meV and $10^{-5}$ Å$^2$. Specific heat capacity is not reasonably replicated for Aluminium. The MTP simulations do not reasonably replicate the studied properties for the system of Silicon.

The KRR models are implemented in the most direct way, and do not yield reasonably low errors even when trained on all available 10000 time steps of DFT training data. On the other hand, the MTPs require only to be trained on approximately 100 time steps to replicate the physical properties of Aluminium with accuracy comparable to DFT. After being trained on 100 time steps, the trained MTPs achieve mean absolute errors in the orders of magnitudes for the energy per atom and force magnitude predictions of $10^{-3}$ and $10^{-1}$ respectively for Aluminium, and $10^{-3}$ and $10^{-2}$ respectively for Silicon. At the same time, the MTP simulations require less core hours to simulate the same amount of time steps as the DFT simulations. In conclusion, MTPs could very likely play a role in accelerating both materials simulations themselves and subsequently the emergence of the data-driven materials design and informatics paradigm.

# Acknowledgments

I want to thank William Stenlund with whom, I, together with our project group in the autumn semester of 2020 developed pieces of code, particularly concerning the calculation of physical properties from molecular dynamics simulations, that I based some code on in this thesis project. I also want to thank the group of theoretical physics at the department of physics, chemistry and biology (IFM) for making me feel like part of the team from my first day onward during this thesis project.

I want to thank Ferenc Tasnádi for being the examiner of my thesis work and for sharing his knowledge, in particular of machine learning potentials. I also want to thank my supervisor Rickard Armiento for providing me with this thesis opportunity, guiding me throughout the work and for his patience answering my questions about things I should have already known.

To friends and family, thank you for your support throughout the years; this would not have been possible without you.

# Contents

# List of Figures

# List of Tables

# 1 Introduction

> The underlying physical laws necessary for the mathematical theory of a large part of physics [...] are thus completely known [...]
>
> Paul Dirac

This master thesis work has been conducted at the Department of Physics, Chemistry and Biology (IFM) at Linköping University. This thesis is examined by Ferenc Tasnádi and supervised by Rickard Armiento. The thesis topic is about the interdisciplinary field of materials science and machine learning, or more specifically, how the tools of machine learning can be used for materials design, discovery and informatics.

This chapter introduces the motivation for this thesis project, as well as previous work that has been made in the field of machine learning used for materials discovery and design. The research questions investigated in this thesis, motivated by previous work, are also presented.

## 1.1 Motivation

In 1929 renowned English physicist Paul Dirac, after the advent of quantum physics, claimed that "the underlying physical laws necessary for the mathematical theory of large parts of physics and 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. It, therefore, becomes desirable that 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 [31]." In the decades after Dirac's bold statement, great amounts of research have been conducted to find approximate solutions to Schrödinger's wave equation for systems of several electrons. Important results have been found, contributing to the solution of this problem of overwhelming complexity as described by Dirac [26].

In recent decades, with the dawn of high performance computers or so-called supercomputers, the high-throughput paradigm of materials research, design and informatics has emerged. By utilising this vast computing power together with state-of-the-art quantum mechanical theory, properties of hypothetical, simulated materials can be calculated in high-throughput and collected into databases. Having materials with desired properties is crucial for the emergence of new, innovative industries and technologies at large. These materials databases can be searched for materials with properties that can fulfil the requirements of new technologies. A novel method for constructing such material databases is using text mining technology to structure vast amounts of unstructured materials science literature data [16].

With these new methods of constructing these large databases, the paradigm of data-driven materials design is emerging.

However, in the paradigm of data-driven materials design, despite the great accuracy of modern Density Functional Density (DFT) methods, they are too expensive for modelling large-scale molecular systems in high-throughput. In these simulations, interatomic potentials are of great importance [39]. The demand of efficiency is significant. In this situation of wanting to simulate large quantities of materials in a reasonable time, it would be relevant to construct interatomic potentials that achieve similar accuracy to DFT, but are less computationally expensive, resulting in faster execution times. This thesis investigates different methods of constructing these less time consuming interatomic potentials using the tools of machine learning (ML).

## 1.2 Previous work

The previous work that is most relevant for this thesis project is mainly the work of moment tensor potentials (MTPs) and other work within the field of ML interatomic potentials for materials science [39, 33, 17, 6, 3, 21, 37]. In previous work, MTPs as a class of ML interatomic potentials have shown great promise [39, 33]. For example, MTPs have been used in an active learning scheme for accelerating high-throughput searches for alloys, reducing the required DFT calculations needed in materials prediction [17]. Other ML methods have also seen success, such as using the principle of conversation of energy to develop an efficient gradient-domain ML approach to construct accurate molecular force fields, once again reducing the necessary amount of ab initio MD calculations required [6]. Another important facet of these emerging ML methods in materials science is descriptors, which are "fingerprints" of the systems investigated, designed to encode information in a format available for the ML methods to use. Choosing an appropriate descriptor for a system and problem is an important aspect of all ML methods [21].

These ML methods are especially interesting from the perspective of the paradigm of high-throughput discovery and design of materials and how it can be accelerated [8, 1]. The adoption of high-throughput methods has invoked an ambition to calculate the properties and predict the existence of essentially any material that can be synthesised and is reasonably long-lived [1]. Part of this ambition is to construct databases that can be queried for essentially any material properties, hopefully yielding an appropriate compound if such a material is available in the database.

## 1.3 Aim

The aim of this thesis project is to investigate the possibilities of using the tools of ML for constructing and training interatomic potentials that can be used for molecular dynamics (MD) simulations. These simulations would be computationally cheaper than ab-initio MD using DFT, and it would therefore be interesting to investigate the possibility to accelerate DFT using ML potentials. If ML potentials reach adequate accuracy compared to DFT but evaluate much faster, that would be an advancement for the high-throughput paradigm of materials science investigation and research. To achieve this aim, a piece of highly modifiable simulation software is developed. This aim of re-implementing MD with the option to use ML potentials has not been readily found in the literature.

## 1.4 Research questions

The thesis project's relevant research questions are presented below.

1. What accuracy is achieved using an interatomic potential obtained from an ML model trained on training data obtained from DFT to predict the next position of the atoms for the next time step in a MD simulation, compared with DFT simulations?
2. Investigate simulating the movement of atoms systems using ML potentials. How well do calculated bulk material properties such as specific heat capacity, total energy, and mean squared displacement using an ML approach compare with values obtained from ab-initio simulation methods based on DFT?
3. Investigate how the ML potential approach performs for a metal and a semiconductor, in particular Aluminium and Silicon, in terms of potential energy and force predictions. To which extent does the utility of the ML potentials allow for simulating different types of materials?

## 1.5 Delimitations

This work is limited to the ML models of kernel ridge regression (KRR) and MTPs. The work is limited to the MTP orders of 06 and 10. The data is limited to data sets of simulated bulk Aluminium and Silicon. These data sets consist of 10000 time steps of the systems' time evolution simulated with DFT methods, using the Vienna Ab initio Simulation Package (VASP) [41].

# 2 Quantum Mechanics

> The atoms or elementary particles themselves are not real; they form a world of potentialities or possibilities rather than one of things or facts.
>
> Werner Heisenberg

*Quantum mechanics* arose from anomalies observed in classical physics, for example the so-called ultraviolet catastrophe, which was the inability of a classical physical model to correctly model ideal black body radiation at thermal equilibrium, especially at short wavelengths. Max Planck's solution to this problem assumed that electromagnetic radiation can be emitted or absorbed in discrete packets, called quanta - which in the case of quantised light, today are called photons [5]. Planck's quantisation of electromagnetic radiation led to the correct form of the spectral distribution functions; the ultraviolet catastrophe was solved. Planck viewed this quantisation as a mathematical trick necessary to resolve issues within classical wave mechanics and that it had no physical meaning [23]. This perception of the quanta has since then changed drastically. Ever since Planck's quantisation of light, physics has been introduced to the world of quanta, and the study of quantum mechanics began to emerge. Worth to note is that Albert Einstein did a similar quantisation in his study of the photoelectric effect. After significant contributions from prominent physicists such as Niels Bohr, Werner Heisenberg, Albert Einstein, Max Born, Erwin Schrödinger and many more, the fundamentals of quantum mechanics were developed by the end of the 1920s [5].

Modern quantum mechanics is formulated in various specially developed mathematical formalisms. The most notable one is the wave function which provides information in the form of probability amplitudes as formulated in Schrödinger's equation

$$i\hbar\frac{\partial}{\partial t}|\Psi(x,t)\rangle = \hat{H}|\Psi(x,t)\rangle, \tag{2.1}$$

where $\Psi(x,t)$ is the position- and time-dependent, information-bearing wave function.

This chapter is about the necessary quantum mechanics background knowledge for understanding and working with physical simulations. The outlining structure of this chapter is inspired by the structure written by Kohn in [26].

## 2.1 Wave mechanics

The wave mechanics formalism of quantum mechanics is one of many formalisms and is widely used. This way of articulating quantum mechanics highlights the wave-particle dualism. The foundation of the theory of the electronic structure of matter is the nonrelativistic Schrödinger equation for the many-electron wave function $\Psi$, which after separation into

a time-dependent and a time-independent function, its time-independent part can be expressed as

$$\left( -\frac{\hbar^2}{2m} \sum_j \nabla_j^2 - \sum_{j,l} \frac{Z_l e^2}{|r_j - R_l|} + \frac{1}{2} \sum_{j \neq j'} \frac{e^2}{|r_j - r'_j| - E} \right) \psi = 0, \tag{2.2}$$

where $r_j$ are the positions of the electrons and $R_l, Z_l$ the positions and atomic numbers of the nuclei. Furthermore, $\hbar, m,$ and $e$ are the conventional fundamental constants, which are the Planck constant expressed in J·s/rad, the mass of the electron and the charge of the electron, respectively. E is the energy. Eq. (2.2) reflects the Born-Oppenheimer approximation, which is the assumption that the wave functions of the nuclei and the electrons in an atomistic system can be treated separately. This is based on the fact that atomic nuclei are significantly heavier than the electrons. All physical properties of the electrons depend parametrically on the $R_l$, in particular the density $n(r)$ and the total energy $E$,

$$n(r) = n(r; R_1, ..., R_N) \tag{2.3}$$

$$E = E(R_1, ..., R_N), \tag{2.4}$$

where $N$ is the number of nuclei.

## 2.2 Density Functional Theory

The structure of this chapter is heavily inspired by Kohn in [26]. The starting point of modern Density Functional Theory (DFT), according to Kohn, was the hypothesis that a knowledge of the ground-state density of $n(r)$ for any electronic system (with or without interactions) uniquely determines the system [26]. The fundamental lemma of the Hohenberg-Kohn formulation of DFT is that the ground-state density $n(r)$ of a bound system of interacting electrons in some external potential $v(r)$ determines this potential uniquely. The density $n(r)$ determines both $N$ and $v(r)$, ignoring an irrelevant additive constant in the potential, which in turn yields the full Hamiltonian $H$ for the electronic system. Hence $n(r)$ implicitly determines all properties derivable from $H$ through the solution of the time-independent Schrödinger equation.

The most important property of an electronic ground state is its total energy $E$. By wave-function methods $E$ could be calculated either by a direct approximate solution of the Schrödinger equation $H\psi = E\psi$ or from the Rayleigh-Ritz minimal principle

$$E = \min_{\tilde{\psi}} (\tilde{\psi}, H\tilde{\psi}), \tag{2.5}$$

where $\tilde{\psi}$ is a normalised trial function for the given number of electrons $N$. In 1964, Hohenberg and Kohn formulated the minimal principle in terms of trial densities $\tilde{n}(r)$, rather than trial wave functions $\tilde{\psi}$. After some derivation, the Hohenberg-Kohn minimum principle is yielded as

$$E = \min_{\tilde{n}(r)} E_v[\tilde{n}(r)] = \min_{\tilde{n}(r)} \left\{ \int v(r)\tilde{n}(r)dr + F[\tilde{n}(r)] \right\}, \tag{2.6}$$

where $F[\tilde{n}(r)] \equiv \min_\alpha [\psi^\alpha_{\tilde{n}(r)}, (T + U)\psi^\alpha_{\tilde{n}(r)}]$ and $\psi^\alpha_{\tilde{n}(r)}$ is the class of trial functions with a fixed trial $\tilde{n}(r)$. $F[\tilde{n}(r)]$ is a universal functional of the density $\tilde{n}(r)$, and $T$ is the kinetic energy and $U$ is the interaction energy. This formulation of the minimal principle (2.6) in terms of the functional $F[\tilde{n}(r)]$ is a more concise formulation due to Levy (1982) and Lieb (1982). Furthermore, this so called Hohenberg-Kohn (HK) minimum principle may also be considered as the formal exactification of Thomas-Fermi theory, since Thomas-Fermi (TF) theory can be derived from it after some approximations [26].

Soon after the publication of TF theory in 1928, Hartree proposed a set of self-consistent single-particle equations for the approximate description of the electronic structure of atoms that described atomic ground states better than TF theory. Kohn together with Sham tasked themselves with extracting the Hartree equations from the HK variational principle eq. (2.6), since this was formally exact and therefore had to have the Hartree equations and improvements implicitly in them. They successfully did so, and obtained self-consistent equations that now are called the Kohn-Sham (KS) equations which are

$$\left(-\frac{1}{2}\nabla^2 + v_{eff}(r) - \epsilon_j\right)\phi_j(r) = 0 \tag{2.7}$$

with

$$n(r) = \sum_{j=1}^{N} |\phi_j(r)|^2, \tag{2.8}$$

$$v_{eff}(r) = v(r) + \int \frac{n(r')}{|r - r'|}dr' + v_{xc}(r), \tag{2.9}$$

where $v_{eff}$ is an effective external potential where the system of noninteracting particles is moving and $v_{xc}(r)$ is the local exchange-correlation potential. $v_{xc}(r)$ depends functionally on the entire density distribution $\tilde{n}(r)$ as given by

$$v_{xc}(r) \equiv \frac{\delta}{\delta\tilde{n}(r)} E_{xc}[\tilde{n}(r)]_{\tilde{n}(r)=n(r)}. \tag{2.10}$$

Minimising the density $n(r)$ is given by solving the single particle equation given by eq. (2.7). Knowing this, the ground state energy is given by

$$E = \sum_j \epsilon_j + E_{xc}[n(r)] - \int v_{xc}(r)n(r)dv - \frac{1}{2}\int \frac{n(r)n(r')}{|r - r'|}dr. \tag{2.11}$$

Furthermore, if one neglects the terms $E_{xc}$ and $v_{xc}$, the KS eqs. (2.7)-(2.9) reduce to the self-consistent Hartree equations and furthermore, the KS equations can be regarded as the formal exactification of the Hartree equations. With the exact $E_{xc}$ and $v_{xc}$ all many-body effects are in principle included, which highlights the importance of the functional $E_{xc}[\tilde{n}(r)]$. The practical usefulness of ground-state DFT depends entirely on whether approximations of the functional $E_{xc}[\tilde{n}(r)]$ can be found which are simultaneously sufficiently low in required calculations and sufficiently accurate [26].

So far a purely mathematical framework for viewing electronic structure from the perspective of the electron density $n(r)$ has been presented. As mentioned above, the concrete usefulness of DFT requires efficient approximations for $E_{xc}[n(r)]$ in the KS formulation and for $F[n(r)]$ in the HK formulation. These approximations reflect the physics of electronic structure and come from outside of DFT [26]. Since the KS formulation is more extensively used, the following considerations of approximations will concern the exchange-correlation energy functional $E_{xc}$.

The most common approximations for $E_{xc}[n(r)]$ have a quasilocal form, which can be expressed as

$$E_{xc}[n(r)] = \int e_{xc}(r; [n(\tilde{r})])n(r)dr, \tag{2.12}$$

where $e_{xc}(r; [n(\tilde{r})])$ represents an exchange-correlation energy per particle the point $r$, which is a functional of the density distribution $n(\tilde{r})$. The simplest, and surprisingly effective, approximation for $E_{xc}[n(r)]$ is the so-called *local-density approximation* (LDA),

$$E_{xc}^{LDA} = \int e_{xc}(n(r))n(r)dr, \tag{2.13}$$

where $e_{xc}(n(r))$ is the exchange-correlation energy per particle of a uniform electron gas of density $n$. The exchange part is given, in atomic units, by

$$e_x(n) \equiv -\frac{0.458}{r_s}, \tag{2.14}$$

where $r_s$ is the radius of a sphere containing one electron and given by $(4\pi/3)r_s^3 = n^{-1}$. The correlation part was first estimated to

$$\epsilon_c(n) = -\frac{0.44}{r_s + 7.8}. \tag{2.15}$$

Remarkably, even though LDA is only constructed to be exact for a uniform electron gas, nevertheless it has been found to give extremely useful results for most applications. This has been at least partly rationalised by the observation that the LDA satisfies a sum rule which expresses normalisation of the exchange-correlation hole.

LDA is the base of almost all approximations currently in use in DFT. One such approximation, that is also commonly used, is the *generalised gradient approximation* (GGA). This approximation is born from the following expression for the exhange-correlation energy

$$E_{xc} = -\frac{1}{2}\int dr n(r)\overline{R}_{xc}^{-1}(r,[n(\tilde{r})]), \tag{2.16}$$

where

$$\overline{R}_{xc}^{-1}(r,n(\tilde{r})) \equiv \int dr' \frac{-\overline{n}_{xc}(r,r'[n(\tilde{r})])}{|r-r'|}, \tag{2.17}$$

is the moment of degree $(-1)$ of $-\overline{n}_{xc}(r,r')$, i.e. minus the inverse radius of the $\lambda$-averaged $xc$ hole [28]. Comparing eqs. (2.16) and (2.12) yields the very physical, formally exact relation

$$e_{xc}(r;[n(\tilde{r})]) = -\frac{1}{2}R_{xc}^{-1}(r;[n(r)]). \tag{2.18}$$

The average $xc$ hole distribution around a given point $r$ is an important concept for understanding more complex approximations than the LDA. The physical $xc$ hole is given by

$$n_{xc}(r,r') = g(r,r') - n(r'), \tag{2.19}$$

where $g(r,r')$ is the conditional density at $r'$ given that one electron is at $r$. It describes the "hole" dug into the average density $n(r')$ by the electron at $r$, which further is normalised

$$\int n_{xc}(r,r')dr' = -1, \tag{2.20}$$

which reflects a total "screening" of the electron at $r$, and is localised due to the combined effect of the Pauli principle and electron-electron interaction. To define the average $xc$ hole, which is used in eq. (2.17), one introduces a fictitious Hamiltonian $H_\lambda, 0 \leqslant \lambda \leqslant 1$, for the many body system which differs from the physical Hamiltonian, $H_{\lambda=1}$, by the two replacements

$$\frac{e^2}{|r_i - r_j|} \longrightarrow \frac{\lambda e^2}{|r_i - r_j|}, \tag{2.21}$$

$$v(r) \longrightarrow v_\lambda(r), \tag{2.22}$$

where the fictitious $v_\lambda(r)$ is chosen so that for all $\lambda$ in the interval $(0,1)$ the corresponding density equals the physical density $n(r)$

$$n_\lambda(r) \equiv n_{\lambda=1}(r) = n(r). \tag{2.23}$$

The procedure of replacing terms in the physical Hamiltonian according to eqs. (2.21) and (2.22) represents an interpolation between the KS system ($\lambda = 0$) and the physical system ($\lambda = 1$), in which $n_{\lambda=1}(r) = n(r)$. The *average xc* hole density $\overline{n}(r, r')$ is then defined as

$$\overline{n}_{xc}(r, r') = \int_0^1 d\lambda n_{xc}(r, r'; \lambda). \tag{2.24}$$

Since $R_{xc}^{-1}(r)$ is an functional of $n(r)$ and which is expected to be predominantly short-sighted, $n(\tilde{r})$ can be formally expanded around the point $r$ which is taken to be the origin:

$$n(\tilde{r}) = n + n_i \tilde{r}_i + \frac{1}{2} \sum n_{ij} \tilde{r}_i \tilde{r}_j + ..., \tag{2.25}$$

where $n \equiv n(0)$, $n_i \equiv \nabla_i n(r)|_{r=0}$, etc., and then consider $R_{xc}(r)$ as a function of the coefficients $n, n_i, n_{ij}, ....$ Ordering in powers of the differential operators and respecting the scalar nature of $R_{xc}^{-1}$ yields

$$R_{xc}^{-1}(r) = F_0(n(r)) + F_{21}(n(r))\nabla^2 n(r) + F_{22}(n(r)) \times \sum (\nabla_i n(r))(\nabla_i n(r)) + .... \tag{2.26}$$

Eq. (2.26) substituted into eq. (2.16) for $E_{xc}$ leads to the gradient expansion

$$E_{xc} = E_{xc}^{LDA} + \int G_2(n)(\nabla n)^2 dr + \int [G_4(n)(\nabla^2 n)^2 + ...] dr + ..., \tag{2.27}$$

where $G_2(n)$ is a universal functional of $n$ [27]. Somewhat unfortunately, in application to real systems this expansion has generally been disappointing, often worsening the results of LDA.

To derive GGA, one begins with the series (2.25) [35]. This series can be formally re-summed for increasing terms to result in the following expressions of increasing exactness

$$E_{xc}^0 = \int \epsilon(n(r)) n(r) dr \qquad \text{(LDA)}, \tag{2.28}$$

$$E_{xc}^{(1)} = \int f^{(1)}(n(r), |\nabla n(r)|) n(r) dr \qquad \text{(GGA)}, \tag{2.29}$$

$$E_{xc}^{(2)} = \int f^{(2)}(n(r), |\nabla n(r)|) \nabla^2 n(r) dr. \tag{2.30}$$

$E_{xc}^0$ is the LDA, and requires the independently calculated function of one variable, $x \equiv n$. $E_{xc}^1$ is the GGA, and requires the independently calculated function of two variables, $x \equiv n, y \equiv |\nabla n|$, and so forth.

The LDA and GGA approximations are two of the most used approximations for $E_{xc}$ in the KS formulation, and are widely used in DFT. It requires a lot of thoughtful work to derive successful GGAs of the form of eq. (2.29), but a lot of important progress has been made. The use of GGAs in the place of the LDA has reduced errors of atomization energies of standard sets of molecules, consisting of light atoms, by factors of typically $3 - 5$. The remaining errors are typically $\pm(2 - 3)$ kg moles per atom, about twice as high as for the best performing wave-function methods in 1999 [26]. The improved accuracy in combination with the previously emphasised capability of DFT to deal with large systems of atoms, has over a period of relatively few years beginning about 1990, made DFT a significant and important part of quantum chemistry.

# 3 Molecular Dynamics

> All the effects of Nature are only the mathematical consequences of a small number of immutable laws.
>
> Pierre-Simon Laplace

Computer simulations are a vital tool for bridging theory and experiments. They can, for example be used to predict results from theory that can be verified with experiments. Simulations help in understanding the basic mechanisms of physics and can verify that the current models are correct and sufficient. Experiments provide detailed information about initial and final states, but basically no information about transient processes. This gap can be bridged by simulations, which also offer a great opportunity for visualisation of transient states. Systems of two electrons can be solved analytically, and to study systems of more electrons numerical solutions are required. However, because of the "exponential wall" when increasing the electrons in a system, even numerical solutions quickly become unfeasible and approximative simulations are unavoidably required [31, 26]. In the field of materials science, problems involving many-body systems in transient states are often of interest and simulations prove to be essential. The limiting factors of simulations are CPU and memory, which impose limitations on system size and time scale.

*Molecular Dynamics* (MD) is a simulation method where the trajectories of atoms and molecules are determined by numerically solving Newton's equations of motion for a system of interacting particles, where forces between particles and their potential energies are often calculated using interatomic potentials [14]. This is what is most commonly meant with MD, and that is what has been used in this thesis.

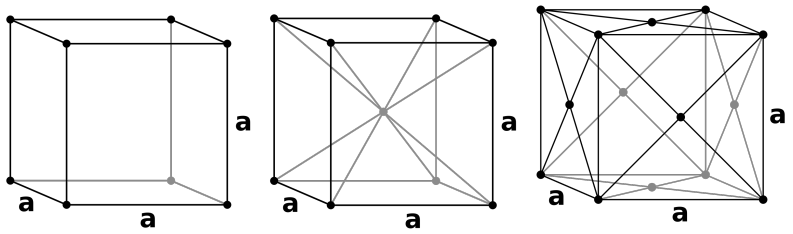
As a curiosity, MD has also been termed Laplace's vision of Newtonian mechanics of predicting the future by animating nature's forces, which is connected to the concept of causal determinism derived from Newtonian mechanics as formulated by Laplace in his book "A philosophical essay on probabilities" [29]. Without knowledge of statistical mechanics, and from it derived physics, Newtonian mechanics imply a sort of scientific determinism. Laplace formulated this as the idea of an intellect, often retrospectively called a demon, which if it knows the precise position and momenta of every atom in the universe, all their past and future values for any given time could be calculated and therefore known. Laplace's demon, as this idea is often called, motivated coming generations of physicists and the subsequent development of statistical mechanics, and later quantum mechanics, offered several repudiations to this idea. Examples of arguments against Laplace's demon are thermodynamic irreversibility and the second law of thermodynamics, since Laplace's demon is based on the premise of reversibility. Furthermore, Laplace's demon makes a canonical assumption of

determinism, which makes it incompatible with the Copenhagen interpretation of quantum mechanics, which stipulates indeterminacy [13].

## 3.1 Crystal structures

One essential part of simulations is of course the materials that are being studied over their time evolution. The crystal structure of such a material is highly relevant for its physical properties. The crystal structure of a material is a description of the ordered arrangement of atoms, ions or molecules in the crystalline material. The smallest group of particles in a material that constitutes a repeating pattern along the principal axes of three-dimensional space in the matter is called the *unit cell* of the structure. The unit cell completely reflects the symmetry and structure of the entire crystal, since the material is constructed by these unit cells repetitively translated along its principal axes. The concept of repeating unit cells that fill space is an expansion of the concept of *Bravais lattice*, which is the periodic repetitions in space along some primitive basis vectors, which, in turn, do not need to be mutually perpendicular [25].

The geometry of the unit cell is defined as a parallelepiped, providing six lattice parameters which are the lengths of the cell edges $(a, b, c)$ and the angles between them $(\alpha, \beta, \gamma)$.




(a) Simple cubic (b) Body-centered cubic (c) Face-centered cubic

Figure 3.1: Different cubic unit cells. Reproduced from [11].

Fig. 3.1 shows three different cubic unit cells, where of course all the angles $(\alpha, \beta, \gamma)$ equal 90° and all the lengths of cell edges $(a, b, c)$ are the same, in this case length $a$. There are many more crystal structures materials can be in, and crucially in many ways decide the material's properties.

## 3.2 Simulations

The physical foundation of simulation of materials is statistical mechanics [14]. Ergodicity is the fundamental assumption of classical statistical mechanics and furthermore the ergodic hypothesis is the basis of computer simulations. The ergodic hypothesis states that, over long periods of time, the time spent by a system in some region of the phase space of microstates with the same energy is proportional to the volume of the region. Ergodicity is discussed further in subsection 3.2 Ergodicity. Any computer simulation starts with a valid statistical ensamble; in a valid ensamble each thermodynamic variable is fixed. There are many ensambles one could choose for the simulation, for example the microcanonical ensamble (constant number of particles $N$, volume $V$ and total energy $E$) or the canonical (constant number of particles $N$, volume $V$ and temperature $T$).

MD is a completely deterministic simulation method in where the time evolution of systems are followed by solving Newton's equation of motion,

$$m\frac{d^2r}{dt^2} = F(r) = -\nabla V(r). \tag{3.1}$$

There are many different ways to integrate the equation of motion and get the forces, for example some common integrator algorithms are the Verlet and Langevin integrators.

An important component of MD when simulating bulk materials is periodic boundary conditions (PBC), which allows for repeating a material's unit cell in space along a set of axes. In this thesis, for the purpose of studying the macro-scale behaviour of bulk materials, unit cells have been repeated in three dimensions. For different purposes, PBC can be applied to repeat a unit cell in one or two dimensions, for example to simulate a surface.

The most common MD algorithm is presented schematically in figure 3.2. Furthermore, the MD algorithm is presented as pseudocode in listing 3.1

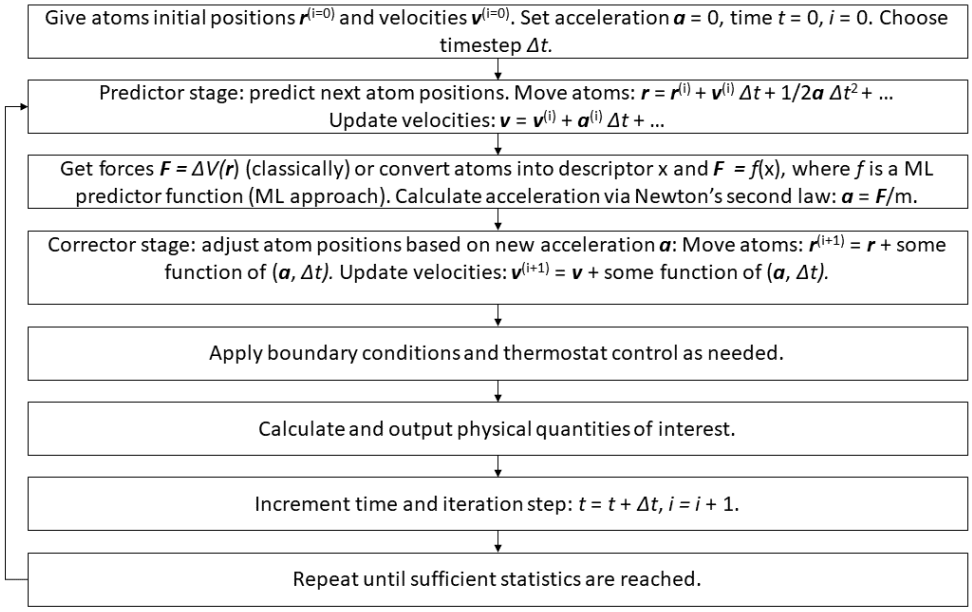



Figure 3.2: Schematic representation of MD.

```
1  program MD                   # simple MD program
2
3  call init                    # initialization
4  t=0
5  do while (t.lt.max)          # MD loop
6      call force(f,en)         # determines the forces
7      call integrate(f,en)     # integrate equations of motion
8      t=t+delt
9      call sample              # sample averages
10 enddo
11 stop
12 end
```

Listing 3.1: MD pseudocode. Reproduced from [14].

### Ergodicity

A base assumption of simulations is that every representative point of a system eventually visits the entire volume of the system [14]. Averages in thermal equilibrium over the phase space of a system, i.e. *ensamble* averages, is a way of studying the average behaviour of a

system in a purely static sense. The phase space is a space in which all possible states of a system are represented, and depending on what is studied, a point in phase space may be said to be a microstate of the system. The idea behind MD when studying the dynamics of a system is to take the average of the quantity of interest over the whole phase space for sufficiently many time steps when solving eq. (3.1) to numerically calculate the time evolution of the system. For these time averages to have physical meaning they have to be calculated over sufficiently long time so that the average does not depend on the initial conditions. An assumption of *ergodicity* is often made for systems studied in computer simulations, which is an assumption that is not generally true. There are many examples of systems that are not ergodic in practice, e.g. glasses and metastable phases, or even in principle, such as nearly all harmonic solids [14].

Ergodicity is the concept that a point in a dynamical system will eventually visit all parts of the space that the system moves in. In such process, the system will lose memory of its initial state, and relax towards equilibrium regardless of its initial configuration after long enough time. Therefore, in ergodic systems the average behaviour can be inferred from the trajectory of a "typical" trajectory. This is furthermore expressed in the so called ergodic hypothesis which is often assumed in computational physics. The hypothesis states that the average of a property $A$ over time equals the average over the statistical ensamble for the same property

$$\overline{A(\mathbf{r}_i)} = \langle A(\mathbf{r}_i) \rangle_{NVE}, \tag{3.2}$$

where $\mathbf{r}_i$ is the distance vector to the $i$th atom in the system, $\overline{A}$ is the time average of the property $A$ and $\langle A \rangle_{NVE}$ is the ensamble average in the microcanonical ensamble.

## 3.3 Potentials

The classic interatomic potentials used in MD are usually developed with great consideration, since the interaction potentials are the most important components in classical MD, as they define how the particles in the simulation will interact. The accuracy of the predictions depend greatly on the interaction quality. An interatomic potential can be written as a series expansion of functional terms that depend on the position of a select amount of atoms at a time. Then the total potential of the system $V_{total}$ can be written as

$$V_{total} = \sum_{i}^{N} V_1(\mathbf{r}_i) + \sum_{i,j}^{N} V_2(\mathbf{r}_i, \mathbf{r}_j) + \sum_{i,j,k}^{N} V_3(\mathbf{r}_i, \mathbf{r}_j, \mathbf{r}_k) + ... \tag{3.3}$$

where $V_i$ is the $i$-body term, $N$ is the number of atoms in the system and $\mathbf{r}_i$ is the position of atom $i$. A fundamental assumption is that this series converges rapidly and often higher order terms above order two are neglected. Based on this expansion, it is common to discuss pair potentials, i.e. potentials of exclusively two-body terms, and many-body potentials, i.e. potentials of three-body terms or higher. Some examples of classical potentials commonly used in MD are the Lennard-Jones potential, and Morse potential.

The Lennard-Jones potential is a pair potential, and is given by

$$V_{LJ} = 4\epsilon \left[ \left( \frac{\sigma}{r} \right)^{12} - \left( \frac{\sigma}{r} \right)^{6} \right], \tag{3.4}$$

where $r$ is the distance between two interacting particles, $\epsilon$ is the depth of the potential well, and $\sigma$ is the distance at which the particle-particle potential energy $V$ is zero. $\sigma$ is also referred to as the size of the particle. The Lennard-Jones potential is a simple model which yields realistic intermolecular interactions. The Lennard-Jones potential models soft repulsive and attractive interactions, which means that it describes electronically neutral particles. Because of its mathematical simplicity and reliable results, the Lennard-Jones potential is one of the most extensively studied interatomic potentials [40].

# 4 Machine Learning

> It is a part of probability that many improbable things will happen
>
> Aristotle

*Machine learning* is a field of study concerned with how computer algorithms can learn from data, which grew out as its own discipline from the quest of artificial intelligence [4]. Machine learning has in recent decades become its own field for solving problems in the realms of, for example, finance, computer vision, and materials science. In its essence, machine learning is about algorithms as learners, that can generalise from experience to a sufficient accuracy depending on the problem of interest. There are different approaches used in machine learning, where the three most significant ones are

- Supervised learning - the algorithm is trained on labelled data where the goal is for the algorithm to map input to correct label output. An illustrative example of supervised learning is presented in section 4.1.
- Unsupervised learning - the algorithm is presented with data that is not labelled, and the goal of the algorithm is to structure the data on its own. For example, presenting an unsupervised learning algorithm with data about a set of songs, with the goal for the algorithm to cluster them into musical genres.
- Reinforcement learning - the algorithm is left to interact with a dynamic environment in which it must perform a certain goal, for example playing a game against an opponent. In this dynamic environment, the algorithm is provided feedback on its performance. This performance is fed back into the algorithm as some kind of reward metric, which the algorithm tries to consequently maximise. One example of a machine trained in this way is Deepmind's AlphaGo, which is the first computer program to defeat a professional player in the complex, intuitive game of go [10]. Another example is OpenAI Five, which was a project where a team of bots were trained in the five-on-five video game Dota 2 [34].

In this thesis, supervised learning is used to train interatomic potentials. The two methods of supervised learning investigated are presented in the subsections 4.4 and 4.5, which are the KRR and MTP methods respectively. In essence, supervised learning is about training a machine to find the relationship between the properties $x$ and $y$ by presenting it training data where the connection between the training property $x$ and target property $y$ is more or less clear. In a generic machine, there is some set of parameters $\theta$ which are modified during training according to some training algorithm (usually some kind of minimisation of loss)

where the goal is to approximate the relationship as best as possible. The prediction function of the machine can be described as the function $f$ as follows:

$$f(x, \theta) = y. \tag{4.1}$$

In supervised training, it is common practice to divide the available data into at least a training set and a test set. Additional data divisions might be of interest for a more advanced ML process, for example also using part of the data as a validation data set for trimming an ML model's hyperparameters, but dividing data into training and test data is often the first step. The machine is trained on the training data and set to predict each target property $y$ for each $x$ in the test data. By comparing the machine's predictions with the actual values of the target properties in the test data, some kind of error for the machine's predictive power can be calculated. What this error measure is depends on what kind of property $y$ is; if $y$ for example is some continuous variable such as the length of a person, *mean absolute error* (MAE) could be an appropriate error metric.

In this work, the property $x$ has been some kind of description of an atomic system and the target property $y$ has either been the system's total potential energy or the forces acting on each atom.

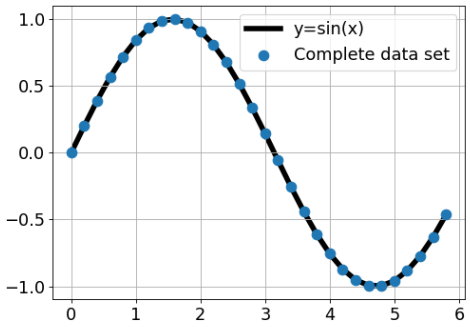
## 4.1 Illustrative machine learning example

In this section, an illustrative example of a supervised learning ML problem is presented, which is inspired by Bishop [4]. Furthermore, the figures in this example are redrawn as inspired by Bishop. In supervised learning, there are two main types of problems, related to what the desired output is. Problems in which it is desired to map each input vector to one of a finite number of discrete categories are called *classification* problems. An example of such a classification problem is training a machine to classify whether or not images of skin lesions show malignancy, potentially automating detection of skin cancer and other conditions [24]. Problems in which the desired output is one or more continuous variables are called *regression* problems. In this section, a simple regression problem is presented where polynomial curve fitting is used to learn a sinusoidal behaviour, highlighting key concepts of ML.

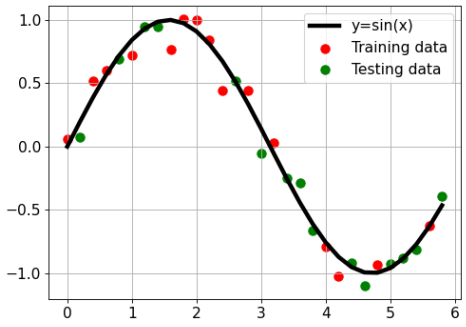
Unavoidably, an ML process requires *data*. A very basic principle is to split the available data into two partitions: a training data set and a testing data set (the testing data is also often called validation data). If possible, these data sets should be uncorrelated, which depending on the problem and application, requires different amounts of data. The choice of *model* for a problem is as important as the available data. For example, models with too many parameters fitted to an insufficiently large training data set will result in models that have "memorised" the training data, with no ability to generalise for before unseen, unknown data. This is the purpose of the testing data, to validate how well a trained model performs for new, uncorrelated data. To measure how well the model performs on new data, an *error function* is used to calculate the model's error. In the case of the previously described model that has "memorised" the training data, the error for the training data is almost zero, but usually significantly large for the testing data, that was not used in training. This type of model is often called an *overfitted* model. To avoid overfitting, a simple strategy is to split the available data into testing and training into equal parts, calculate the errors for both data sets and varying the model parameters until a sufficiently low testing error is achieved, which often is a non-trivial task because of the inherent *bias variance trade-off* of statistics. This trade-off is further discussed in subsection 4.1 Bias-variance tradeoff.

In this section's illustrative example, a polynomial regression model is trained on 30 data points that have been generated by sampling the sine function $sin(x)$ uniformly for $x \in [0, 6]$, which is shown in fig. 4.1(a). Gaussian noise is added to these samples of the sine function, to simulate a more real-life scenario where a sinusoidal behaviour is studied and where the measurements have some uncertainty, caused by for example the process itself being intrinsically stochastic or that there are sources of variability that are themselves unobserved during

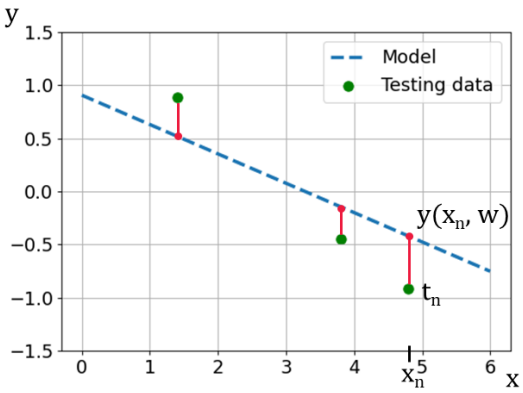
measurements. This data artificially simulates properties of many real data sets, namely that there is an underlying regularity which is desired for the model to learn, but that the individual measurements are affected by random noise. The data therefore consists of $x$ and their corresponding $sin(x)$ values with added noise, which is further equally split into training and testing data sets, shown in fig. 4.1(b). The target for the to-be trained polynomial regression model is of course to reproduce the underlying sinusoidal behaviour when it is applied on data the model has not seen before.




(a) All 30 data points sampled from the sine function.




(b) Sampled data divided equally into training and testing data sets with added Gaussian noise.




(c) Visualisation of the error function, where the red lines show distances between predictions and target values.

Figure 4.1: Visualisations of the complete data set 4.1(a), the divided data set 4.1(b) and the error function used 4.1(c).

### Training and testing

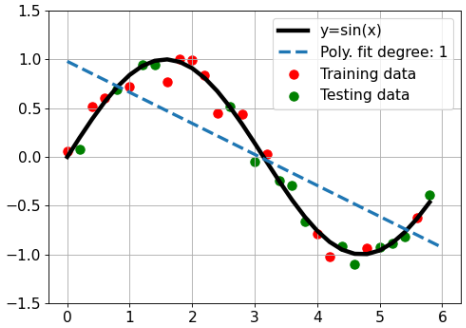
The polynomial regression model that is going to be fit to the training data is of the form

$$y(x, \overline{w}) = w_0 + w_1 x + ... + w_M x^M = \sum_{j=0}^{M} w_j x^j, \tag{4.2}$$

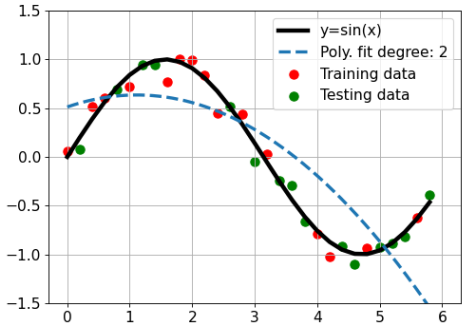
where $M$ is the order of the polynomial, and $\overline{w}$ is a vector of the polynomial coefficients $w_0, ..., w_M$, also called weights, which are determined during the fitting of the model. This will be done by minimising the squared error

$$E(\overline{w}) = \sum_{n=0}^{k} |y(x_n, \overline{w}) - t_n|^2. \tag{4.3}$$

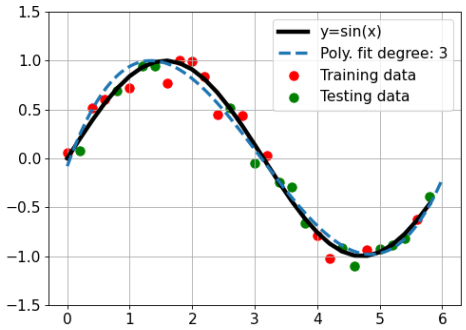
The misfit or distance between the prediction $y(x_n, \overline{w})$ for a set of weights for each training data point $x_n$ and their corresponding target value $t_n$ is visualised by the red line in fig. 4.1(c). The square of these distances are minimised during fitting. Polynomial regression fittings of differing degrees are presented in fig. 4.2.



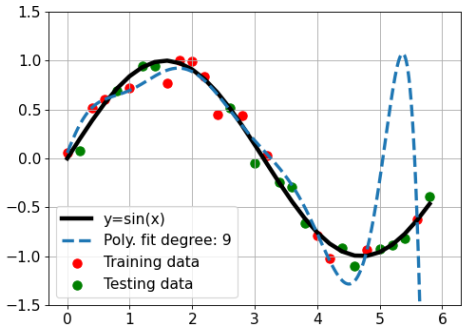

(a) Polynomial regression model, degree 1.




(b) Polynomial regression model, degree 2.




(c) Polynomial regression model, degree 3.




(d) Polynomial regression model, degree 9.

Figure 4.2: Polynomial regression models, fitted for degrees 1, 2, 3 and 9.

From fig. 4.2 it is observed that the models in figs. 4.2(a), 4.2(b) and 4.2(d) do not capture the underlying sinusoidal behaviour of the data, while the model in fig. 4.2(c) seems to reasonably capture the behaviour. The figs. 4.2(a) and 4.2(b) do not have enough parameters to adequately capture the underlying trend of the data, typical cases of *underfitting*. The model in fig. 4.2(d) on the other hand is a typical case of the previously discussed concept of overfitting, the model follows the training data closely, yielding a very low training error, while being unable to generalise for unseen data, yielding a model that does not capture the underlying trend. This illustrates the dangers of only using training error as a metric to choose a suitable model to solve a problem.

The error function eq. (4.3) can also be used to calculate testing errors, which is an equally important metric to determine the quality of a model. Choosing a model among many to solve a problem is fittingly often called *model selection*, and there are many metrics one could consider when selecting the most suitable model. In fig. 4.3 the mean square error (MSE) for the training and testing data are presented. The mean square error is simply the mean over the square error eq. (4.3), i.e. the total error divided by the amount of data points in the set. The MSE metric would allow a fair comparison between the errors of the training and testing data set if they were of different sizes, which however is not the case for this example.

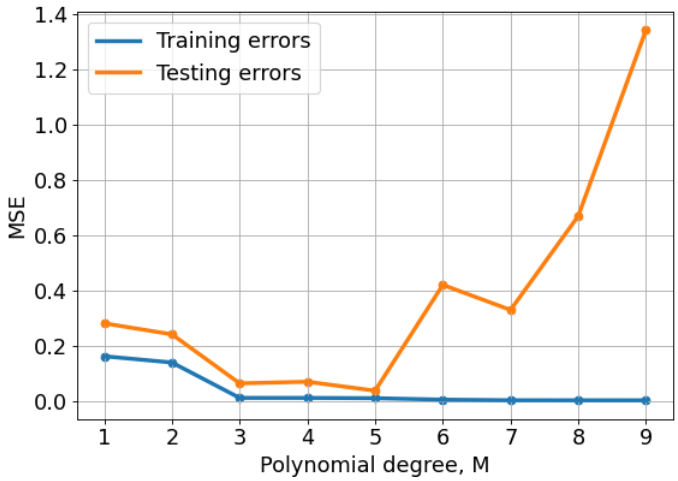



Figure 4.3: The MSEs calculated for the training and testing data for each degree of the fitted polynomial regression models.

It is observed that the training error decreases with increasing polynomial degree, but as discussed earlier, it is seldom a good choice to choose the model with the lowest training error. For example, in fig. 4.3 the model with $M = 9$ has a training error of approximately zero, but the highest testing error of all considered models, yielding an unreasonable model as visualised in fig. 4.2(d). An important concept to consider when selecting model is the before mentioned bias-variance tradeoff.

### Bias-variance tradeoff

The bias-variance tradeoff is the property of a model that the *variance* of the parameter estimates across samples can be reduced by increasing the *bias* in the estimated parameters and vice versa. In the previously presented example of polynomial regression, the parameter estimates are the values of the vector $\overline{w}$. Bias, often formulated as the bias of an *estimator*, in statistics is a measure of the difference between the estimator's expected value and the true value of the parameter of the estimator. In ML, the bias error is the error caused by the ML algorithm's training regiment assuming incorrect behaviour of the training data, missing relevant information and yielding a model that misses important relations between input and output. A high bias is a symptom of underfitting. Variance in statistics is the expectation of the squared deviation of a random variable from it's mean; it is a measure of dispersion i.e. how far an estimator's values are spread from their average. In ML, the variance is an error cause by a model's oversensitivity to small fluctuations in the training data, i.e. the model includes random noise into the predicted behaviour. A high variance is a symptom of overfitting.

To illustrate this tradeoff, consider the subfigures of fig. 4.2; the polynomial model of degree 1 in fig. 4.2(a) exhibits a high bias but a low variance, and the model of degree 9 in fig. 4.2(d) exhibits a low bias but high variance. The model of degree 1 has a low variance since it does not reproduce the small fluctuations caused by random noise in the training data. However, the model has too few parameters to capture the underlying behaviour of the training data, yielding a high bias error. Meanwhile, the model of degree 9 has enough parameters to yield a very low bias, as seen by the small training error in fig. 4.3, but the model captures too much noise, has a high variance, and inadequately models the behaviour of interest. From this bias-variance tradeoff perspective, the model of degree 3 shown in fig. 4.2(c) seems to adequately capture the sinusoidal behaviour of the training data. This model

has a low enough bias to capture the behaviour, and a low enough variance to not get too disturbed by noise and to be able to generalise to new, unseen data points. This is further underlined in fig. 4.3, the model with degree 3 has a low training and testing error. It is common practice to select the model where the training and testing error meet in comparative graphs such as fig. 4.3. Arguably, the model of degree 5 would be a more suitable choice since its training and testing graphs meet closer, but other factors such as model complexity, training time, etc. weighs the model selection towards simpler models.

In conclusion, to solve this illustrative problem the polynomial regression model of degree 3 should be selected, since it is the simplest model which adequately captures the behaviour of the training data, which also is able to generalise to new, previously unseen data.

## 4.2 Workflow for ML in materials science

The typical workflow for ML of atomistic systems is summarised in fig. 4.4, which is redrawn from a figure in [18]. The workflow of course begins with first establishing what atomistic system is of interest. To be able to present the system of interest to an learning model, it needs to first be transformed into a set of descriptive features, as will be discussed in the section below. Using several sets of these descriptors, say for each available time step in the training data, an ML model can be trained to predict a certain physical property of interest.

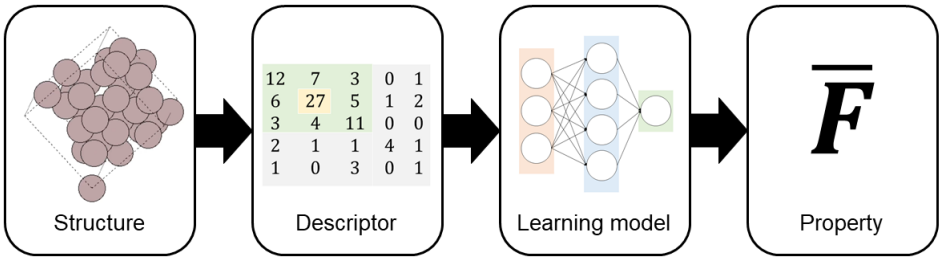



Figure 4.4: Typical workflow for machine learning of atomistic systems.

## 4.3 Descriptors for ML in materials science

The predictive accuracy of ML models of materials properties depends a lot on the chosen descriptor for the material [19]. A descriptor is a feature transformation for atomistic systems. In this thesis, descriptors from the DScribe package are used; as of writing, the package contains implementations of the descriptors Coulomb matrix, sine matrix, Ewald sum matrix, Atom-centered Symmetry Functions (ACSF), Smooth Overlap of Atomistic Positions (SOAP), Many-body Tensor Representation (MBTR) and Local Many-body Tensor Representation (LMBTR) [38, 12, 2, 9, 22, 20, 18]. In this work, the sine matrix descriptor is used for the KRR machine, since that descriptor captures features of interacting atoms in a periodic system with a very low computational cost. This is appropriate to use for the simple, homogeneous systems of Aluminium and Silicon. Another descriptor used in this thesis is the proprietary descriptor used by the MTPs.

Atomistic machine learning establishes a relationship between the atomistic structure of a system and its properties. This "structure-property" relation is illustrated in fig. 4.4. It is analogous to the structure-property relation in quantum mechanics. For a set of nuclear charges and atomic positions of a system, the solution to the Schrödinger equation $H|\Psi >= E|\Psi >$ yields the properties of the system since both the Hamiltonian $H$ and the wave function $\Psi$ only depend on the nuclear charges and atomic positions. Atomistic machine learning bypasses the computationally costly step of solving the Schrödinger equation by training a replacement model. Once trained, the replacement model is typically very fast to evaluate structure-property predictions.

Unlike for the Schrödinger equation, the nuclear charges and atomic positions are not a suitable representation of atomistic systems for ML. For example, they are not rotationally or translationally invariant. If presented with raw atomic positions, the ML model would have to learn rotational and translational invariance for *every* data set. This would significantly increase the required amount of training data. For this reason, the input data has to be transformed into a representation that is suitable for ML. This transformation step is often referred to as *feature engineering* and the selected features are called a *descriptor*.

### Sine matrix descriptor

The sine matrix descriptor's matrix elements are defined as

$$M_{ij}^{\text{sine}} = \begin{cases} 0.5 Z_i^{2.4} & \text{for } i = j \\ \frac{Z_i Z_j}{|\mathbf{B} \cdot \sum_{k=\{x,y,z\}} \hat{\mathbf{e}}_k \sin^2(\pi \mathbf{B}^{-1} \cdot (\mathbf{R}_i - \mathbf{R}_j))|} & \text{for } i \neq j \end{cases} \tag{4.4}$$

where $\mathbf{B}$ is a matrix formed by the lattice vectors, $\hat{\mathbf{e}}_k$ are the Cartesian vectors, and $i$ and $j$ are the indices for the matrix rows and columns respectively. This descriptor is used in the KRR machines, the training of which is explained below.

## 4.4 Kernel Ridge Regression

Ridge regression is a regression method, which when kernalized is the basis for *kernel ridge regression*. The underlying idea is to create a KRR model for some property, $y$, for some system, $\tilde{X}$. This property $y$ could for example be the atomization energy of a molecule. This kind of model is described by

$$y(\tilde{X}) = \sum_i \alpha_i K(\tilde{X}, X_i). \tag{4.5}$$

In eq. (4.5) $\alpha$ are the regression coefficients and $K$ is some kernel function, for example a Gaussian kernel function with Frobenius norm as given by (4.6).

$$K_{ij} = K(X_i, X_j) = \exp\left(-\frac{||X_i - X_j||_2^2}{2\sigma^2}\right). \tag{4.6}$$

When describing the system $\tilde{X}$ different types of atomic descriptors can be used, for example sine or Coulomb matrix representations. As mentioned earlier, in this thesis the sine matrix descriptor is used.

## 4.5 Moment Tensor Potentials

Moment Tensor Potentials are a class of systematically, improvable interatomic potentials developed at Skoltech by Shapeev et al., implemented in the MLIP (and later, MLIP2) package [39, 32]. MTPs are able to actively select configurations and parametrize the potential while running. Demonstrably MTPs accurately reproduce energies, forces and stresses calculated ab-initio [33]. The outline of this section is heavily inspired by Novoselov et al. in [33].

As with most interatomic potentials, it is assumed that the total interaction energy of a configuration can be represented as a sum of atomic contributions. The contribution from a single atom $i$ can be defined as $V(\mathbf{r}_i)$, where $V$ is the interatomic potential and $\mathbf{r}_i = (r_{i,1}, ..., r_{i,n})$ is a collection of vectors pointing from the atom $i$ to its neighbours inside the potential cut-off. MTP then postulates linear representations of each of the atomic contributions $V(\mathbf{r}_i)$

$$V(\mathbf{r}_i) = \sum_{j=1}^{m} \theta_j B_j(\mathbf{r}_i), \tag{4.7}$$

where $\theta_j$ are adjustable parameters, $B_j$ are pre-defined basis functions and $m$ is the number of functions in the basis. The total energy of the system is then given by the sum of $V(\mathbf{r}_i)$ over all atoms

$$E(x) = \sum_{i=1}^{N} \sum_{j=1}^{m} \theta_j B_j(\mathbf{r}_i), \tag{4.8}$$

where $N$ is the number of atoms in the configuration $x$. The force acting on $j$-th atom $f_j(x)$ is determined as a derivative of $E(x)$ with respect to the atom position $x_j$

$$f_j(x) = -\nabla E(x) \tag{4.9}$$

Similarly, the virial stresses can be calculated as derivatives of $E(x)$ with respect to the lattice vectors $L$

$$\sigma(x) = \frac{1}{|\det(L)|}(\nabla_L E(x))L^\top \tag{4.10}$$

As observed from eqs. (4.7), (4.8), (4.9) and (4.10), the energy, forces and stresses for a given configuration are determined by the set of basis functions $B_j$, and the values of the adjustable parameters $\theta_j$. These parameters can be found through fitting an unfitted MTP to the results of DFT calculations. Suppose that DFT calculations were run for some amount of time steps, for some collection of configurations $X_{TS}$, which is denoted as the training set. For each configuration $x_i$ in the collection $X_{TS}$, the "exact" energy ($E^{DFT}(x_i)$), per-atom forces ($f_j^{DFT}(x_i)$) and the components of the stress tensor ($\sigma_j^{DFT}(x_i)$) are known. Therefore, the MTP energy error for configuration $x_i$ is given by

$$\Delta E(x_i) = |E(x_i) - E^{DFT}(x_i)| \tag{4.11}$$

The errors of the forces and stresses are defined in a way similar to (4.11). The values of $\theta_j$ can be found through minimisation of the functional

$$\sum_{x_i \in X_{TS}} \left[ C_E^2 \Delta E(x_i)^2 + C_f^2 \sum_{j=1}^{N_i} \Delta f_j(x_i)^2 + C_s^2 \Delta \sigma(x_i)^2 \right] \tag{4.12}$$

As can be noted in eq. (4.12), the energy, forces and stress are weighted by the weight coefficients $C_E$, $C_f$ and $C_s$ respectively. These are the weights being changed during the fitting routine in order to minimise the error. The so called fitting weights also reflect the relative importance of the properties during fitting.

MTPs are able to select configurations for the training set based on several features: per-atom energy ($V(\mathbf{r}_i)$), energy of a configuration ($E(x_k)$), forces ($f_i(x_k)$) and stresses ($\sigma_j(x_k)$). To each of these features there are a corresponding selection strategy, namely selection by neighbours, energies, forces or stresses respectively. The strategies could be used simultaneously, similarly how different quantities are used in the fitting routine described by eq. (4.12). The selection process is characterised by the set of weights, each strategy has an associated weight and the weight determines its relative importance in the process. For a given set of weights, the number of selected configurations is controlled by a threshold value, which determines the allowed degree of extrapolation. If the threshold is exceeded, the configuration is added to the optimised training set, which of course implies that a decrease of the threshold leads to an increase in the number of selected configurations.

### .cfg files for MTP

The MTPs handle proprietary .cfg files that are text files containing configurations, with computed energies, forces, stresses etc. An example of such a .cfg file is given in figure 4.5. For a thorough explanation of these files, see the MLIP2 documentation[1]. Access to the MLIP2 GitLab repository has to be requested from Skoltech[2].

[1] https://gitlab.com/ashapeev/mlip-2/-/tree/master/doc
[2] https://mlip.skoltech.ru/download/

```
BEGIN_CFG
 Size
  4
 Supercell
  2.86      0.00      0.00
  0.00      5.71      0.00
  0.00      0.00      4.05
 AtomData:  id type cartes_x cartes_y cartes_z     fx     fy     fz
             1    0     0.00    -0.13     0.00   0.00   0.75   0.00
             2    0     1.43     1.45     2.06   0.00  -0.13   0.00
             3    1     0.00     2.87     0.00   0.00  -0.41   0.00
             4    1     1.43     4.26     2.06   0.00  -0.20   0.00
 Energy
  -16.023765555539
 PlusStress:  xx     yy     zz     yz     xz     xy
           -0.29   0.23  -0.09   0.00   0.00   0.00
Feature   EFS_by   vasp
Feature   from     database:p.cfg
Feature   mindist  2.70
END_CFG
```

Figure 4.5: An example of an MTP proprietary .cfg file for a configuration.

# 5 Method

> What we observe is not nature itself, but nature exposed to our method of questioning.
>
> Werner Heisenberg

This chapter outlines the methodology of this project work, which is mainly done in the programming languages Python version 3.6.4 and bash. Most of this work is available in a corresponding GitHub repository[1]. All the dependencies in the virtual environment used in this work are shown in the file `requirements.txt` in the repository[2].

Worth to note is that the majority of the work in this thesis project is spent developing the software to conduct the MD simulations described below. Throughout the project, a simulation software is developed which allows for significant control over workflow, data storage, MD parameters such as thermostats, potentials etc. There are of course simulation software available as of writing such as LAMMPS that could have been used to simulate the physical properties of interest, but part of the aim is the development itself of the highly modifiable simulation software that is used in this thesis [36].

Different ways to implement interatomic potentials are investigated. In particular KRR and MTP based potentials are implemented and tested. These potentials are then used in MD to calculate physical properties of bulk Aluminium and Silicon, and compared to properties calculated from DFT simulations. Historically, throughout the work, MTPs became an increasingly larger focus in the project and makes up a majority of the method and results.

## 5.1 Data

The main data set used in this thesis are DFT simulations of Aluminium and Silicon simulated by Davide Gambino and Johan Klarbring using VASP [15]. The simulations were done in the NVT ensamble, using the Nosé-Hoover thermostat, at a temperature of 300 Kelvin. The simulations were done for 10000 time steps with a time length of 1 fs, i.e. for in total 10000 fs. The simulated system of Aluminium consisted of 32 atoms; the simulated system of Silicon consisted of 8 atoms. The data is divided into a training set and a data set in various ways to investigate how this division of data would affect performance of the ML models. The data is divided as described by figs. 5.1, 5.2, 5.3 and 5.4 for different situations as described further below.

[1] `https://github.com/obsqyr/master-thesis`
[2] `https://github.com/obsqyr/master-thesis/blob/main/requirements.txt`

The reasoning for why Aluminium and Silicon data sets are used is because DFT simulates the bulk properties of these materials efficiently and with a high accuracy. Therefore, if the ML potential simulations would reasonably replicate the DFT simulations it would indicate that the ML potentials are a viable way to accelerate the DFT simulations. Furthermore, it would indicate that the ML potential simulations could most likely be extended for other materials as well.

The first division of data uses all available data in a 90/10 ratio for training data to test data, as shown in fig. 5.1.

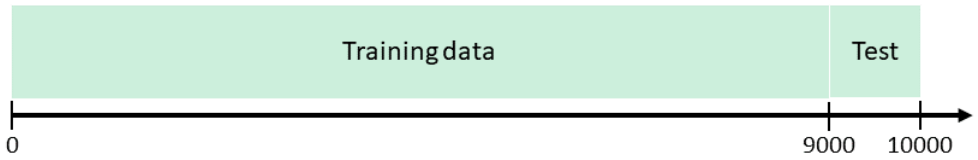



Figure 5.1: The initial 9000 time steps are used as training data and the last 1000 time steps are used for test data.

Using all available data would of course be the best course of action if all of the data is reliable, but by looking at when both the Aluminium and Silicon DFT simulations reach equilibrium, it is observed that this occurs after around 2000 time steps. Reasonably, training on these time steps before equilibrium has been reached should produce potentials which reproduce nonphysical behaviour. This motivates the data division in fig. 5.2.

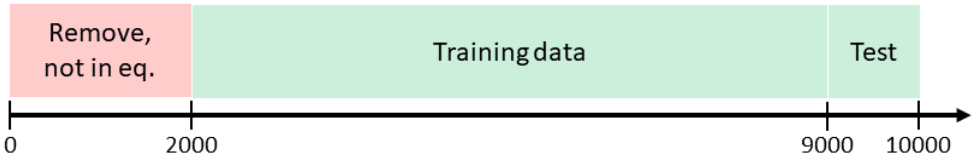



Figure 5.2: The time steps after the 2000th time step until the 9000th are used as training data and the last 1000 time steps as test data.

Another argument which is considered is the correlation of time steps. To be certain that the training and test data sets are uncorrelated, the data division in figs. 5.3 and 5.4 introduce a buffer of 1000 time steps between the training and test data sets. The size of the buffer is chosen as half of the equilibrium time and is assumed to be a sufficiently long time to uncorrelate the data sets.

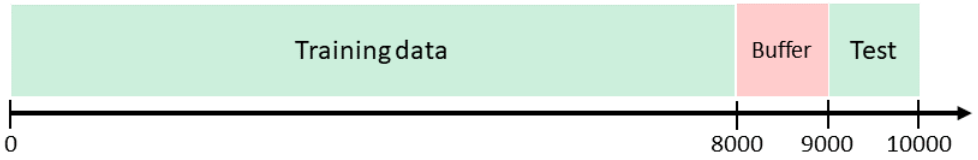



Figure 5.3: The initial 8000 time steps are used as training data and the last 1000 time steps are used as test data.

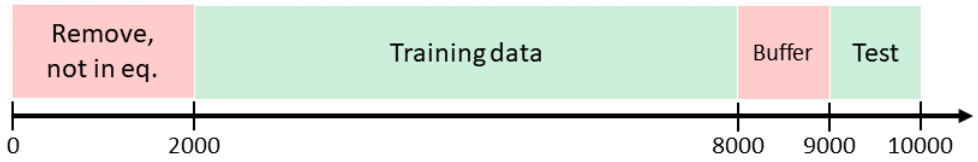



Figure 5.4: The time steps after the 2000th time step until the 8000th are used as training data and the last 1000 time steps as test data.

## 5.2 Training

This section outlines how the ML models are trained. For the sake of being potentials usable in MD, the ML models are required to be able to predict the potential energy for the whole system and forces for each atom for each atomic system descriptor.

### Kernel Ridge Regression

Different ways of implementing KRR machines are tested. Initially, training data is generated from the DFT data by reading it and transforming each atomistic system at each time step to a descriptor. The sine matrix descriptor as implemented in the Python package DScribe is used for this [18]. The resulting transformed data set is then divided into a training set and a test set, which are denoted $X_{tr}$ and $X_{te}$ respectively from now. The first 9000 time steps are assigned to $X_{tr}$ and the last 1000 to $X_{te}$, as per the data division in fig. 5.1.

The KRR models are constructed using the quantum machine learning (QML) Python package [7]. When training the KRR machines, first kernels are generated as described by eq. (4.6) for some subset of $X_{tr}$. To be able to do the regression described by eq. (4.5), $\boldsymbol{\alpha}$ has to be calculated. This can be done through kernel matrix inversion and multiplication with target labels $\mathbf{y}$,

$$\boldsymbol{\alpha} = (\mathbf{K} + \lambda \mathbf{I})^{-1}\mathbf{y}. \tag{5.1}$$

Eq. (5.1) was solved in this work by Cholesky-decomposition, using a function implemented in QML. With the obtained $\boldsymbol{\alpha}$ it is then possible to do predictions as described by eq. (4.5). What the target label $\mathbf{y}$ is, depends on which machine is being trained; the target labels are either the potential energy for the whole system or the three force components for each atom.

The KRR machine for predicting the potential energy of the atomistic system is quite simple to train. Only one fitting as described above is needed to obtain a set of $\boldsymbol{\alpha}$ that can be used to predict the total potential energy. In the case of predicting forces the situation is different. Three machines are fitted for each atom in the system; one for each force component. So for each atomistic system, three machines per atom are fitted on the descriptor for the $\mathbf{y}$ targets of the force in the Cartesian x, y and z directions. The KRR machines are trained on 1, 10, 100, 1000 and 10000 time steps and their testing errors are shown as part of the results, see section 6.1.

### Moment Tensor Potential

The MTPs are trained as outlined by section 4.5. Practically, the training is done with a training script using functions in the MLIP2 package. Firstly, the DFT data has to be converted to the appropriate .cfg format used by MLIP2, and then a similar division into $X_{tr}$ and $X_{te}$ as with the KRR machines is done, i.e. the 1000 last configurations are reserved for the testing data. Different from the KRR training however, training data sets $X_{tr}$ starting from both 0 and 2000 time steps are constructed and used, as per the data divisions in figs. 5.1 and 5.2. MTPs with different amounts of parameters are investigated, namely the 06 and 10 MTPs, which have 21 and 40 parameters respectively in the case of monoatomic materials. These MTPs are

fitted to training data sets $X_{tr}$ containing 1, 10, 100, 1000 and 10000 (or 8000, for the case of the training data that includes time steps after the initial 2000) time steps.

During development, initially when fitting the MTPs, the default value for the maximum iterations, which is 100 iterations, was used for the Broyden-Fletcher-Goldfarb-Shanno (BFGS) optimiser. This default value was not sufficient for the data sets used for finding an optimum. The amount of maximum iterations was later increased to 1000, which improved the quality of the yielded potentials as the optimiser often required more than the default 100 iterations to converge with the default convergence tolerance parameter for the error, which is 0.001 and which was used in this work. A piece of advice for other users is to log the output of the the BFGS optimisation during training and observe if and when the optimisation problem converges with the selected maximum iterations and convergence error tolerance.

## 5.3 Testing

This section describes the testing of the ML models. The metric used to calculate the error is the mean absolute error (MAE) metric. In the case of the KRR machines, predictions are made using the obtained $\boldsymbol{\alpha}$ values from training and compared to the target labels $\mathbf{y}$ in $\mathbf{X}_{te}$, calculating the MAEs in the comparison. In the case of the MTPs, a testing script using functions from the MLIP2 package is used to calculate the MAE errors. These errors and how they are affected by the amount of time steps used in training are shown in figs. 6.1 and 6.2 in section 6.1.

## 5.4 MD simulations

The MD simulations are implemented using the atomic simulation environment (ASE), which are a set of tools and Python modules for running atomistic simulations and more [30]. ASE is used in conjunction with the package ASAP3 for a huge performance increase. In the MD simulations, the same sizes for the unit cells as in the DFT simulations are used, i.e. 32 and 8 atoms for Aluminium and Silicon respectively. The simulations are done in the NVT ensamble, the same ensamble which was used in the DFT simulations, and Langevin dynamics are used. In Langevin dynamics, each atom is coupled to a heat bath through a fluctuating force and friction term; essentially a friction term and a fluctuating force are added to Newton's second law, which is then integrated numerically. The friction coefficient is typically chosen between 0.0001 and 0.01. In both the Aluminium and Silicon simulations a friction coefficient of 0.01 is used for the Langevin thermostat.

All simulations are run at the 110-node HPC cluster Sigma at the National Supercomputer Centre at Linköping University.

### ML calculators

In the context of MD, a calculator as implemented in ASE is a class that in essence calculates the potential energy and forces for the atoms in an atomistic system only knowing the positions of each atom and the composition of the material. In this project calculators using KRR and MTP to predict the potential energy and forces are created. Each calculator can be initialised with a specified amount of time steps, if there's a corresponding fitted potential. Also, for the MTP calculators the order number of the MTP can be chosen, i.e. either 06 or 10, and whether the fitted potential was trained on time steps from 0 or 2000. For both types of calculators, it is possible to choose an offset, which means for how long the simulation will run before physical properties are calculated and logged.

For the KRR calculator the current ASE atoms object in the MD run is transformed into its corresponding sine matrix representation, which the trained potential used by calculator uses to predict potential energy and forces. These predictions are then used in the MD scheme

to further the simulation by one time step. The MTP calculator works similarly, with the difference being that the atoms object is transformed into the MTP appropriate .cfg format, which is then used to produce similar predictions of potential energy and forces.

### Physical properties calculations

The physical properties that are calculated during the MD simulations are the specific heat capacity, the total energy and the mean square displacement (MSD). Time averages and instantaneous values for these properties are calculated and plotted, which are presented in chapters 7 and 8. The hyperparameters used in the simulations that affect the calculation of physical properties are presented in table 5.1.

Table 5.1: Hyperparameters used in MD simulation and their descriptions.

| Hyperparameter | Description |
|---|---|
| eq | The potential is trained on time steps after *eq* amount of time steps. |
| offset | The simulation is allowed to run for *offset* amount of time steps before properties are logged. |

The MSD is calculated as

$$\text{MSD} \equiv < |\overline{x}(t) - \overline{x}_0|^2 > = \frac{1}{N} \sum_{i=1}^{N} |\overline{x}^{(i)}(t) - \overline{x}^{(i)}(0)|^2, \tag{5.2}$$

where $N$ is the number of particles to be averaged, the vector $\overline{x}_0^{(i)} = \overline{x}^{(i)}(0)$ is the reference position of the $i$-th atom, and the vector $\overline{x}^{(i)}(t)$ is the position of the $i$-th particle at time $t$ [14]. The specific heat capacity in the NVT ensamble is calculated as

$$c_v = \frac{\langle E^2 \rangle - \langle E \rangle^2}{z k_B T^2}, \tag{5.3}$$

where $E$ is the total energy, $z$ is the total mass of the system, $k_B$ is the Boltzmann constant, and $T$ is the instantaneous temperature [14]. If $E$ is in the unit J, $z$ in the unit kg, $k_B$ in the unit J/K, and $T$ in the unit K, eq. (5.3) of course yields the specific heat capacity in its SI unit of J$\cdot$ kg$^{-1}\cdot$ K$^{-1}$. The total energy is calculated by ASE during the simulation and given in the unit of $eV$ per atom. These bulk properties are interesting to look at since these bulk properties for Aluminium and Silicon are well simulated by DFT, and if they could be reproduced using ML potentials it would indicative of their utility.

## 5.5 Runtime

In table 5.2 the different simulations' run times are presented. The simulations ran on one core at the Sigma supercomputer and were simulated for 10000 time steps.

Table 5.2: CPU minutes for each studied simulation, simulated for 10000 time steps on one core on the Sigma supercomputer. Both MTPs presented are trained on 100 time steps starting from the first time step. For clarification, 'MTP training time' is the time the MTP requires to be trained, 'DFT generation time' is the time it takes to produce the DFT training data, and 'Simulation time' is the time it takes to simulate 10000 time steps.

| | MTP training time | DFT generation time | Simulation time* | Total time |
|---|---|---|---|---|
| **Al DFT** | - | - | - | 29.2 |
| Al MTP 06, 100 | 1 | 0.3 | 2.2 | 3.5 |
| **Si DFT** | - | - | - | 15.7 |
| Si MTP 06, 100 | 0.2 | 0.16 | 0.9 | 1.32 |

*Worth to note is that the implementation of the MTP simulations is prototypical at best, and the time a simulation would take with a more robust implementation is trivial.

# 6 Potentials training

> The most exciting phrase to hear in science, the one that heralds the most discoveries, is not "Eureka!" but "That's funny..."
>
> Isaac Asimov

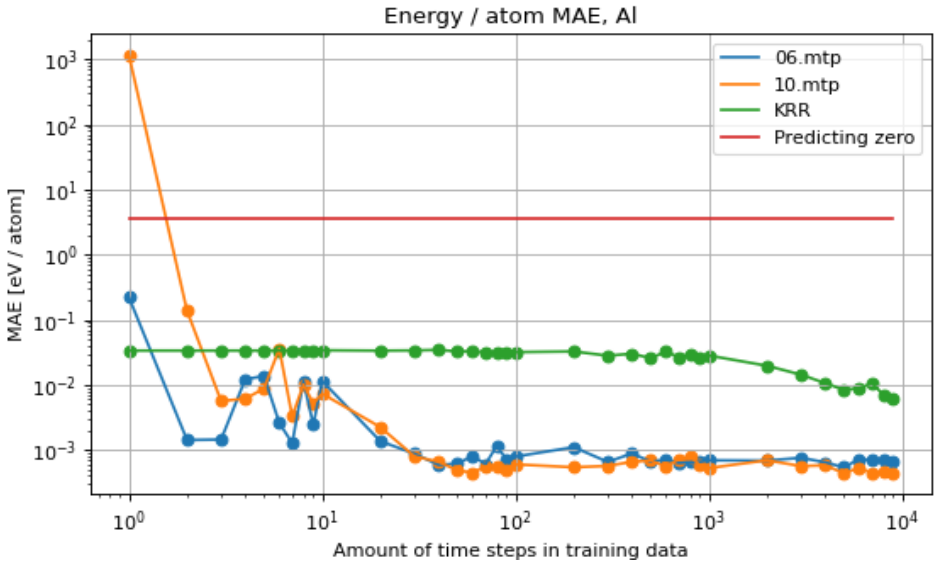
The results of this thesis project are presented in this chapter, chapter 7 and chapter 8. In the section 6.1 in this chapter the testing errors of the implemented machines are presented, and in a subsection the training vs. testing errors for the MTPs are presented.

## 6.1 Testing errors

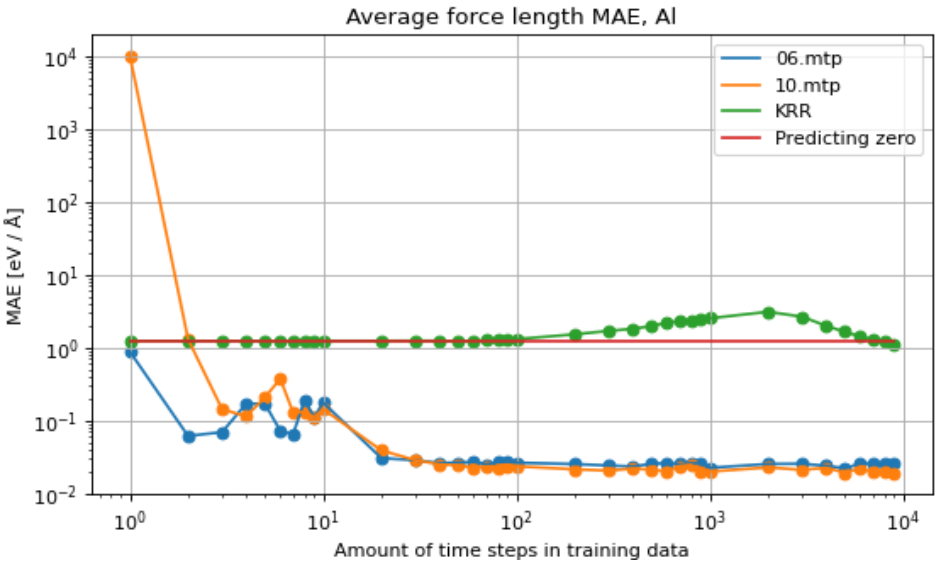
In figs. 6.1 and 6.2 the data is divided as fig. 5.1, i.e. using the initial 9000 time steps for training and the last 1000 time steps for testing. In these figures, both axes are on a logarithmic scale, where the vertical axis is the mean absolute error for the properties energy per atom in figs. 6.1(a) and 6.2(a) and average force length in figs. 6.1(b) and 6.2(b), and the horizontal axis is the amount of training data used. The force lengths are simply the magnitudes of the force vectors. The testing errors for all studied models are plotted in these figures, i.e. the 06 and 10 MTPs and the KRR model.

As is observed in figs. 6.1(a) and 6.1(b), after being trained on all available time steps, the 06 and 10 MTPs converge towards energy per atom MAEs of 0.0007, 0.0004 eV respectively, and average force length MAEs of 0.03 and 0.02 eV / Å respectively. The KRR potential machine does not converge during training, it is observed in fig. 6.1(a) that the error curve is on a downwards slope when the machine is trained on all available data. It would require more data to see how this KRR implementation would converge. Worth to note is that all models exhibit fairly low energy per atom errors after only being trained for a few time steps, and that all models have lower errors than a baseline machine predicting zero energy, which is shown as the red line in fig. 6.1(a). The MTPs lower their energy per atom error with with approx. two magnitudes after only having been trained on around 30 time steps. Training the MTPs for more data does not significantly increase performance.

Similarly, as in the case for the KRR potentials machine, the KRR forces machine does not converge using the available training data as is observed in fig. 6.1(b). For the first 100 time steps of training, the KRR forces machine performs equally to the baseline of predicting zero. After the initial 200 time steps the KRR machine yields errors that are larger than baseline. When nearly trained on all data, the KRR machine starts yielding errors lower than baseline, and similarly to the KRR potentials machine, a downwards slope is observed and more data would be required to see when this implementation converges. The MTPs predicting forces similarly do not require much data to yield errors of almost two magnitudes lower than the




(a) Energy per atom MAE for ML models trained on Aluminium data.




(b) Average force length MAE for ML models trained on Aluminium data.

Figure 6.1: The MAE testing errors for the properties energy per atom and average force length for the different ML models trained for the Aluminium data.

error after being trained on only 1 time step. After being trained on 30 time steps the MTPs do not significantly improve with more training data. It is also observed that both MTPs converge to very similar errors at around the same amount of training data.

As is observed in figs. 6.2(a) and 6.2(b), after being trained on all available time steps, the 06 MTP and the 10 MTP trained on Silicon data reach energy per atom MAEs of 0.0003 and 0.0001 eV respectively, and average force length MAEs of 0.01 and 0.005 eV / Å respectively. The KRR potential machine does not converge during training, it is observed in fig. 6.2(a) that the error curve is on a downward slope when the machine is trained on all available data. This echoes the KRR potential machines for Aluminium. Once again, it is worth to note

that all models exhibit fairly low energy per atom errors after only being trained for a few time steps compared to baseline. Worth to note is that compared to the Aluminium training, more data is required before the MTPs converge, which occurs after being trained for 100 to 200 time steps. The slope towards convergence is not as steep, but after 100 to 200 time steps the MTPs' error do not significantly change.

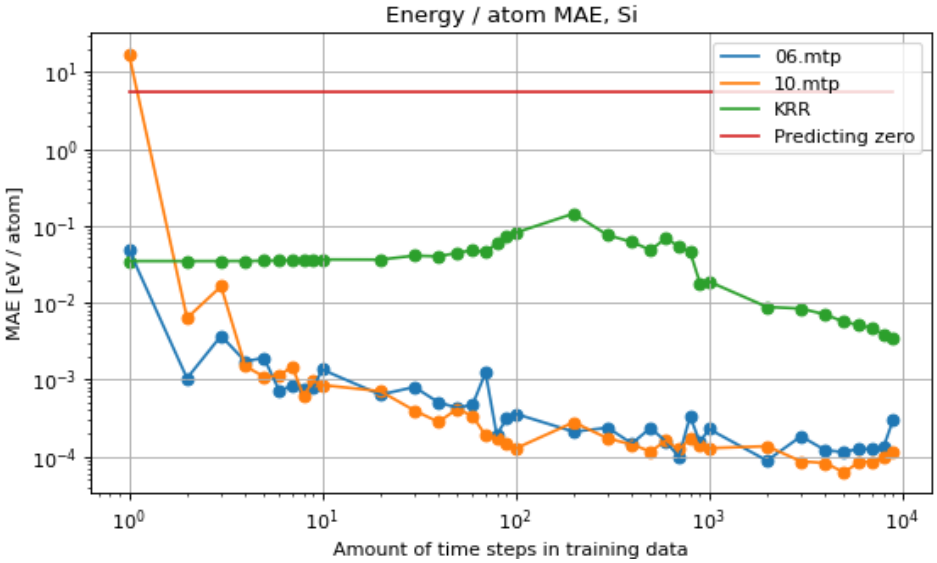
The KRR forces machine for Silicon, different from the case of Aluminium, does seem to converge after being trained on 2000 to 3000 time steps to a value in the same order of magnitude as baseline. But similarly to the Aluminium KRR forces machines, after being trained on around 50 time steps the Silicon forces machine yields errors larger than baseline until it is trained on 1000 time steps. The MTPs predicting forces similarly require to be trained on 100 to 200 time steps before they reach errors of approx. two magnitudes lower than the initial errors. It is once again observed that the MTPs converge at this point towards similar values, and do not significantly improve being trained on more time steps.

Worth to note for both MTPs, for both Aluminium and Silicon, and for both the properties energy per atom and forces no significant difference in errors in the limit of the training data is observed.

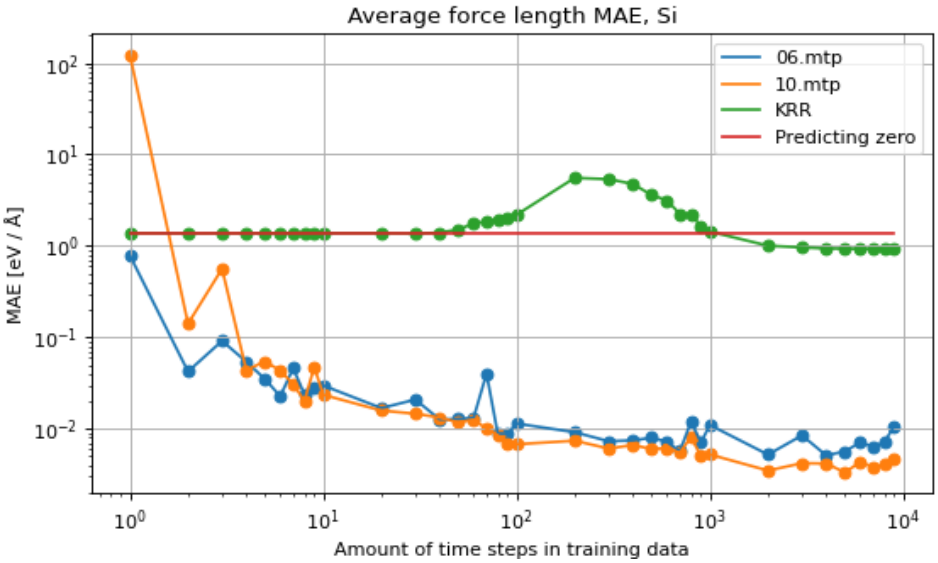
### Training vs. testing errors for the MTPs

In figs. 6.3 and 6.5 the data is divided as fig. 5.3, i.e. using the initial 8000 time steps for training and the last 1000 time steps for testing. In these figures, both axes are on a logarithmic scale, where the vertical axis is the mean absolute error for the properties energy per atom and average force length for the 06 MTPs for Aluminium and Silicon in figs. 6.3(a) and 6.5(a) respectively in figs. 6.4(a) and 6.6(a) and similarly for the 10 MTPs for Aluminium and Silicon in figs. 6.3(b) and 6.5(b) respectively in figs. 6.4(b) and 6.6(b). The horizontal axis is the amount of training data used.

As is observed from figs. 6.3 and 6.4 the training and testing errors meet at 1000 time steps for both properties energy per atom and average force length, and both the 06 and 10 MTPs for Aluminium. In the limit of 8000 time steps, the energy per atom training and validation errors for MTP 06 and 10 reach 0.0005 and 0.0006 eV, and 0.0004 and 0.0004 eV respectively. In the limit of 8000 time steps, the average force length training and validation errors for MTP 06 and 10 reach 0.02 and 0.02 eV / Å, and 0.02 and 0.02 eV / Å respectively.

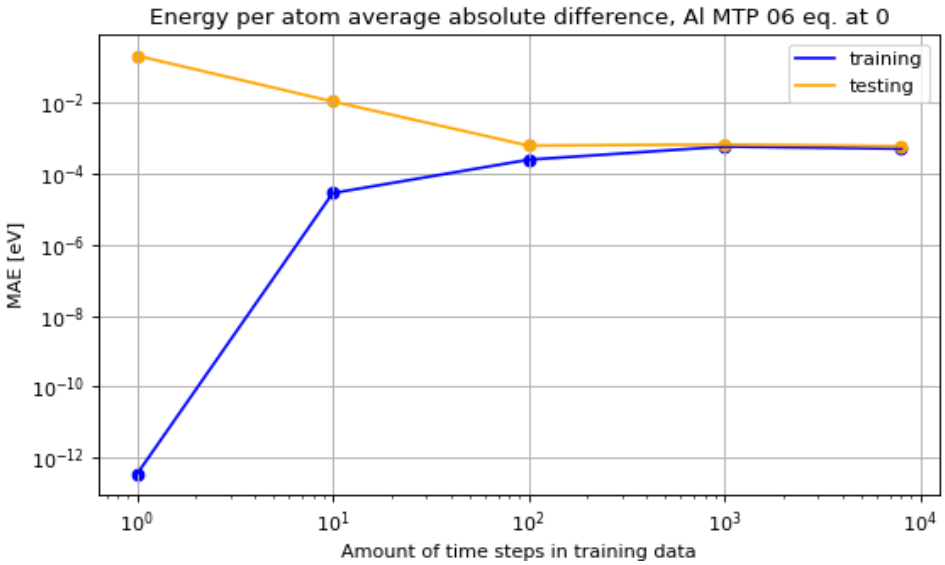



(a) Energy per atom MAE.

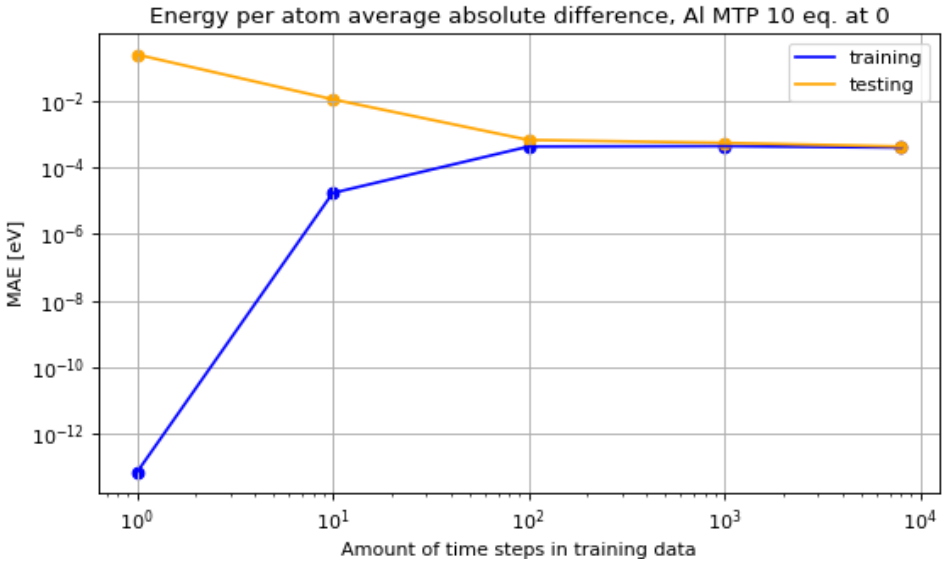



(b) Average force length MAE.

Figure 6.2: The testing errors for the different ML models trained for the Silicon data.

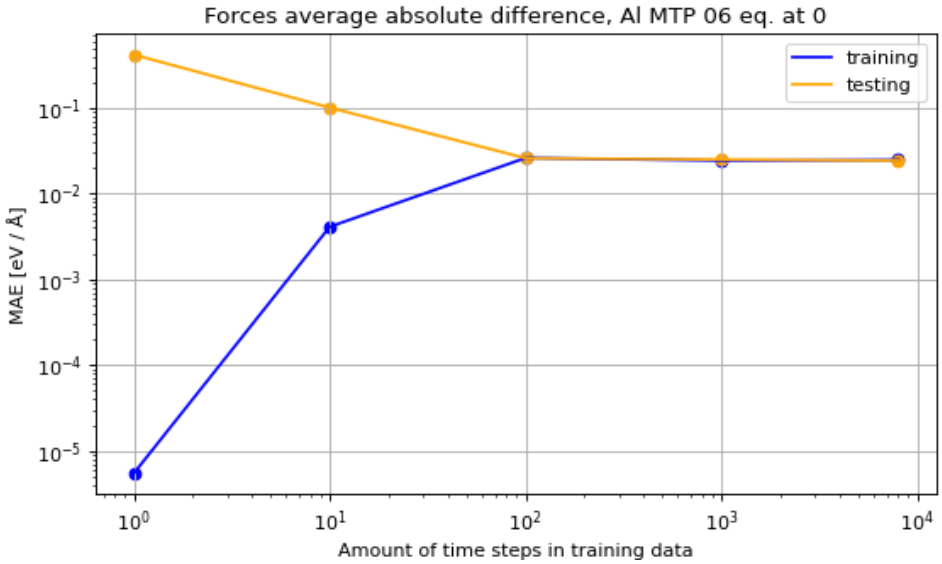



(a) Energy per atom average absolute difference for Aluminium MTP 06 trained assuming eq. at 0.

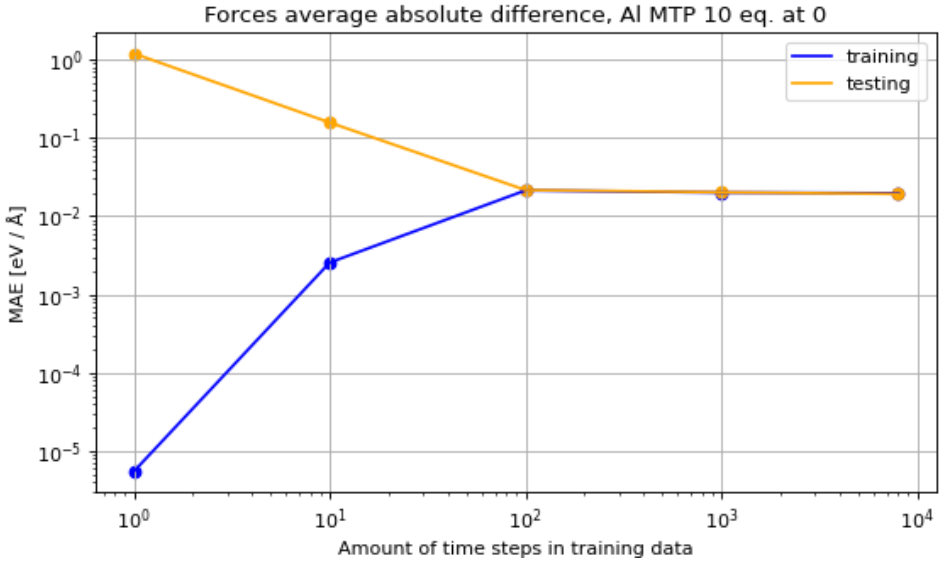



(b) Energy per atom average absolute difference for Aluminium MTP 10 trained assuming eq. at 0.

Figure 6.3: Training vs. testing energy per atom average absolute difference errors for Aluminium MTPs 06 and 10 trained on data starting from the first time step.



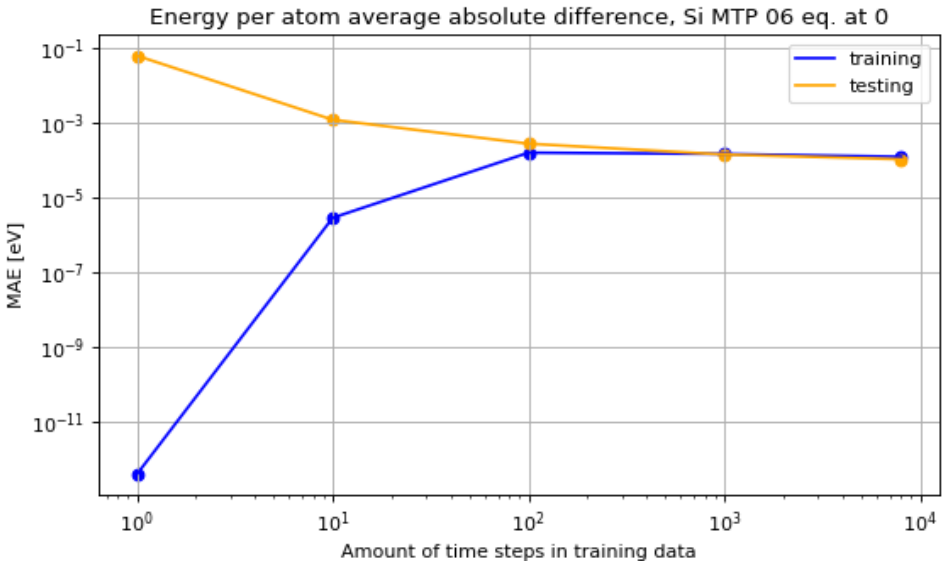

(a) Forces average absolute difference for Aluminium MTP 06 trained assuming eq. at 0.



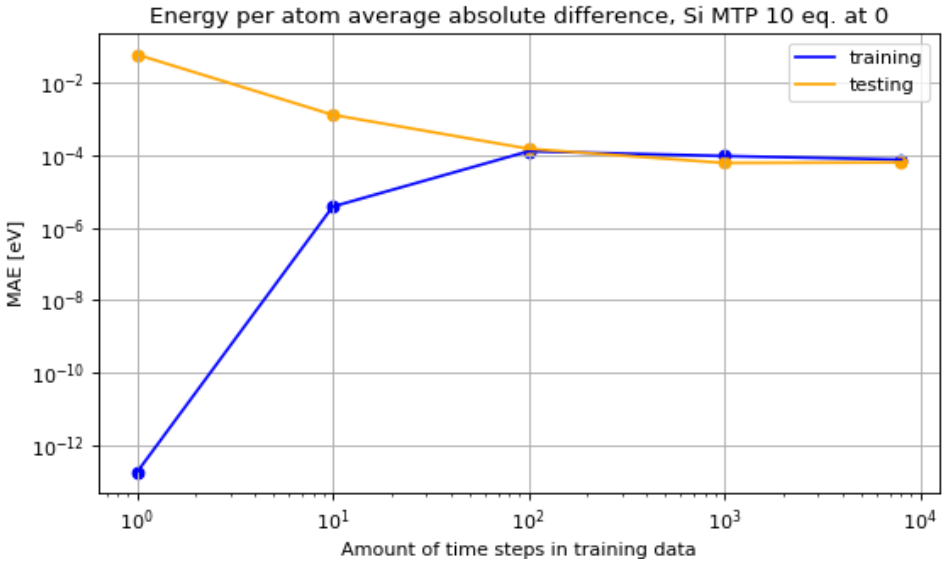

(b) Forces average absolute difference for Aluminium MTP 10 trained assuming eq. at 0.

Figure 6.4: Training vs. testing forces average absolute difference errors for Aluminium MTPs 06 and 10 trained on data starting from the first time step.

As is observed from figs. 6.5 and 6.6 the training and testing errors meet at 1000 time steps for both properties energy per atom and forces, and both the 06 and 10 MTPs for Silicon. In the limit of 8000 time steps, the energy per atom training and validation errors for MTP 06 and 10 reach 0.0001 and 0.0001 eV, and 0.00008 and 0.00006 eV respectively. In the limit of 8000 time steps, the forces training and validation errors for MTP 06 and 10 reach 0.004 and 0.004 eV / Å, and 0.003 and 0.003 eV / Å respectively.

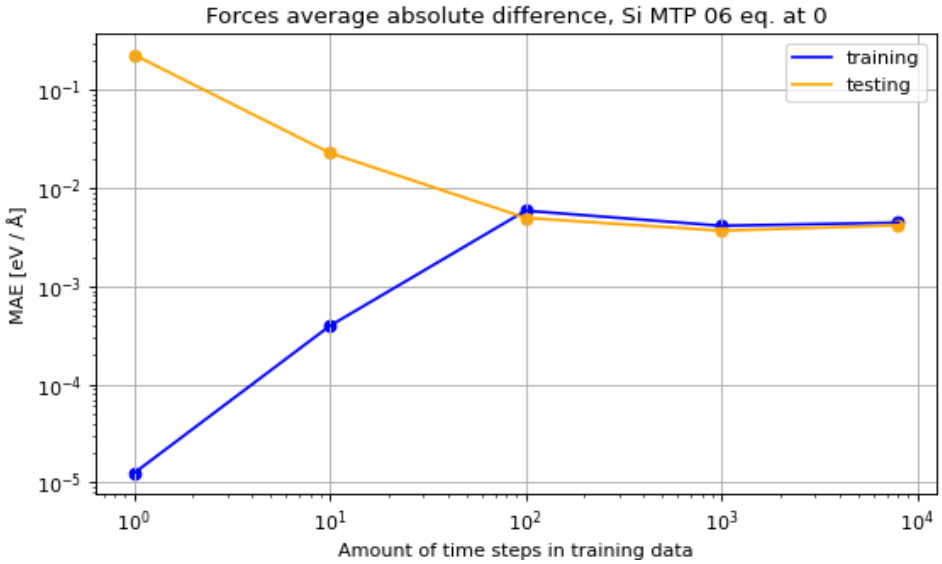



(a) Energy per atom average absolute difference for Silicon MTP 06 trained assuming eq. at 0.

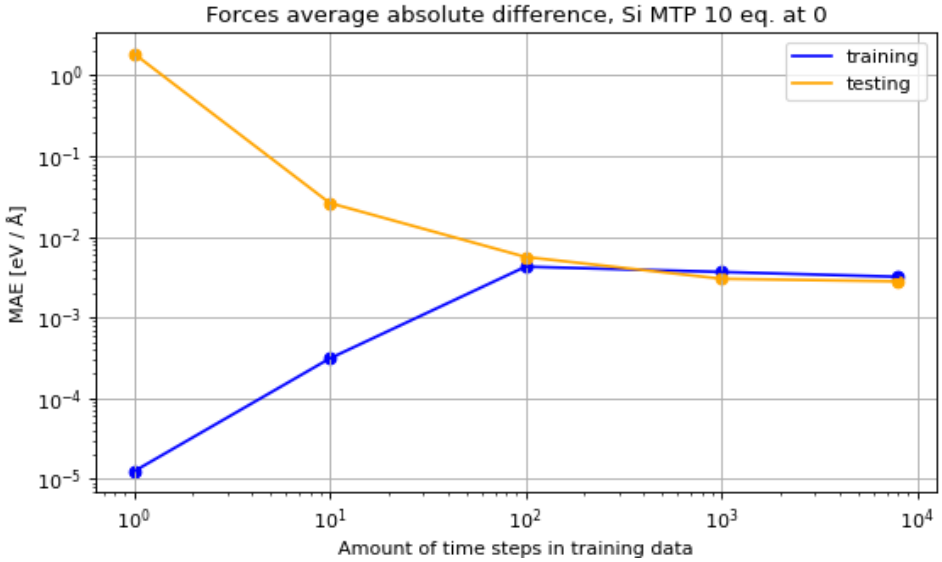



(b) Energy per atom average absolute difference for Aluminium MTP 10 trained assuming eq. at 0.

Figure 6.5: Training vs. testing energy per atom average absolute difference errors for Silicon MTPs 06 and 10 trained on data starting from the first time step.




(a) Forces average absolute difference for Silicon MTP 06 trained assuming eq. at 0.




(b) Forces average absolute difference for Aluminium MTP 10 trained assuming eq. at 0.

Figure 6.6: Training vs. testing forces average absolute difference errors for Silicon MTPs 06 and 10 trained on data starting from the first time step.

Figs. 6.3, 6.4, 6.5 and 6.6 echo the same observations as in the testing errors section that both MTP 06 and 10 require the same amount of training data to learn the underlying behaviour of both Aluminium and Silicon and perform equivalently.

# 7 Aluminium simulations

In the section 7.1 in this chapter the calculated physical properties' instantaneous and time averaged values are presented for Aluminium. In section 7.2 the variance of the MTPs during simulation are illustrated.

## 7.1 Calculated physical properties

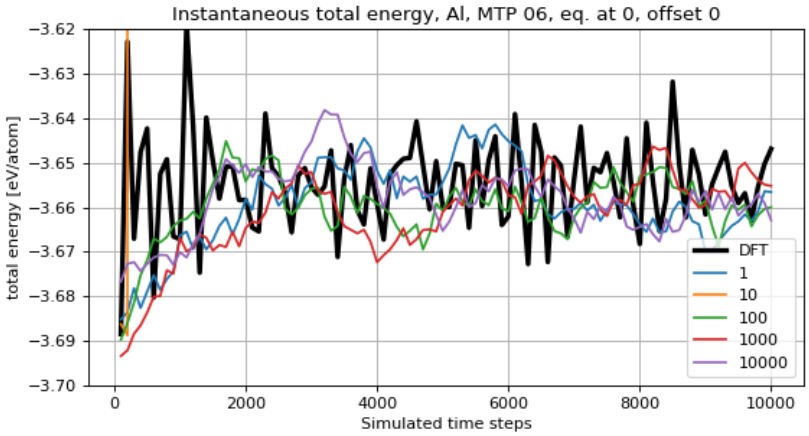
The instantaneous and time averaged values of the calculated physical properties total energy, mean squared displacement and specific heat are presented in this section. The instantaneous values are plotted to show if the properties oscillate reasonably and the time averaged values are plotted to show whether the properties converge to the equivalent DFT value. All these properties are calculated from MD simulations using the Langevin thermostat i.e. in a NVT ensamble. The friction coefficients used for the thermostat is 0.01.

### Total energy

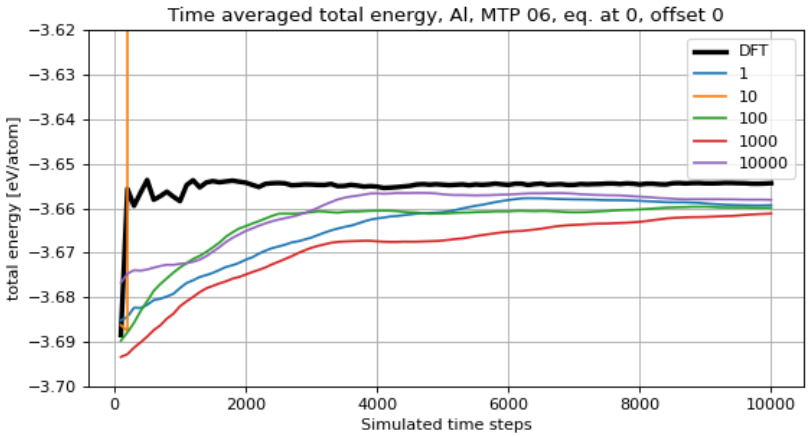
From fig. 7.1(a) it is observed that the DFT oscillations are reasonably replicated by the MTPs and that the time averages of all the MTPs in fig. 7.1(b) converge after being trained on approx. 6000 time steps. The MTPs converge towards approx. the same value around -3.66 eV / atom, differing with a few meV from the converged DFT value in the limit. Offsetting with 2000 as in fig. 7.1(c) does not significantly improve the time averages.

Worth to note, which is true for all plots with offset in this chapter and chapter 8, is that it would of course be possible to offset the simulation in fig. 7.1(b) to produce the respective simulation in fig. 7.1(c). However, the plots with offset have been simulated separately. Also worth to note is that the MTP simulations exhibit a significant variance. This is presented in section 7.2.

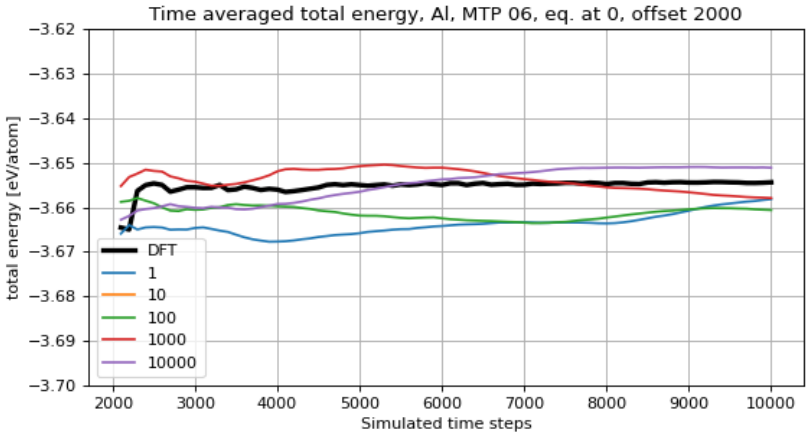
Similarly, from fig. 7.2(a) it is observed that the DFT oscillations are reasonably replicated by the MTPs and that the time averages in of all the MTPs in fig. 7.2(b) converge to approx. the same value, at first seemingly more in accordance to the DFT convergence than the equivalent fig. 7.1(b). However, the MTPs in fig. 7.2(c) seem to behave similarly to fig. 7.1(b), disputing the at first glance increase in accuracy when training on time steps starting from the 2000th compared to training on time steps starting from the first. Furthermore, it is worth to note that these simulations exhibit variance, which is presented in section 7.2. It is also observed




(a) Instantaneous total energy for Aluminium simulated with a 06 MTP trained on time steps starting from the first time step, with offset 0.

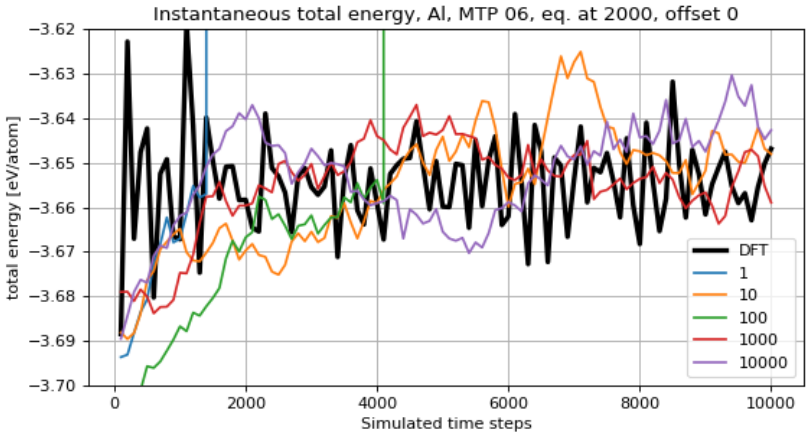



(b) Time averaged total energy for Aluminium simulated with a 06 MTP trained on time steps starting from the first time step, with offset 0.

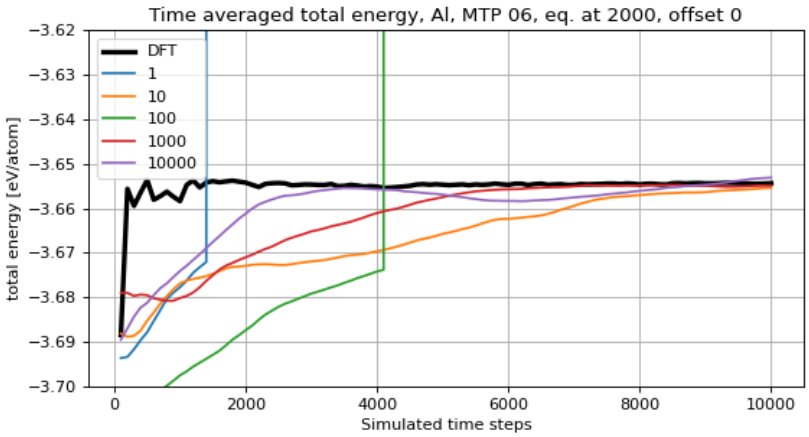



(c) Time averaged total energy for Aluminium simulated with a 06 MTP trained on time steps starting from the first time step, with offset 2000.

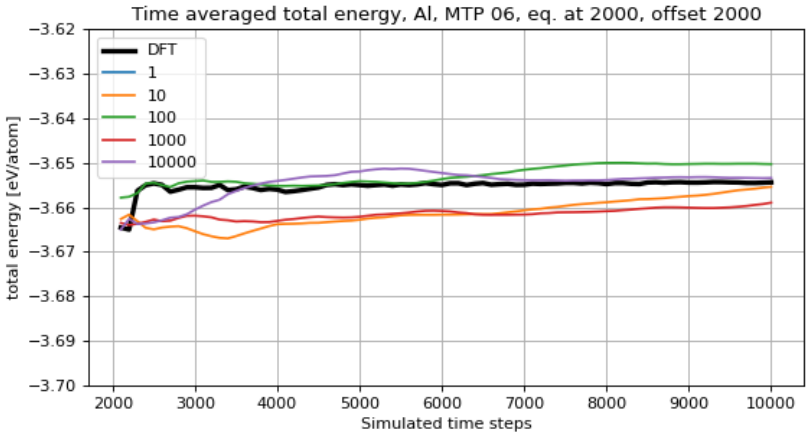
Figure 7.1: Instantaneous and time averaged total energy for Aluminium simulated with a 06 MTP trained on time steps starting from the first time step, with offset 0 and 2000.




(a) Instantaneous total energy for Aluminium simulated with a 06 MTP trained on time steps starting from the 2000th time step, with offset 0.




(b) Time averaged total energy for Aluminium simulated with a 06 MTP trained on time steps starting from the 2000th time step, with offset 0.




(c) Time averaged total energy for Aluminium simulated with a 06 MTP trained on time steps starting from the 2000th time step, with offset 2000.

Figure 7.2: Instantaneous and time averaged total energy for Aluminium simulated with a 06 MTP trained on time steps starting from the 2000th time step, with offset 0 and 2000.

that the MTPs trained on 1 and 100 time steps diverge in the simulation, see for example fig. 7.2(b) where it is also observed that the MTPs trained on more data converge earlier, see for example the MTP trained on 10000 time steps which converges around 4000 time steps while the MTP trained on 1000 converges after around 6000 time steps.

Using the MTP 10, increasing the free parameters, does not seemingly increase the the performance of the MTPs, as can for example be observed in fig. 7.4(b) where the potentials similarly to fig. 7.1(b) converge after around 6000 time steps with approx. the same error margin.

It is once again observed that training for time steps starting from the 2000th does not significantly increase the performance of the MTPs, even for MTP 10. The convergence in fig. 7.4 occurs around 6000 time steps with an equivalent errors compared to DFT.

It is observed from these figures of the total energy for Aluminium figs. 7.1, 7.2, 7.3 and 7.4 that the DFT simulations converge to approx. $-3.65438$ eV / atom for both the offsets 0 and 2000.

### Mean square displacement

From fig. 7.5(a) it is observed that the oscillations are reasonably replicated by the MTPs and from the time averages in fig. 7.5(b) it is observed that all MTPs except for the ones trained on 1 and 10 time steps converge to the DFT value after around 4000 time steps. The MTP trained on 1 time step, as would be expected, converge to a value that significantly differs from the from the DFT value. The MTP trained on 10 time steps diverges. In fig. 7.5(b) the MTPs trained on 100, 100 and 10000 time steps converge to the DFT value with an error in the magnitude of $10^{-5}$ Å$^2$, after around 4000 time steps.

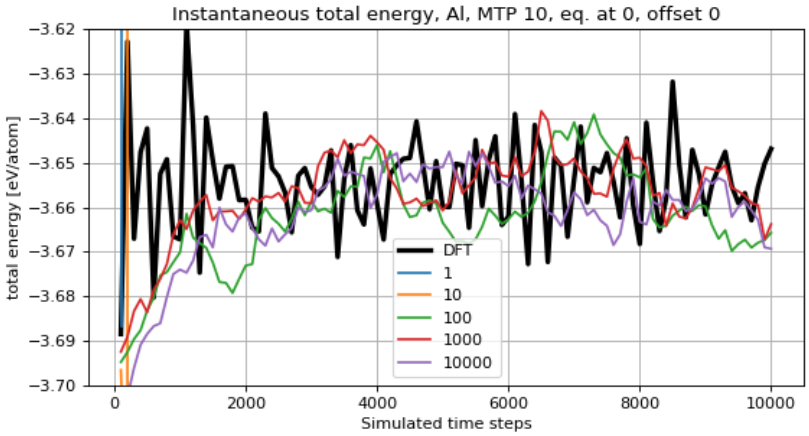
In fig. 7.5(c) at a first glance it seems like using the offset worsens the MTPs' performance. This is however most likely explained by the fact that when the offset is calculated, each time step is compared to the configuration of the atom system at the start of the sampling, which is after 2000 time steps. It is very likely that each MTP simulation would have different atomic positions as reference and therefore converge to different values. In fig. 7.5(b) the MTPs all have the same reference positions, which are the perfect lattice positions. For clarity, this is the case for all MSD plots with offset in this chapter and chapter 8.

The DFT oscillations in fig. 7.6(a) are once again reasonably replicated by the MTPs. In fig. 7.6(b) it is observed that the MTPs trained on 1 and 100 time steps diverge, and that the MTPs that do converge do so at different times. The MTP trained on 10000 time steps converges at around 2000 time steps, the MTP trained on 1000 time steps converges at around 5000 time steps and the MTP trained on 10 time steps converges after around 8000 time steps. The error of these converging MTPs are in the order of magnitude of $10^{-5}$. The MTPs in fig. 7.6(c) shows the MTPs converging to significantly different values.

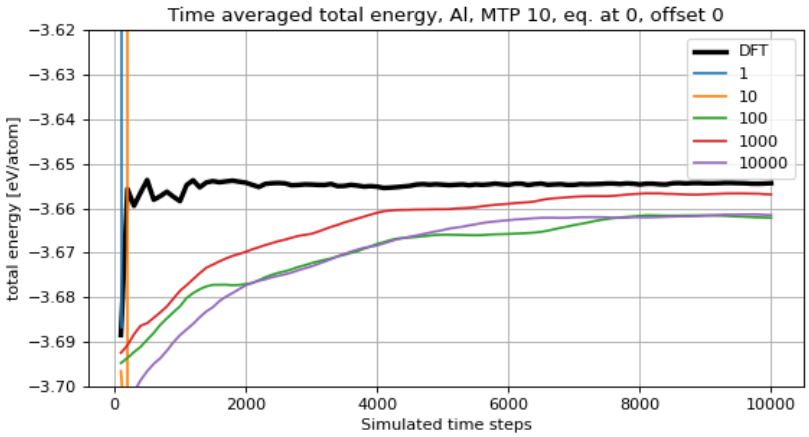
The DFT oscillations are reasonably replicated by the MTPs in fig. 7.7(a). In fig. 7.7(b) it is observed that the MTPs trained on 1 and 10 time steps diverge, and the other MTPs converge to the DFT value after around 5000 time steps. In the limit of 10000 time steps the MTPs yield errors in the magnitude of $10^{-5}$ compared to the DFT value. In fig. 7.7(c) the converging MTPs converge to different values.

The DFT oscillations are reasonably replicated by the MTPs in fig. 7.8(a). In fig. 7.8(c) it is observed that the MTPs converge to the DFT value after around 6000 time steps. In the limit of 10000 time steps the MTPs yield errors in the magnitude of $10^{-5}$ compared to the DFT value. In fig. 7.8(c) the converging MTPs converge to significantly different values.

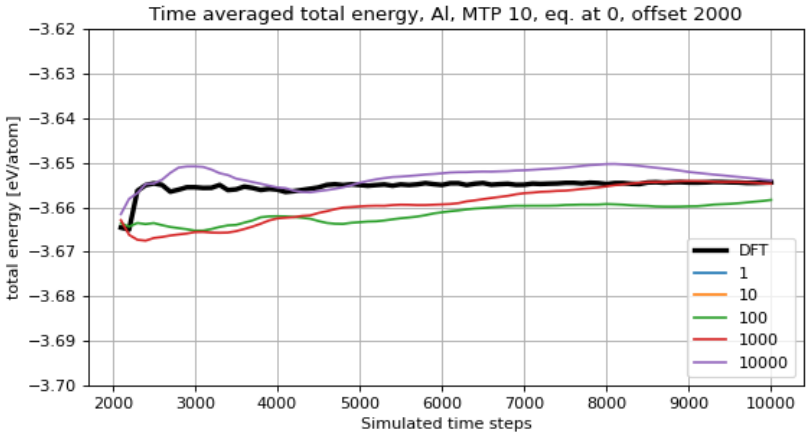
It is observed from these figures of the mean squared displacement for Aluminium figs. 7.5, 7.6, 7.7 and 7.8 that the DFT simulations converge to approx. 0.0004 and 0.0007 Å$^2$ for the offsets 0 and 2000 respectively.




(a) Instantaneous total energy for Aluminium simulated with a 10 MTP trained on time steps starting from the first time step, with offset 0.

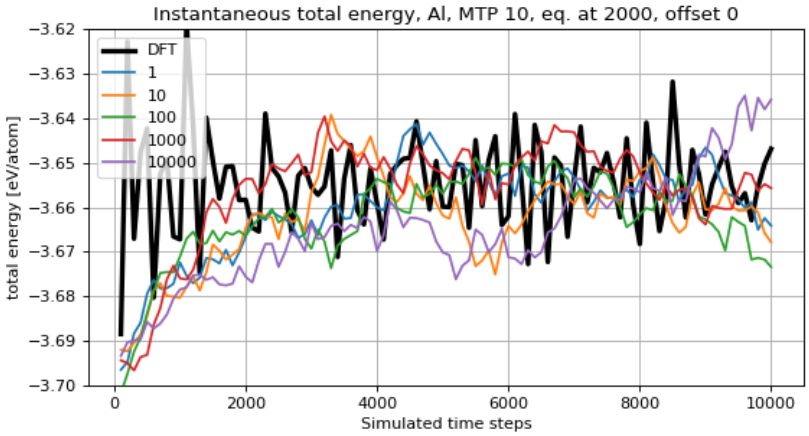



(b) Time averaged total energy for Aluminium simulated with a 10 MTP trained on time steps starting from the first time step, with offset 0.

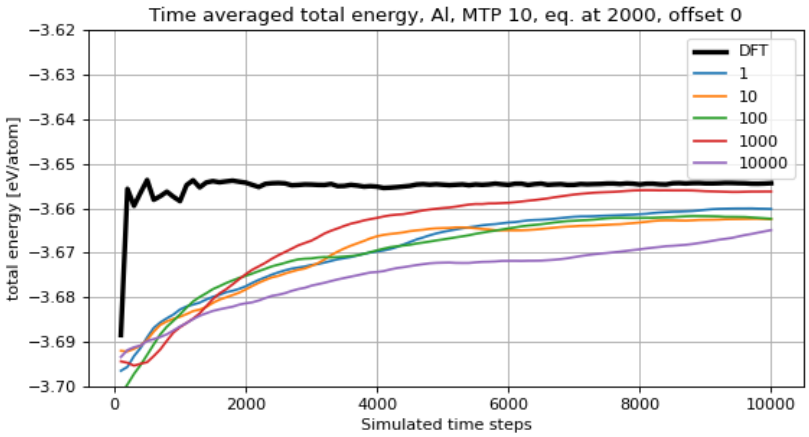



(c) Time averaged total energy for Aluminium simulated with a 10 MTP trained on time steps starting from the first time step, with offset 2000.

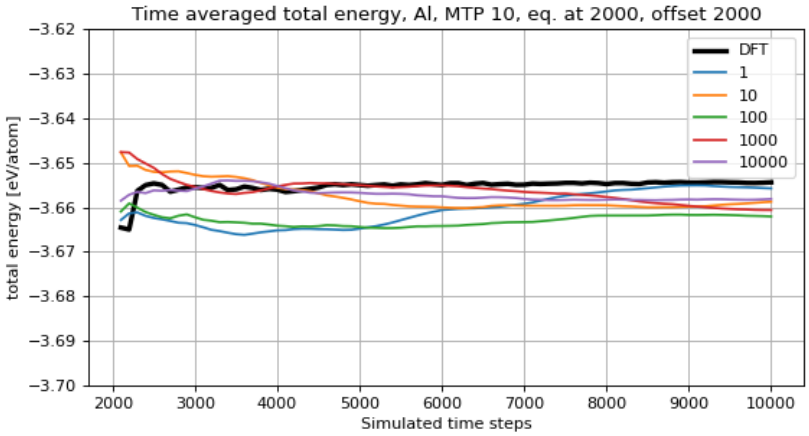
Figure 7.3: Instantaneous and time averaged total energy for Aluminium simulated with a 10 MTP trained on time steps starting from the first time step, with offset 0 and 2000.




(a) Instantaneous total energy for Aluminium simulated with a 10 MTP trained on time steps starting from the 2000th time step, with offset 0.

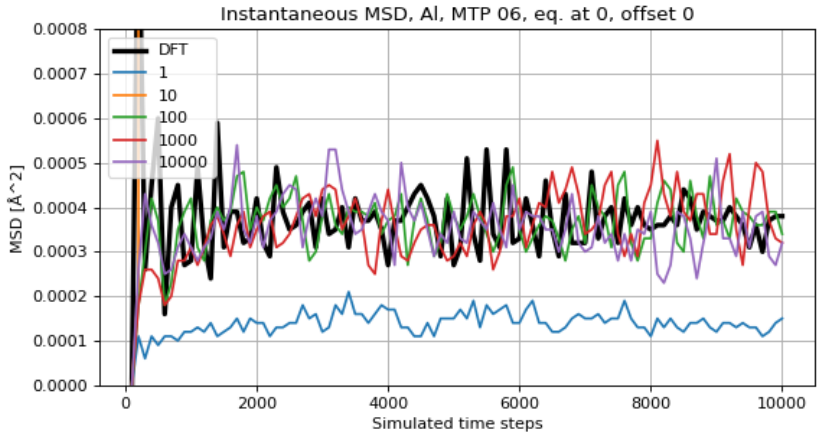



(b) Time averaged total energy for Aluminium simulated with a 10 MTP trained on time steps starting from the 2000th time step, with offset 0.

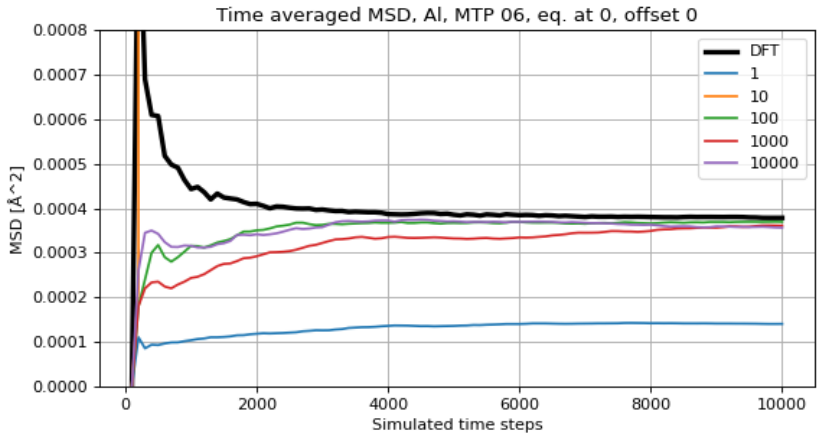



(c) Time averaged total energy for Aluminium simulated with a 10 MTP trained on time steps starting from the 2000th time step, with offset 0.

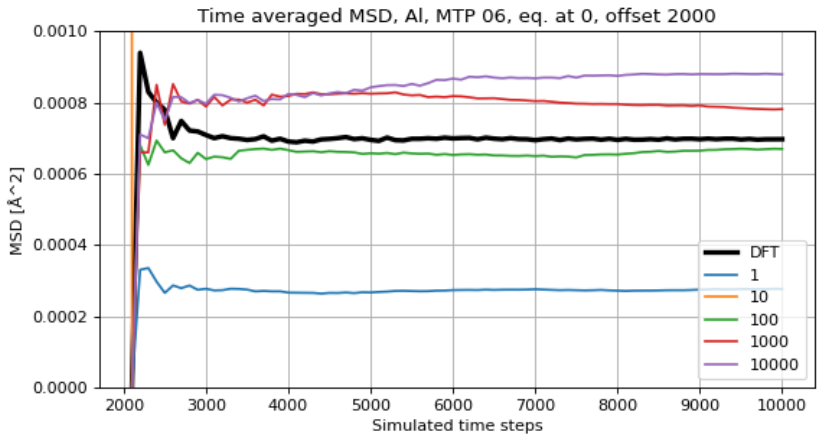
Figure 7.4: Instantaneous and time averaged total energy for Aluminium simulated with a 10 MTP trained on time steps starting from the 2000th time step, with offset 0 and 2000.




(a) Instantaneous mean squared displacement for Aluminium simulated with a 06 MTP trained on time steps starting from the first time step, with offset 0.

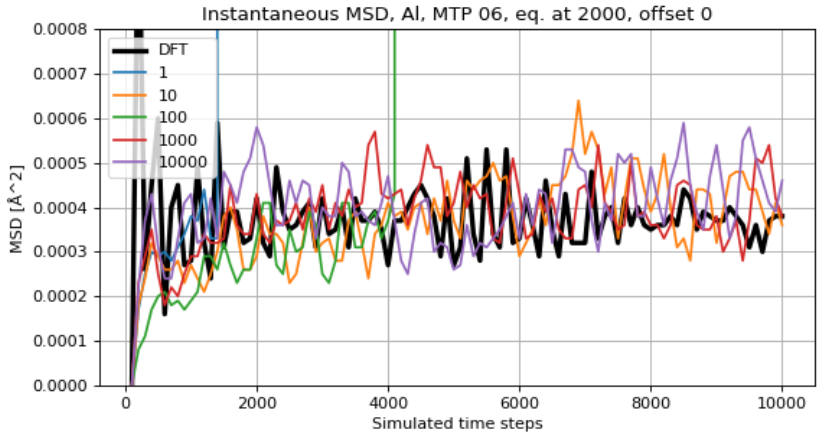



(b) Time averaged mean squared displacement for Aluminium simulated with a 06 MTP trained on time steps starting from the first time step, with offset 0.

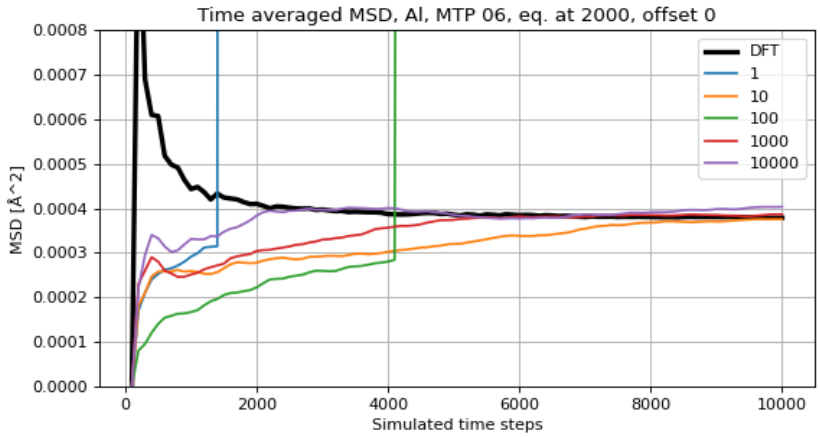



(c) Time averaged mean squared displacement for Aluminium simulated with a 06 MTP trained on time steps starting from the first time step, with offset 2000.

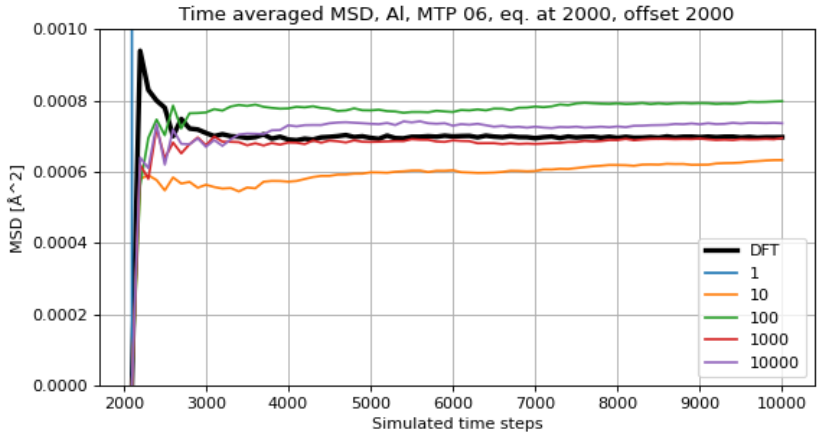
Figure 7.5: Instantaneous and time averaged mean squared displacement for Aluminium simulated with a 06 MTP trained on time steps starting from the first time step, with offset 0 and 2000.




(a) Instantaneous mean squared displacement for Aluminium simulated with a 06 MTP trained on time steps starting from the 2000th time step, with offset 0.

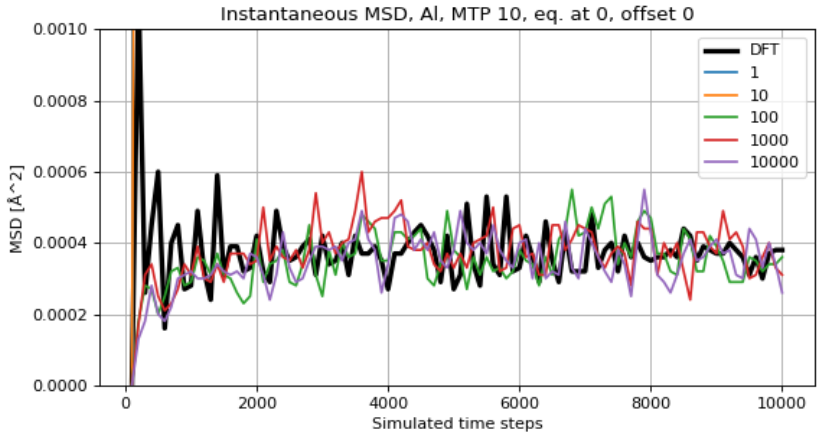



(b) Time averaged mean squared displacement for Aluminium simulated with a 06 MTP trained on time steps starting from the 2000th time step, with offset 0.

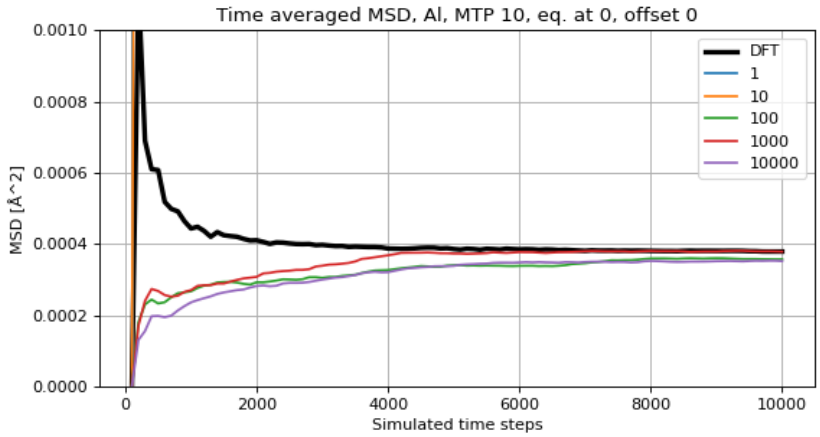



(c) Time averaged mean squared displacement for Aluminium simulated with a 06 MTP trained on time steps starting from the 2000th time step, with offset 2000.

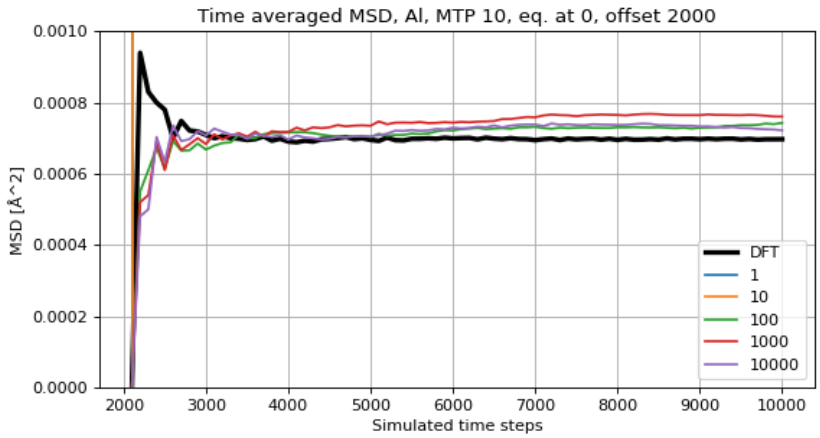
Figure 7.6: Instantaneous and time averaged mean squared displacement for Aluminium simulated with a 06 MTP trained on time steps starting from the 2000th time step, with offset 0 and 2000.




(a) Instantaneous mean squared displacement for Aluminium simulated with a 10 MTP trained on time steps starting from the first time step, with offset 0.

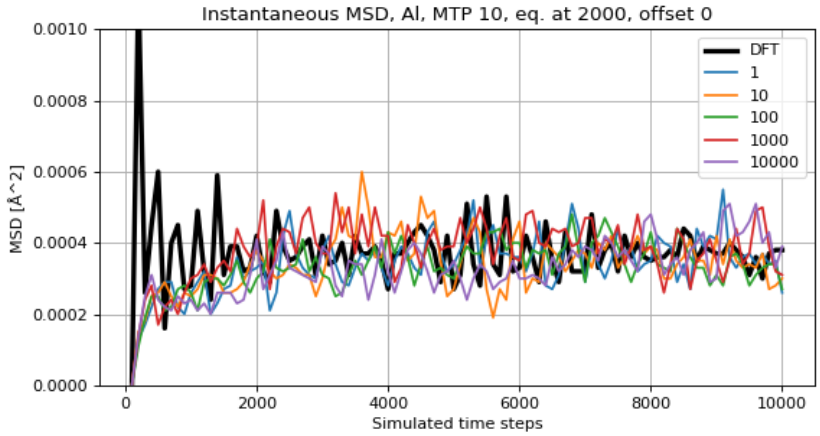



(b) Time averaged mean squared displacement for Aluminium simulated with a 10 MTP trained on time steps starting from the first time step, with offset 0.

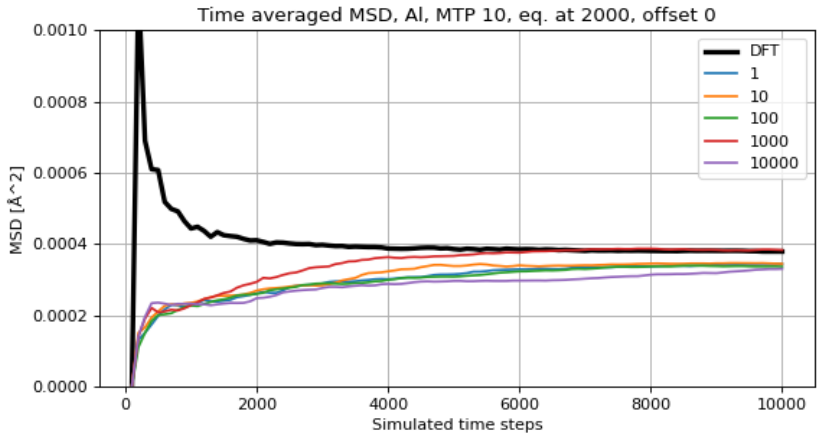



(c) Time averaged mean squared displacement for Aluminium simulated with a 10 MTP trained on time steps starting from the first time step, with offset 2000.

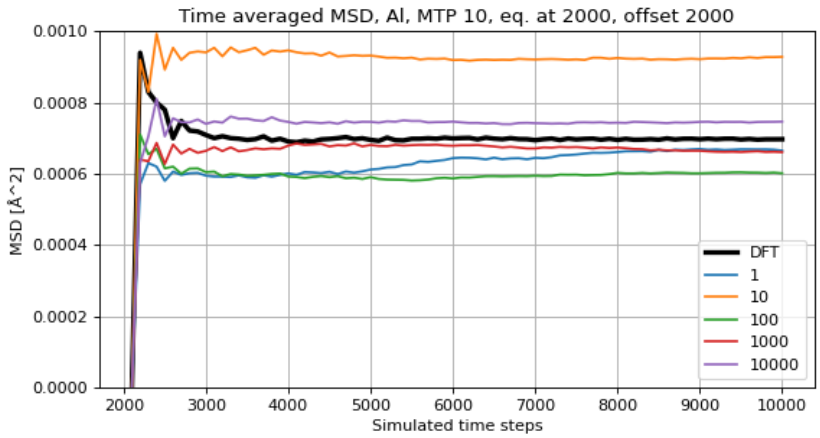
Figure 7.7: Instantaneous and time averaged mean squared displacement for Aluminium simulated with a 10 MTP trained on time steps starting from the first time step, with offset 0 and 2000.




(a) Instantaneous mean squared displacement for Aluminium simulated with a 10 MTP trained on time steps starting from the 2000th time step, with offset 0.

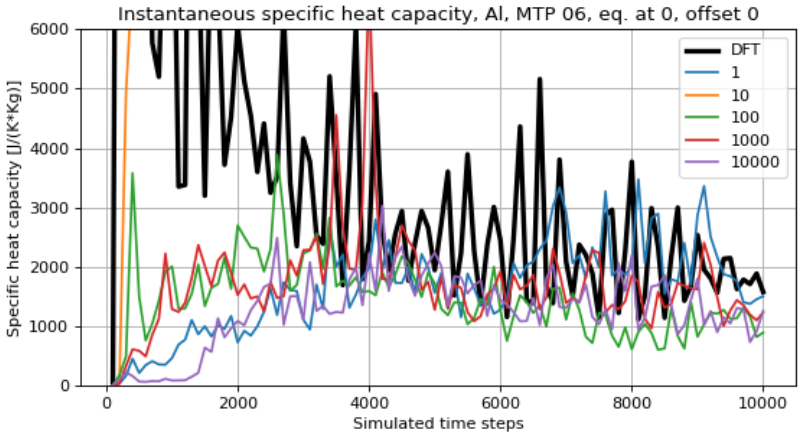



(b) Time averaged mean squared displacement for Aluminium simulated with a 10 MTP trained on time steps starting from the 2000th time step, with offset 0.

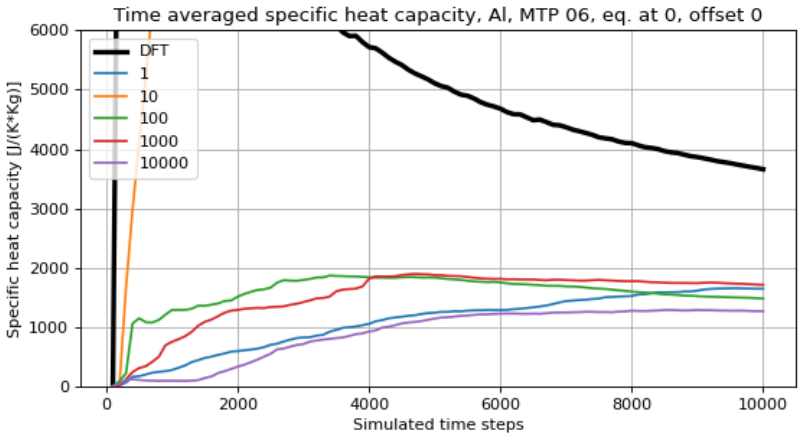



(c) Time averaged mean squared displacement for Aluminium simulated with a 10 MTP trained on time steps starting from the 2000th time step, with offset 2000.

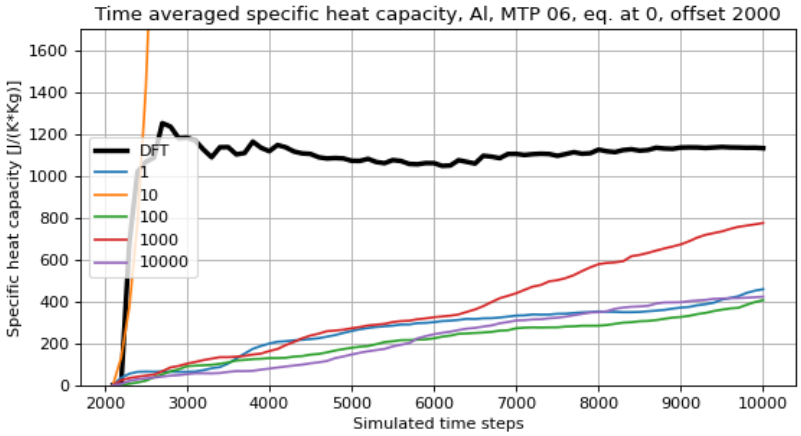
Figure 7.8: Instantaneous and time averaged mean squared displacement for Aluminium simulated with a 10 MTP trained on time steps starting from the 2000th time step, with offset 0 and 2000.




(a) Instantaneous specific heat capacity for Aluminium simulated with a 06 MTP trained on time steps starting from the first time step, with offset 0.




(b) Time averaged specific heat capacity for Aluminium simulated with a 06 MTP trained on time steps starting from the first time step, with offset 0.




(c) Time averaged specific heat capacity for Aluminium simulated with a 06 MTP trained on time steps starting from the first time step, with offset 2000.

Figure 7.9: Instantaneous and time averaged specific heat capacity for Aluminium simulated with a 06 MTP trained on time steps starting from the first time step, with offset 0 and 2000.

### Specific heat capacity

As is observed in fig. 7.9(a) the DFT oscillations decrease in amplitude around 4000 time steps and that the MTPs reproduce the DFT oscillations fairly after 4000 time steps. It is also observed that the MTP that is trained on 10 time steps diverges. In fig. 7.9(b) it is observed that the MTPs converge after around 4000, and that the DFT does not seemingly converge during simulation time, as the DFT plot after 10000 time steps slopes downwards. In fig. 7.9(c) the DFT simulation does converge rather quickly, meanwhile the MTPs do not converge as the plots end in an upwards slope.

As is observed in fig. 7.10(a) the DFT oscillations decrease in amplitude around 4000 time steps and that the MTPs reproduce the DFT oscillations fairly after 4000 time steps. It is also observed that the MTPs that are trained on 1 and 100 time steps diverge. In fig. 7.10(b) it is observed that the MTPs converge after around 4000, and that the DFT does not seemingly converge during simulation time, as the DFT plot after 10000 time steps slopes downwards. In fig. 7.9(c) the DFT simulation does converge rather quickly. It is also observed that the MTP trained on 1 time step diverges in this simulation and that the other MTPs converge after approx. 4000 time steps. The MTPs converge to values with an error in the magnitude of $10^2$ compared to the DFT value.

As is observed in fig. 7.11(a) the DFT oscillations decrease in amplitude around 4000 time steps and that the MTPs reproduce the DFT oscillations fairly after 4000 time steps. It is also observed that the MTPs that are trained on 1 and 10 time steps diverge. In fig. 7.11(b) it is observed that the MTPs converge after around 4000, and that the DFT does not seemingly converge during simulation time, as the DFT plot after 10000 time steps slopes downwards. In fig. 7.11(c) the DFT simulation does converge rather quickly, meanwhile the MTPs do not converge as the plots end in an upwards slope.

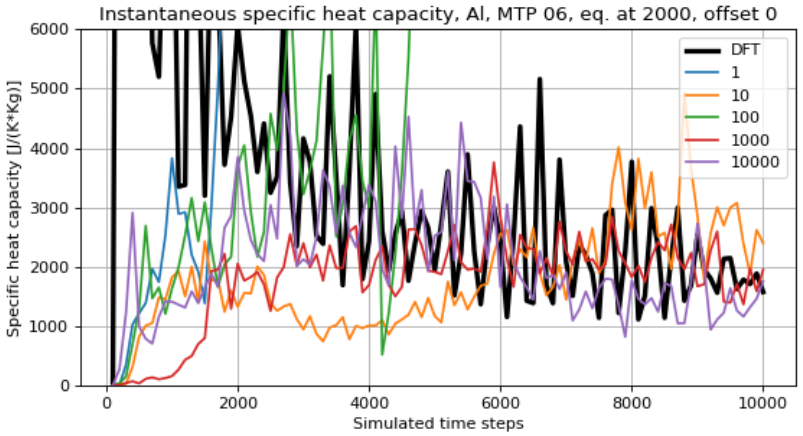
As is observed in fig. 7.12(a) the DFT oscillations decrease in amplitude around 4000 time steps and that the MTPs reproduce the DFT oscillations fairly after 4000 time steps. In fig. 7.12(b) it is observed that the MTPs converge after around 4000, and that the DFT does not seemingly converge during simulation time, as the DFT plot after 10000 time steps slopes downwards. In fig. 7.11(c) the DFT simulation does converge rather quickly. It is also observed that the MTPs trained on 10, 100 and 10000 time steps converge after approx. 4000 time steps. The MTPs converge to values with an error in the magnitude of $10^2$ compared to the DFT value.

It is observed from these figures of the specific heat capacity for Aluminium figs. 7.5, 7.6, 7.7 and 7.8 that the DFT simulations does not converge for offset 0 but does converge to approx. 1145 J / (K $\cdot$ Kg) for offset 2000.

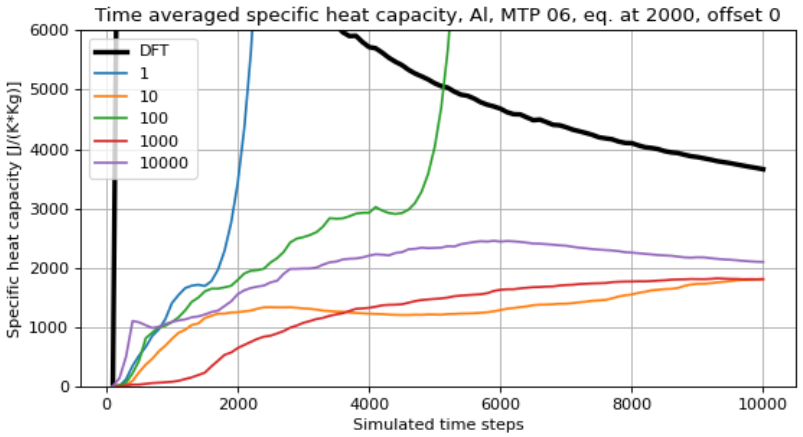
## 7.2 MTP simulation variance

This section contains plots that illustrate the variance of the MTP simulations i.e. how separate instances of the same MTP fitted to the same training data yield different results when simulated in succession. Fig. 7.13 shows time averaged values for the total energy, mean square displacement and specific heat capacity for 10 additional fittings of the MTP 06 potential fitted on the same 100 time steps starting from the first time step with offset 0 for Aluminium. The additional fittings are shown in a more transparent green colour. These plots illustrate the variance of the MTP simulations.

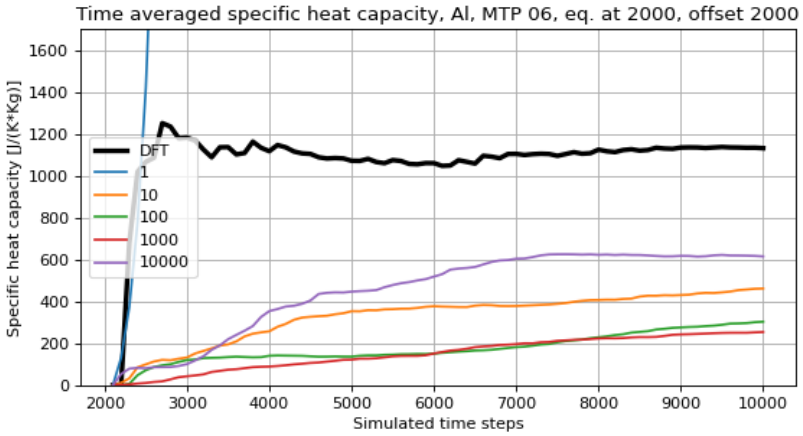
As is observed in figs. 7.13(a), 7.13(b) and 7.13(c), the variance of the total energy is in the magnitude of $10^{-2}$, the variance of the MSD is in the magnitude $10^{-5}$ and the variance of the specific heat capacity is in the magnitude of $10^3$.




(a) Instantaneous specific heat capacity for Aluminium simulated with a 06 MTP trained on time steps starting from the 2000th time step, with offset 0.

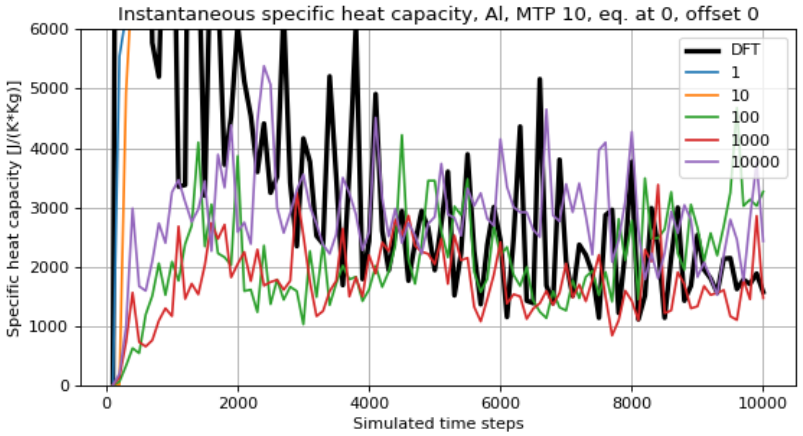



(b) Time averaged specific heat capacity for Aluminium simulated with a 06 MTP trained on time steps starting from the 2000th time step, with offset 0.

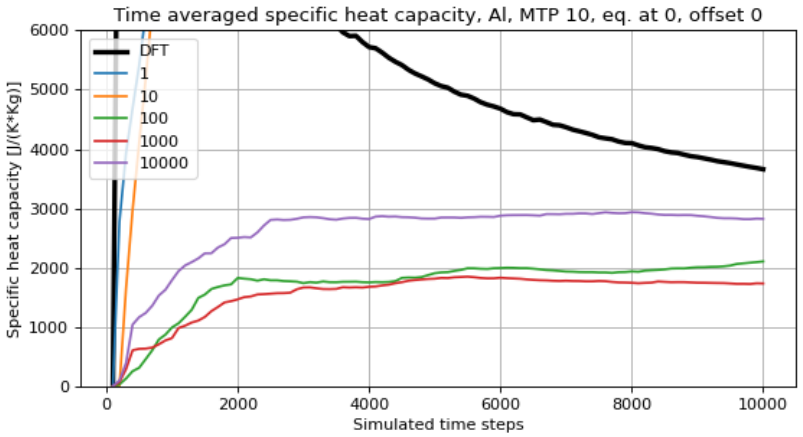



(c) Time averaged specific heat capacity for Aluminium simulated with a 06 MTP trained on time steps starting from the 2000th time step, with offset 2000.

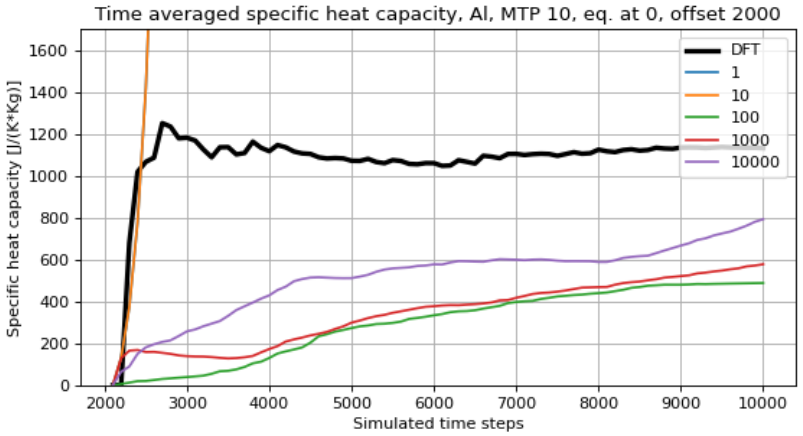
Figure 7.10: Instantaneous and time averaged specific heat capacity for Aluminium simulated with a 06 MTP trained on time steps starting from the 2000th time step, with offset 0 and 2000.




(a) Instantaneous specific heat capacity for Aluminium simulated with a 10 MTP trained on time steps starting from the first time step, with offset 0.

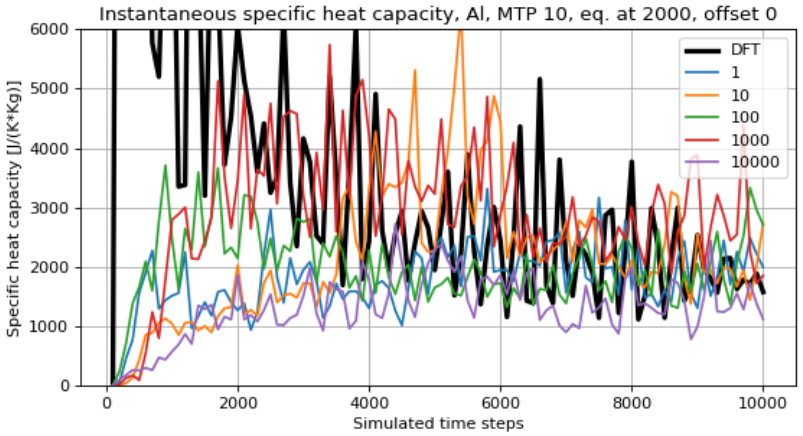



(b) Time averaged specific heat capacity for Aluminium simulated with a 10 MTP trained on time steps starting from the first time step, with offset 0.

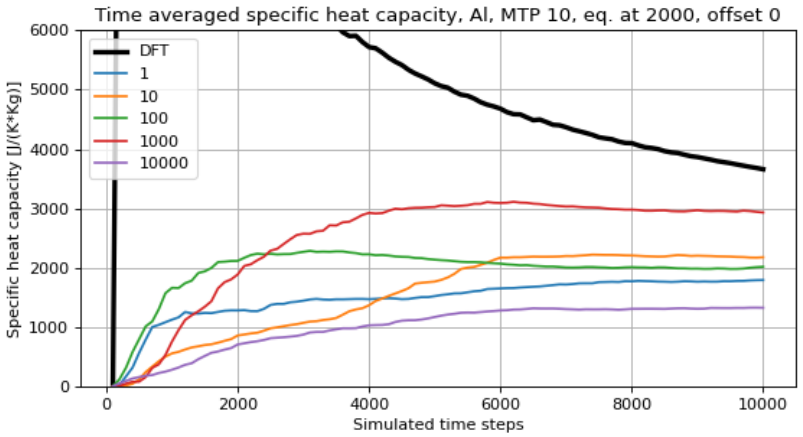



(c) Time averaged specific heat capacity for Aluminium simulated with a 10 MTP trained on time steps starting from the first time step, with offset 2000.

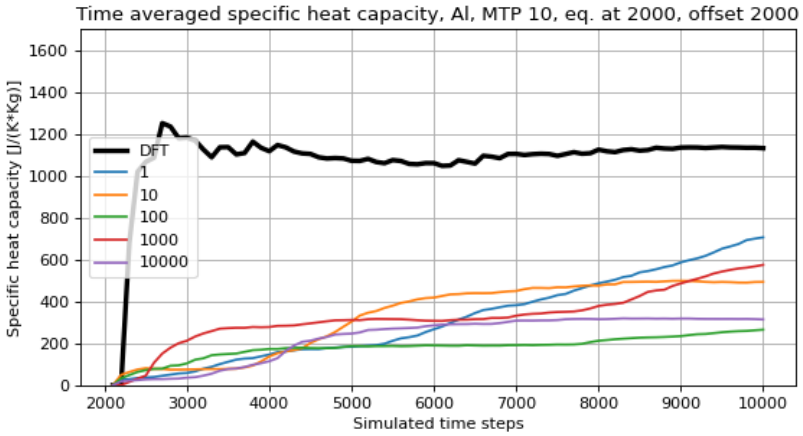
Figure 7.11: Instantaneous and time averaged specific heat capacity for Aluminium simulated with a 10 MTP trained on time steps starting from the first time step, with offset 0 and 2000.




(a) Instantaneous specific heat capacity for Aluminium simulated with a 10 MTP trained on time steps starting from the 2000th time step, with offset 0.

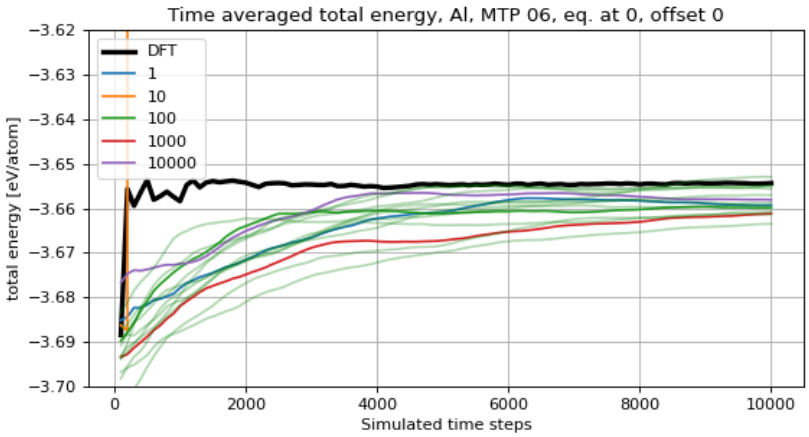



(b) Time averaged specific heat capacity for Aluminium simulated with a 10 MTP trained on time steps starting from the 2000th time step, with offset 0.

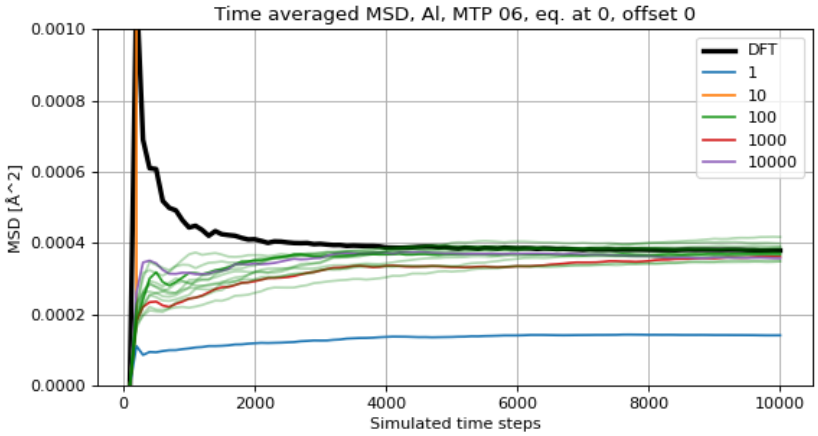



(c) Time averaged specific heat capacity for Aluminium simulated with a 10 MTP trained on time steps starting from the 2000th time step, with offset 2000.

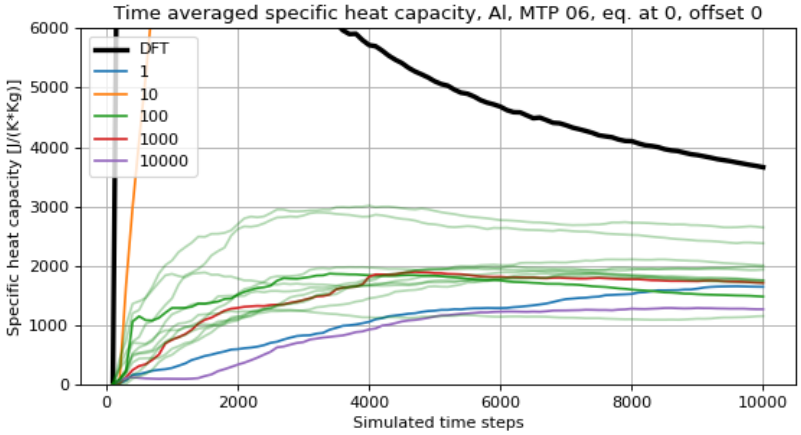
Figure 7.12: Instantaneous and time averaged specific heat capacity for Aluminium simulated with a 10 MTP trained on time steps starting from the 2000th time step, with offset 0 and 2000.




(a) Time averaged total energy for Aluminium simulated with a 06 MTP trained on time steps starting from the first time step, with offset 0.




(b) Time averaged mean square displacement for Aluminium simulated with a 06 MTP trained on time steps starting from the first time step, with offset 0.




(c) Time averaged specific heat capacity for Aluminium simulated with a 06 MTP trained on time steps starting from the first time step, with offset 0.

Figure 7.13: Time averaged total energy, mean square displacement and specific heat capacity for Aluminium simulated with a 06 MTP trained on time steps starting from the first time step, with offset 0. The transparent green lines visualise 10 additional 06 MTPs fitted to the same 100 time steps.

# 8 Silicon simulations

In the section 8.1 in this chapter the calculated physical properties' instantaneous and time averaged values are presented for Silicon. In section 5.5 the comparative run times between the different simulation methods are presented.

## 8.1 Calculated physical properties

The instantaneous and time averaged values of the calculated physical properties total energy, mean squared displacement and specific heat are presented in this section. The instantaneous values are plotted to show if the properties oscillate reasonably and the time averaged values are plotted to show whether the properties converge to the equivalent DFT value. All these properties are calculated from MD simulations using the Langevin thermostat i.e. in a NVT ensamble. The friction coefficients used for the thermostat is 0.01.

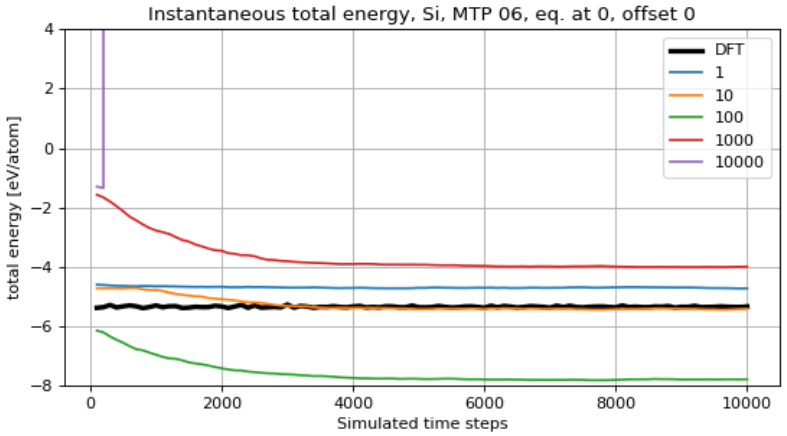
### Total energy

In fig. 8.1(a) it is observed that because of the scale, it is not possible to visually observe the oscillations of the MTPs. In fig. 8.1(b) it is observed that the MTP trained on 10000 time steps diverges and that the other MTPs converge to significantly different values. The same is observed in fig. 8.1(c). It is also observed that the DFT simulation converges to a value of approx. $-5.4$ eV.

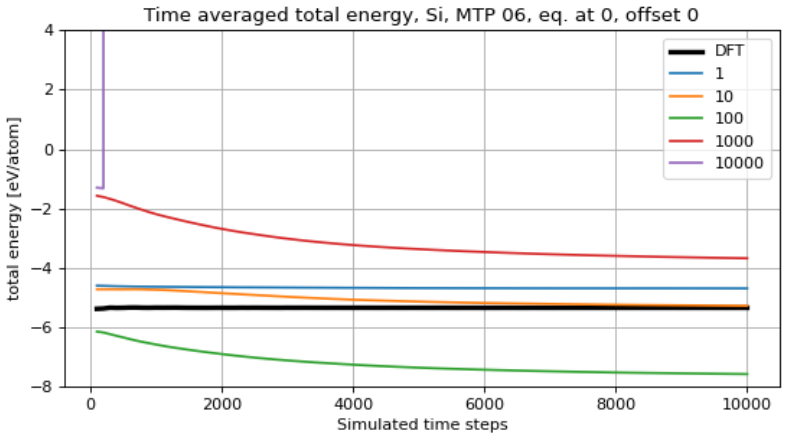
In fig. 8.2(a) it is observed that because of the scale, it is not possible to visually observe the oscillations of the MTPs. In fig. 8.2(b) it is observed that the MTPs converge to significantly different values. The same is observed in fig. 8.2(c), where some MTPs converge to values that are not even observable in the figure with the chosen axis values.

In fig. 8.3(a) it is observed that because of the scale, it is not possible to visually observe the oscillations of the MTPs. In fig. 8.3(b) it is observed that the MTP that is trained on 10000 time steps diverges and that the other MTPs converge to significantly different values. It is also observed that the MTP trained on 100 time steps exhibits outlying behaviour. In fig. 8.3(c), it is again observed that the MTPs converge to significantly different values.

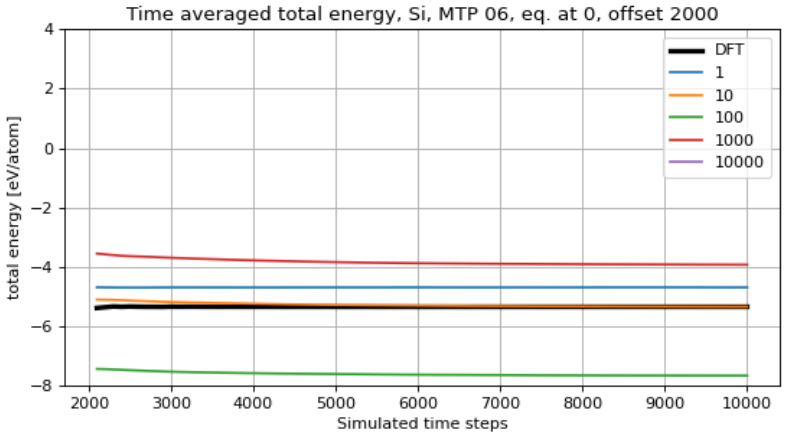
In fig. 8.4(a) it is observed that because of the scale, it is not possible to visually observe the oscillations of the MTPs. In fig. 8.4(b) it is observed that the MTP that is trained on 1000 time steps diverges and that the other MTPs converge to significantly different values. Most




(a) Instantaneous total energy for Silicon simulated with a 06 MTP trained on time steps starting from the first time step, with offset 0.

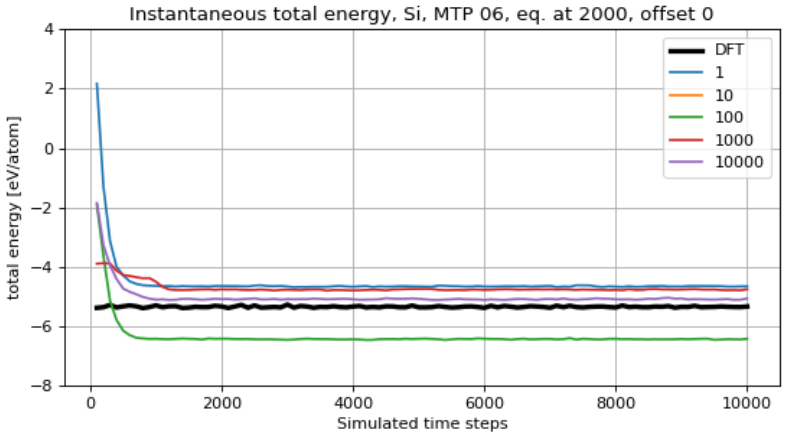



(b) Time averaged total energy for Silicon simulated with a 06 MTP trained on time steps starting from the first time step, with offset 0.

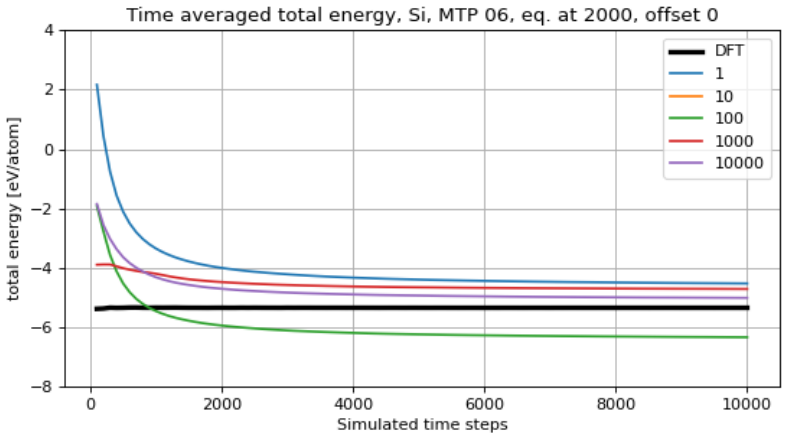



(c) Time averaged total energy for Silicon simulated with a 06 MTP trained on time steps starting from the first time step, with offset 2000.

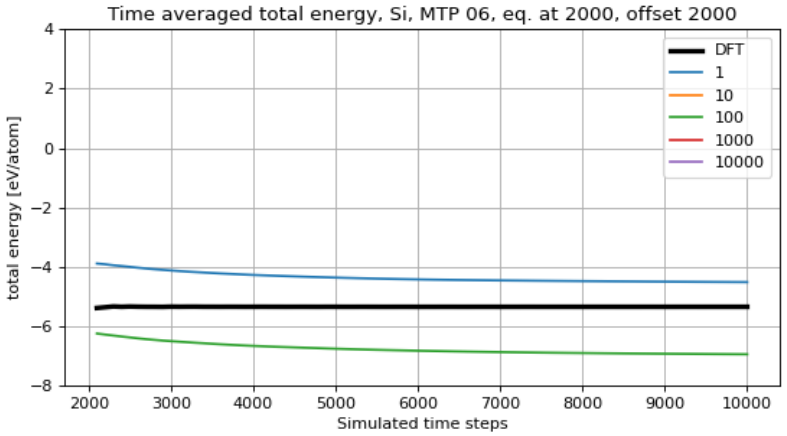
Figure 8.1: Instantaneous and time averaged total energy for Silicon simulated with a 06 MTP trained on time steps starting from the first time step, with offset 0 and 2000.




(a) Instantaneous total energy for Silicon simulated with a 06 MTP trained on time steps starting from the 2000th time step, with offset 0.

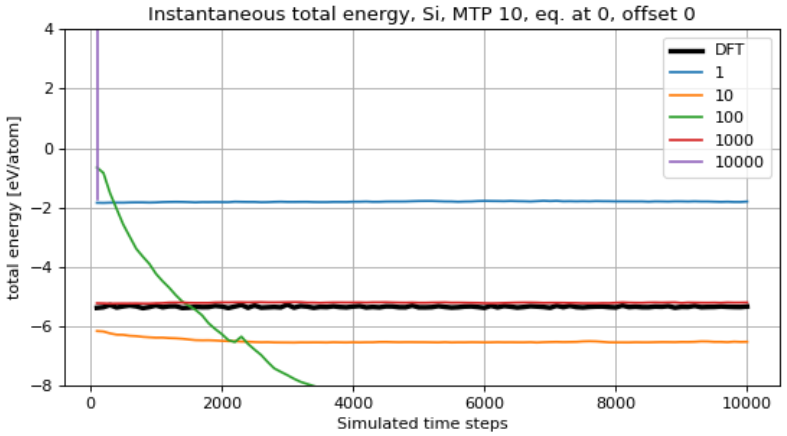



(b) Time averaged total energy for Silicon simulated with a 06 MTP trained on time steps starting from the 2000th time step, with offset 0.

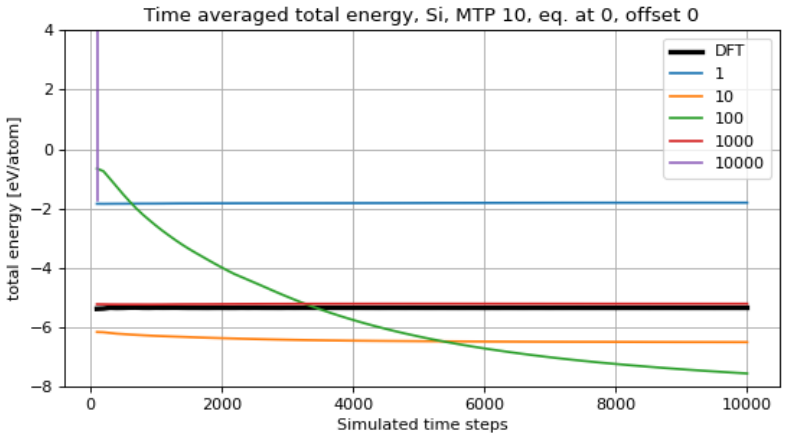



(c) Time averaged total energy for Silicon simulated with a 06 MTP trained on time steps starting from the 2000th time step, with offset 2000.

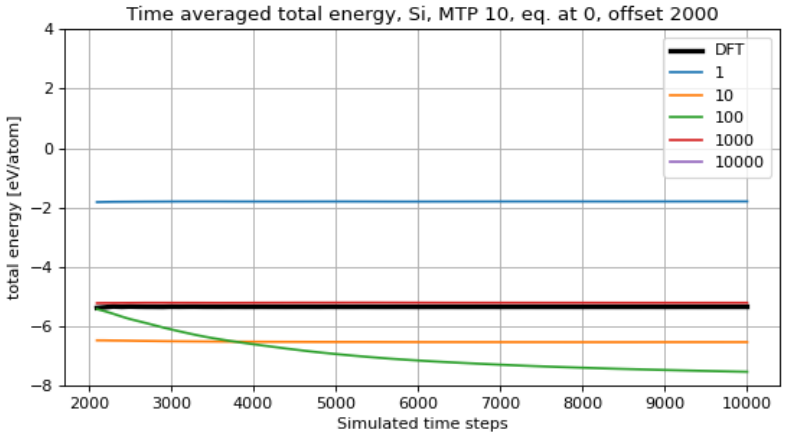
Figure 8.2: Instantaneous and time averaged total energy for Silicon simulated with a 06 MTP trained on time steps starting from the 2000th time step, with offset 0 and 2000.




(a) Instantaneous total energy for Silicon simulated with a 10 MTP trained on time steps starting from the first time step, with offset 0.

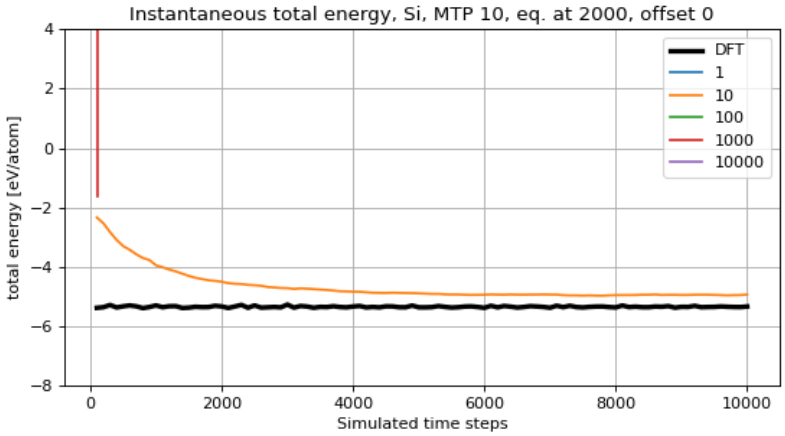



(b) Time averaged total energy for Silicon simulated with a 10 MTP trained on time steps starting from the first time step, with offset 0.

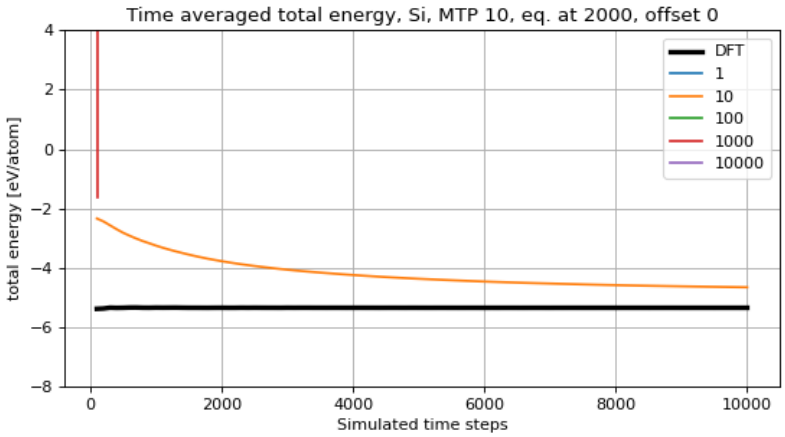



(c) Time averaged total energy for Silicon simulated with a 10 MTP trained on time steps starting from the first time step, with offset 2000.

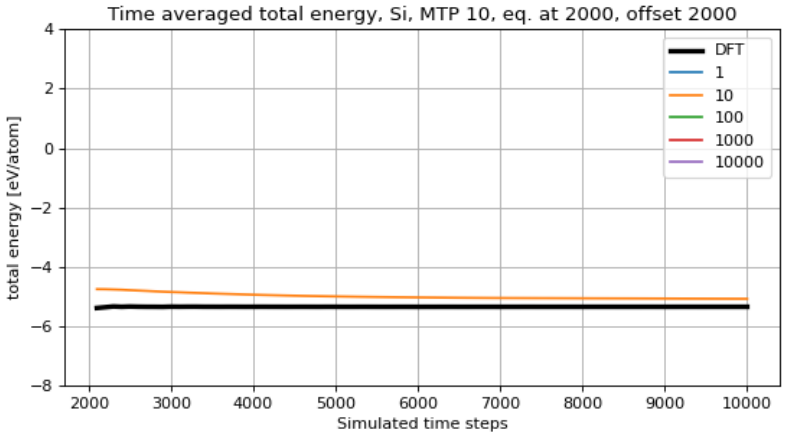
Figure 8.3: Instantaneous and time averaged total energy for Silicon simulated with a 10 MTP trained on time steps starting from the first time step, with offset 0 and 2000.




(a) Instantaneous total energy for Silicon simulated with a 10 MTP trained on time steps starting from the 2000th time step, with offset 0.




(b) Time averaged total energy for Silicon simulated with a 10 MTP trained on time steps starting from the 2000th time step, with offset 0.




(c) Time averaged total energy for Silicon simulated with a 10 MTP trained on time steps starting from the 2000th time step, with offset 2000.

Figure 8.4: Instantaneous and time averaged total energy for Silicon simulated with a 10 MTP trained on time steps starting from the 2000th time step, with offset 0 and 2000.

of the MTPs converge to values that cannot be observed as they converge to values that are not represented on the vertical axis. The same is similarly observed in fig. 8.4(c).

### Mean square displacement

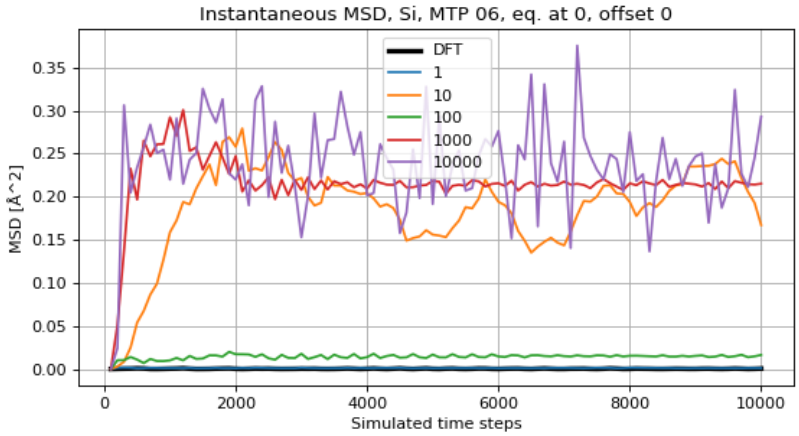
In fig. 8.5(a) it is observed that the MTPs show quite significant oscillations. In fig. 8.5(b) it is observed that the MTPs converge after around 2000 time steps and that they converge to significantly different values in an order of magnitude of $10^{-1}$. In fig. 8.5(c) it is also observed that the MTPs converge to significantly different values.

In fig. 8.6(a) it is observed that the MTPs show quite significant oscillations. In fig. 8.6(b) it is observed that the MTPs converge after around 1000 time steps and that they converge to significantly different values in an order of magnitude of $10^{-1}$. It is also observed that the MTPs trained on 1, 100 and 10000 time steps converge to approx. the same value. In fig. 8.6(c) it is also observed that the MTPs converge to significantly different values and that the MTPs trained on 10, 1000, and 10000 time steps converge to approx. the same value.

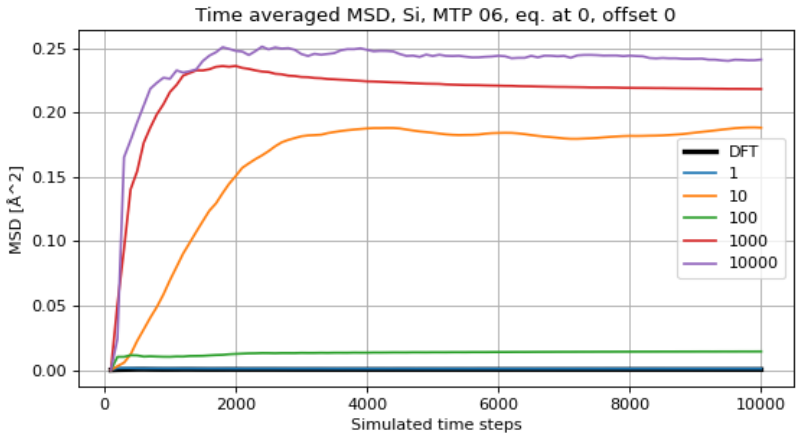
In fig. 8.7(a) it is observed that the MTPs show quite significant oscillations. In fig. 8.7(b) it is observed that the MTPs converge after around 2000 time steps and that they converge to significantly different values in an order of magnitude of $10^{-1}$. In fig. 8.7(c) it is also observed that the MTPs converge to significantly different values.

In fig. 8.8(a) it is observed that the MTPs show quite significant oscillations. In fig. 8.8(b) it is observed that the MTPs converge after around 1000 time steps and that they converge to significantly different values in an order of magnitude of $10^{-1}$. It is also observed that the MTPs trained on 100, 1000 and 10000 time steps converge to approx. the same value. In fig. 8.8(c) it is also observed that the MTPs converge to significantly different values and that the MTPs trained on 10, 1000, and 10000 time steps converge to approx. the same value. Note, in figs. 8.8(a) and 8.8(b) the MTP trained on 10 in not included and in fig. 8.8(c) the legend does not include 10 but the MTP trained on 10 time steps is included in the graph.

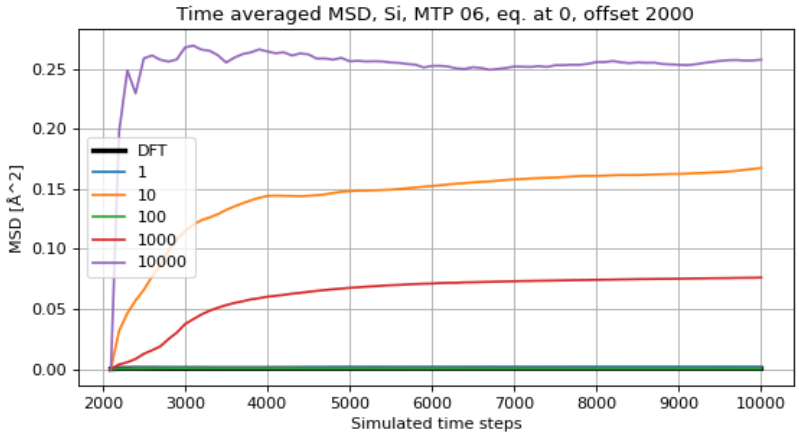
From the plots of total energy and mean squared displacement for Silicon, figs. 8.1, 8.2, 8.3, 8.4 and 8.5, 8.6, 8.7, 8.8 respectively, it is observed that the energies and MSDs in the Silicon simulations converge to very different values for the differently trained MTPs. This happens because in the simulation, the Silicon crystal melts and then reforms into different crystal structures with different potential energies. This is discussed in more detail in the discussion section 9.1. Therefore, physical properties calculated from the Silicon simulations cannot really be interpreted. Because of this, Silicon results are not included further in this chapter. Also, the anomalous behaviour of the MTP 10 trained for 100 time steps in fig. 8.3 is briefly detailed further in appendix B Anomalies.




(a) Instantaneous mean squared displacement for Silicon simulated with a 06 MTP trained on time steps starting from the first time step, with offset 0.

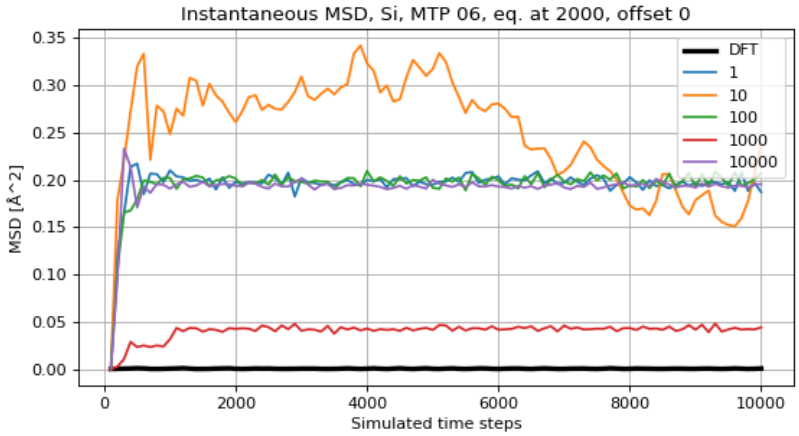



(b) Time averaged mean squared displacement for Silicon simulated with a 06 MTP trained on time steps starting from the first time step, with offset 0.

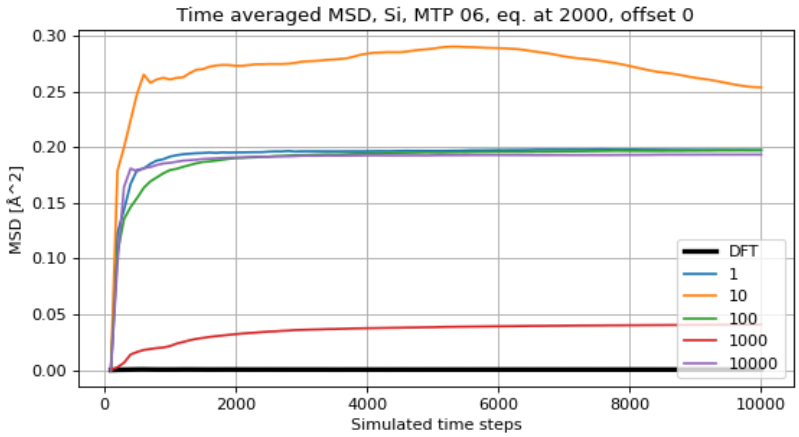



(c) Time averaged mean squared displacement for Silicon simulated with a 06 MTP trained on time steps starting from the first time step, with offset 2000.

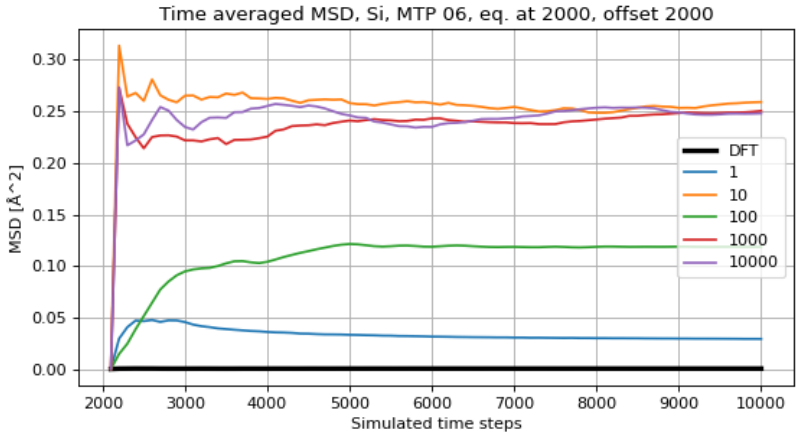
Figure 8.5: Instantaneous and time averaged mean squared displacement for Silicon simulated with a 06 MTP trained on time steps starting from the first time step, with offset 0 and 2000.




(a) Instantaneous mean squared displacement for Silicon simulated with a 06 MTP trained on time steps starting from the 2000th time step, with offset 0.

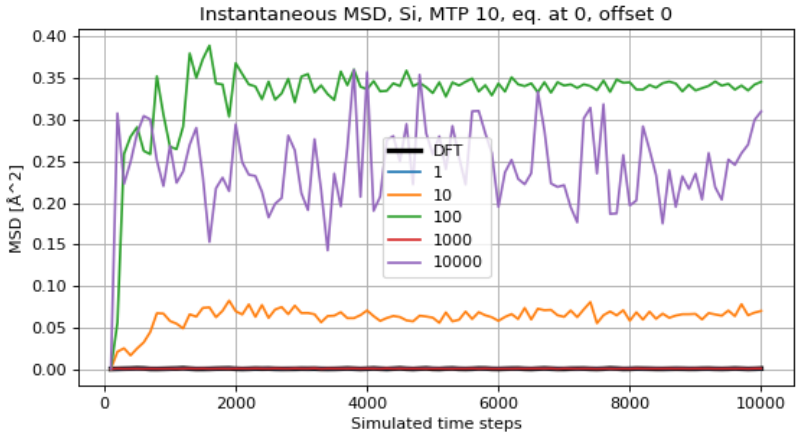



(b) Time averaged mean squared displacement for Silicon simulated with a 06 MTP trained on time steps starting from the 2000th time step, with offset 0.

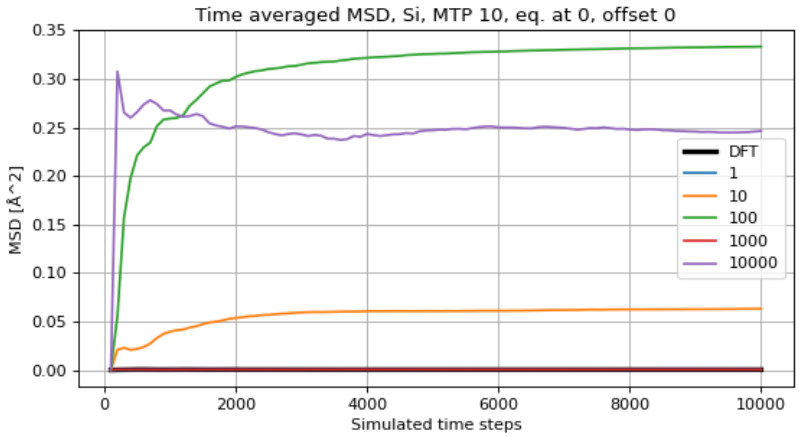



(c) Time averaged mean squared displacement for Silicon simulated with a 06 MTP trained on time steps starting from the 2000th time step, with offset 2000.

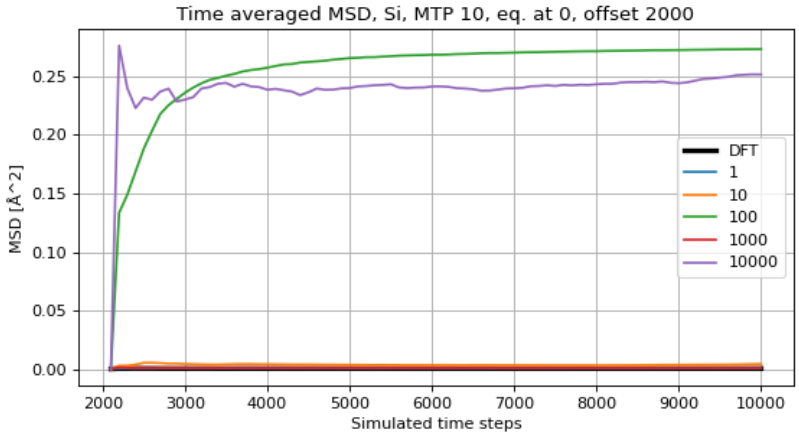
Figure 8.6: Instantaneous and time averaged mean squared displacement for Silicon simulated with a 06 MTP trained on time steps starting from the 2000th time step, with offset 0.




(a) Instantaneous mean squared displacement for Silicon simulated with a 10 MTP trained on time steps starting from the first time step, with offset 0.

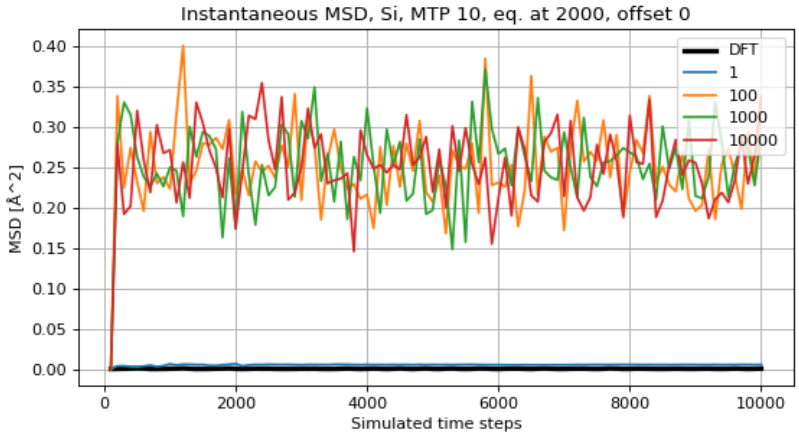



(b) Time averaged mean squared displacement for Silicon simulated with a 10 MTP trained on time steps starting from the first time step, with offset 0.

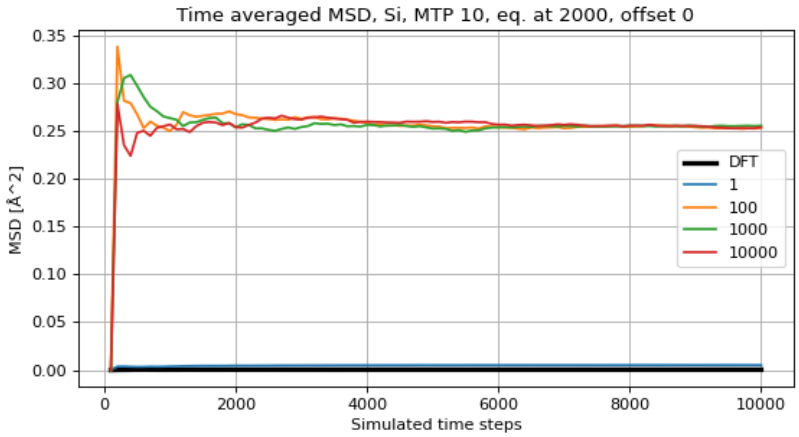



(c) Time averaged mean squared displacement for Silicon simulated with a 10 MTP trained on time steps starting from the first time step, with offset 2000.

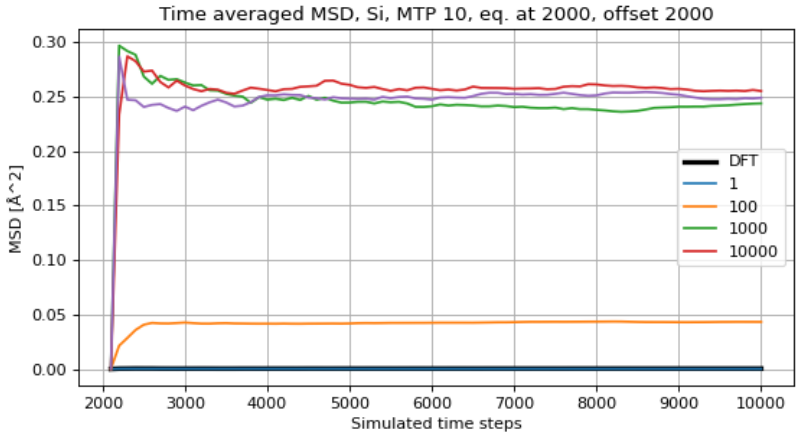
Figure 8.7: Instantaneous and time averaged mean squared displacement for Silicon simulated with a 10 MTP trained on time steps starting from the first time step, with offset 0 and 2000.




(a) Instantaneous mean squared displacement for Silicon simulated with a 10 MTP trained on time steps starting from the 2000th time step, with offset 0.




(b) Time averaged mean squared displacement for Silicon simulated with a 10 MTP trained on time steps starting from the 2000th time step, with offset 0.




(c) Time averaged mean squared displacement for Silicon simulated with a 10 MTP trained on time steps starting from the 2000th time step, with offset 2000.

Figure 8.8: Instantaneous and time averaged mean squared displacement for Silicon simulated with a 10 MTP trained on time steps starting from the 2000th time step, with offset 0 and 2000.

# 9 Discussion

When you change the way you look at things,
the things you look at change.

Max Planck

In this chapter the results and their implications are discussed.

## 9.1 Results

From the testing error plots for both Aluminium and Silicon in figs. 6.1 and 6.2 respectively, it is observed that the MTPs outperform the implementation of KRR rapidly. The MTPs reach MAEs in the order of magnitudes of $10^{-2}$ and $10^{-1}$ for Aluminium energy per atom and average force lengths, and $10^{-3}$ and $10^{-1}$ for Silicon energy per atom and average force lengths after only being trained on around 10 time steps. The KRR machines require a lot more training to reach comparably low errors. Training the KRR machines on all 10000 time steps is not enough to produce comparably low errors as the MTPs. Surprising however, is how much lower the error the of the energy per atom KRR machines are after only 1 training point, especially compared to the zero predictions baseline. This is not replicated for the KRR forces machines, where the machines trained on few time steps predictably perform as well as the zero predictions baseline. This is explained by the fact that predicting zero for the average force length i.e. no atomic moment is a more reasonable prediction than predicting zero for energy per atom, which is reasonably not zero. Because of the MTPs outperforming the KRR machines significantly, the rest of the work is focused on using MTPs in the MD simulations, which is reflected in the results chapters 6, 7 and 8.

The KRR machines that are implemented in this thesis are the most direct possible implementation, which most likely explains their inability to predict potential energy and forces to a sufficient degree. For example, no reduction of redundancy of the data is done, which causes the KRR machine to not learn the fact that all 32 Aluminium atoms are similar (equivalently for the 8 atoms in the Silicon data).

From the training vs. testing errors figs. 6.3, 6.4, 6.5 and 6.6 it is observed that for both Aluminium and Silicon, for both the 06 and 10 MTPs, for both energy per atom and average force length, 1000 time steps seems to be a suitable size for the training data, since the training and testing error curves meet with certainty at this point. Training for additional time steps does not significantly alter the the curves or improve the potentials. Since the 06 and 10 MTPs seem to perform equivalently, it indicates that the 06 MPT contains enough free parameters to learn the behaviour of Aluminium and Silicon to a degree that the additional free parameters in MTP 10 do not contribute in training.

One important observation to be made from the figures for the physical properties in chapters 7 and 8, is that training on time steps starting from 0 or 2000 does not seem to make a significant difference in the physical properties calculations. See for example the time averaged total energies for Aluminium, simulated with MTP 06 for eq. at 0 and eq. at 2000 in figs. 7.1(b) and 7.2(b). Regardless of whether or not the MTPs have been trained for eq. at 0 or 2000, all of the trained MTPs converge towards the DFT value similarly.

As previously mentioned in the section 8.1, the Silicon in their simulations most likely melt and reform into different crystal structures with different potential energies. This was investigated by looking at the equilibrium positions after the suspected melting and observing that simulations for each differently fitted MTP did have different equilibrium positions for the atoms. To further verify this, it would be suitable to visualise the simulated crystals and confirm visually that they really have different crystal structures. A look into the reasons for this was made, and preliminarily it was observed that the fitting of the MTPs affected which crystal structure the simulated Silicon crystal would reform to after melting. This should be looked into until some kind of solution for how to remedy this issue has been found, since it is a significant source of error affecting all the Silicon simulations.

It is observed that the MTPs seem to reproduce the total energies and MSDs for Aluminium, see figs. 7.1, 7.2, 7.3 and 7.4 for the total energies and figs. 7.5, 7.6, 7.7 and 7.8 for the MSDs, fairly well. All trained MTPs converge to the DFT value reasonably fast. However, no real distinction between the MTPs fitted for different amounts of time steps can be made considering the variance exhibited in fig. 7.13. Unfortunately, the calculated specific heat capacities do not correspond to the DFT simulation equally well, as seen in figs. 7.9, 7.10, 7.11 and 7.12. Considering the small training and testing errors shown for the MTPs in subsection 6.1, the question of how sufficiently small the energy and force errors need to be to accurately reproduce the specific heat is raised. Errors in the magnitude of meV for total energy predictions, as are presented for the simulations of Aluminium, are usually indicative of potentials being able to fairly replicate thermodynamic properties.

The variance of the simulated MTPs for Aluminium presented in fig. 7.13, is partly because of the random initialisation of MTP parameters in training but most likely mainly because of the effect of the thermostat during MD. As the Langevin thermostat tries to control the temperature by adjusting the kinetic energy accordingly, this would add to the variance. Another factor is the DFT data used has some particularities when using small supercells, a distinction which applies for the supercells of 32 for Aluminium and 8 for Silicon used in this project. For supercells of this size, the system may "lock into" certain phonon modes, in essence they get fixated into specific oscillatory states, which could possibly affect the limiting values. This should not explain the energy differences of multiple electron volts in the total energy for the Silicon simulations, but it could be a major contributing factor this this observed Aluminium variance.

As a final note regarding the simulated physical properties, it is also observed that it is more difficult to simulate the properties of Silicon than to simulate the properties of Aluminium. The directed bonding in the semiconductor Silicon might be more difficult for the MTPs to model and predict, than the simple metallic bonds in bulk Aluminium.

## 9.2 Method

In this section the methodology of this thesis project is discussed.

### Division of data

The more data that is used for training the potentials, the closer that training data comes to the test data in time, increasing dependence. To mitigate this, one could use a buffer as in the data divisions in figs. 5.3 and 5.4. Another possible method could be to use DFT data of

lower fidelity, i.e. longer time steps, to decrease the dependence between data points in the data. This other method was not attempted in this thesis work.

### MD simulations

Different kinds of integrators could have been investigated and compared. A small investigation into different integrators such as Velocity Verlet, Andersen, and Berendsen dynamics as available in the ASE package is done, but could be done much more extensively [30]. Langevin dynamics is chosen for the MD simulations in this work because of its simplicity compared to other equivalent thermostats.

## 9.3 The work in a wider context

As can be observed from the results and from related works, the ML methods for generating interatomic potentials seem promising in accelerating the field of data-driven materials design [17, 33, 37]. The implications of accelerating the materials design are significant; the discovery of new materials is useful in many technologies such as battery materials, catalysts, hard impact coatings, and many more [1].

MTPs have previously been shown to be able to reach the accuracy of computationally heavy quantum mechanical models, such as DFT, with lower execution time, especially using active learning approaches [37, 33]. Even though no active learning is implemented in this thesis project, the results are further indicative of the usefulness of MTPs in accelerating quantum mechanical simulations.

## 9.4 Future work

In the future, it would be interesting to look at other ways of implementing KRR machines. For example, instead of training three different machines for each force component when predicting the forces, another approach could be to train a single machine, say on the x axis, and rotate the cell for each additional axis. Also reducing the redundancy of the training data would likely yield lower errors.

Investigating the Silicon crystal melting and reformation which causes crystals of different potential energies would not only be highly interesting but also essential, with the goal of coming up with a way to remedy this issue. Also looking at how reasonable specific heat capacities for Aluminium could be yielded, for example by finding ways to further decrease the prediction errors of the potential energy and forces. This might be achieved by using MTPs with more parameters than the MTP 06 and 10 that are studied in this work. An investigation of how the DFT simulations' particular behaviour regarding getting fixated into specific oscillatory states for small supercells should be done. This could be done by producing several DFT data sets with unit cells of different sizes, and comparing how ML models trained on these different data sets perform. This would give an insight into how the effect of small unit cells affect the simulation of physical properties in this ML approach.

# 10 Conclusion

Insight must precede application.

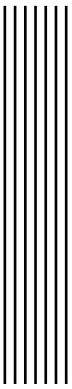
Max Planck

In summary, running MD with the trained MTPs has yielded total energies and MSDs for Aluminium that are comparable to those yielded by DFT, while at the same time requiring less CPU time. However, reasonable specific heat capacities have not been yielded and the question of how further accurate the MTPs need to be for this has been raised. The Silicon simulations have not yielded reasonable physical properties with the current implementation, which causes an undesired melting and reforming of the Silicon crystal during simulation. These simulations require more consideration, and are an testament for how simulating a semiconductor like Silicon is more challenging than simulating a simple metal like Aluminium.

Relating back to the research questions, to answer the first question, it has been shown that the accuracy achieved using interatomic potentials obtained from ML models in MD simulations achieve quite high accuracy. As has been observed from the results chapter 6 the MTPs achieve errors in the orders of magnitudes for the energy per atom and force predictions of $10^{-3}$ and $10^{-1}$ respectively for Aluminium, and $10^{-3}$ and $10^{-2}$ respectively for Silicon, after only being trained on 100 time steps starting from the first time step. For complete certainty that the MTPs have learned what they can learn they should be trained on 1000 time steps. The KRR implemented in this thesis has not yielded equivalently small errors as the MTPs.

To answer the second question, as has been observed from the results sections 7.1 and 8.1, the MTPs achieve comparable accuracy to DFT simulations for the properties total energy and MSD for Aluminium, with errors in the orders of magnitudes of meV and $10^{-5}$ Å$^2$ respectively. The specific heat capacity has not been reasonably predicted for Aluminium. None of the studied properties have been reasonably predicted for Silicon, which has been discussed in section 9.1. To answer the third question, as has been observed from the results sections 7.1 and 8.1, the current implementation of MTPs has been shown to simulate the physical properties of the simple metal Aluminium more accurately than the properties of the semiconductor Silicon.

In conclusion, with some more additional work, especially for the Silicon simulations, MTPs could very likely be able to accelerate the speed which high-throughput calculations of the materials properties investigated in this thesis can be made, accelerating the discovery of new materials and the paradigm of data-driven materials design in general. In appendix A, a recommended workflow for how to practically work with MTPs is presented.



# Bibliography


[1] Rickard Armiento. "Database-Driven High-Throughput Calculations and Machine Learning Models for Materials Design". In: *Lecture Notes in Physics* (2020), pp. 377–395. ISSN: 1616-6361. DOI: 10.1007/978-3-030-40245-7_17. URL: http://dx.doi.org/10.1007/978-3-030-40245-7_17.

[2] Jörg Behler. "Atom-centered symmetry functions for constructing high-dimensional neural network potentials". In: *The Journal of Chemical Physics* 134.7 (2011), p. 074106. DOI: 10.1063/1.3553717. eprint: https://doi.org/10.1063/1.3553717. URL: https://doi.org/10.1063/1.3553717.

[3] Jörg Behler. "Perspective: Machine learning potentials for atomistic simulations". In: *The Journal of Chemical Physics* 145.17 (2016), p. 170901. DOI: 10.1063/1.4966192. eprint: https://doi.org/10.1063/1.4966192. URL: https://doi.org/10.1063/1.4966192.

[4] Christopher Bishop. *Pattern Recognition and Machine Learning*. Springer, Jan. 2006. URL: https://www.microsoft.com/en-us/research/publication/pattern-recognition-machine-learning/.

[5] Luis J. Boya. "The Thermal Radiation Formula of Planck". In: *Rev. Academia de Ciencias, Zaragoza* 58 (2003), pp. 91–114. URL: https://arxiv.org/pdf/physics/0402064.pdf.

[6] Stefan Chmiela, Alexandre Tkatchenko, Huziel E. Sauceda, Igor Poltavsky, Kristof T. Schütt, and Klaus-Robert Müller. "Machine learning of accurate energy-conserving molecular force fields". In: *Science Advances* 3.5 (2017). DOI: 10.1126/sciadv.1603015. eprint: https://advances.sciencemag.org/content/3/5/e1603015.full.pdf. URL: https://advances.sciencemag.org/content/3/5/e1603015.

[7] Anders S. Christensen, Felix A. Faber, Bing Huang, Lars A. Bratholm, Alexandre Tkatchenko, Klaus-Robert Muller, and O. Anatole von Lilienfeld. *QML: A Python Toolkit for Quantum Machine Learning*. 2017.

[8] Stefano Curtarolo, Gus L. W. Hart, Marco Buongiorno Nardelli, Natalio Mingo, Stefano Sanvito, and Ohad Levy. "The high-throughput highway to computational materials design". In: *Nature Materials* 12.3 (Mar. 2013), pp. 191–201. ISSN: 1476-4660. DOI: 10.1038/nmat3568. URL: https://doi.org/10.1038/nmat3568.

[9] Sandip De, Albert P. Bartók, Gábor Csányi, and Michele Ceriotti. "Comparing molecules and solids across structural and alchemical space". In: *Phys. Chem. Chem. Phys.* 18 (20 2016), pp. 13754–13769. DOI: 10.1039/C6CP00415F. URL: http://dx.doi.org/10.1039/C6CP00415F.

[10] DeepMind. *AlphaGo*. https://deepmind.com/research/case-studies/alphago-the-story-so-far. [Online; accessed 2021-04-23]. 2021.

[11] "Baszoetekouw at en.wikipedia". 2006. URL: https://commons.wikimedia.org/w/index.php?curid=3512062,%20https://commons.wikimedia.org/w/index.php?curid=3512101,%20https://commons.wikimedia.org/w/index.php?curid=3512107.

[12] Felix Faber, Alexander Lindmaa, O. Anatole von Lilienfeld, and Rickard Armiento. "Crystal structure representations for machine learning models of formation energies". In: *International Journal of Quantum Chemistry* 115.16 (2015), pp. 1094–1101. DOI: https://doi.org/10.1002/qua.24917. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/qua.24917. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/qua.24917.

[13] Jan Faye. "Copenhagen Interpretation of Quantum Mechanics". In: *The Stanford Encyclopedia of Philosophy*. Ed. by Edward N. Zalta. Winter 2019. Metaphysics Research Lab, Stanford University, 2019.

[14] Daan Frenkel and Berend Smit. *Understanding Molecular Simulation*. Academic Press, 2001. ISBN: 9780122673511.

[15] Davide Gambino and Johan Klarbring. *Private communication*. 2021.

[16] Xiang Gao, Rong Tan, and Guanghui Li. "Research on Text Mining of Material Science Based on Natural Language Processing". In: *IOP Conference Series: Materials Science and Engineering* 768 (Mar. 2020), p. 072094. DOI: 10.1088/1757-899x/768/7/072094. URL: https://doi.org/10.1088/1757-899x/768/7/072094.

[17] Konstantin Gubaev, Evgeny V. Podryabinkin, Gus L.W. Hart, and Alexander V. Shapeev. "Accelerating high-throughput searches for new alloys with active learning of interatomic potentials". In: *Computational Materials Science* 156 (2019), pp. 148–156. ISSN: 0927-0256. DOI: https://doi.org/10.1016/j.commatsci.2018.09.031. URL: https://www.sciencedirect.com/science/article/pii/S0927025618306372.

[18] Lauri Himanen, Marc O.J. Jäger, Eiaki V. Morooka, Filippo Federici Canova, Yashasvi S. Ranawat, David Z. Gao, Patrick Rinke, and Adam S. Foster. "DScribe: Library of descriptors for machine learning in materials science". In: *Computer Physics Communications* 247 (2020), p. 106949. ISSN: 0010-4655. DOI: https://doi.org/10.1016/j.cpc.2019.106949. URL: https://www.sciencedirect.com/science/article/pii/S0010465519303042.

[19] Bing Huang and O. Anatole von Lilienfeld. "Communication: Understanding molecular representations in machine learning: The role of uniqueness and target similarity". In: *The Journal of Chemical Physics* 145.16 (2016), p. 161102. DOI: 10.1063/1.4964627. eprint: https://doi.org/10.1063/1.4964627. URL: https://doi.org/10.1063/1.4964627.

[20] Haoyan Huo and Matthias Rupp. *Unified Representation of Molecules and Crystals for Machine Learning*. 2018. arXiv: 1704.06439 [physics.chem-ph].

[21] Giulio Imbalzano, Andrea Anelli, Daniele Giofré, Sinja Klees, Jörg Behler, and Michele Ceriotti. "Automatic selection of atomic fingerprints and reference configurations for machine-learning potentials". In: *The Journal of Chemical Physics* 148.24 (2018), p. 241730. DOI: 10.1063/1.5024611. eprint: https://doi.org/10.1063/1.5024611. URL: https://doi.org/10.1063/1.5024611.

[22] Marc O. J. Jäger, Eiaki V. Morooka, Filippo Federici Canova, Lauri Himanen, and Adam S. Foster. "Machine learning hydrogen adsorption on nanoclusters through structural descriptors". In: *npj Computational Materials* 4.1 (July 2018), p. 37. ISSN: 2057-3960. DOI: 10.1038/s41524-018-0096-5. URL: https://doi.org/10.1038/s41524-018-0096-5.

[23] Claes Johnson. *Mathematical Physics of Blackbody Radiation*. 2012. URL: https://www.csc.kth.se/~cgjoh/ambsblack.pdf.

[24] Mohammad Ali Kadampur and Sulaiman Al Riyaee. "Skin cancer detection: Applying a deep learning based model driven architecture in the cloud for classifying dermal cell images". In: *Informatics in Medicine Unlocked* 18 (2020), p. 100282. ISSN: 2352-9148. DOI: https://doi.org/10.1016/j.imu.2019.100282. URL: https://www.sciencedirect.com/science/article/pii/S2352914819302047.

[25] Charles Kittel. *Introduction to Solid State Physics*. 8th ed. Wiley, 2004. ISBN: 9780471415268.

[26] W. Kohn. "Nobel Lecture: Electronic structure of matter—wave functions and density functionals". In: *Rev. Mod. Phys.* 71 (5 Oct. 1999), pp. 1253–1266. DOI: 10.1103/RevModPhys.71.1253. URL: https://link.aps.org/doi/10.1103/RevModPhys.71.1253.

[27] W. Kohn and L. J. Sham. "Self-Consistent Equations Including Exchange and Correlation Effects". In: *Phys. Rev.* 140 (4A Nov. 1965), A1133–A1138. DOI: 10.1103/PhysRev.140.A1133. URL: https://link.aps.org/doi/10.1103/PhysRev.140.A1133.

[28] Walter Kohn and Ann E. Mattsson. "Edge Electron Gas". In: *Phys. Rev. Lett.* 81 (16 Oct. 1998), pp. 3487–3490. DOI: 10.1103/PhysRevLett.81.3487. URL: https://link.aps.org/doi/10.1103/PhysRevLett.81.3487.

[29] Pierre Simon Laplace. *A philosophical essay on probabilities*. J. Wiley Sons, 1902.

[30] Ask Hjorth Larsen, Jens Jørgen Mortensen, Jakob Blomqvist, Ivano E Castelli, Rune Christensen, Marcin Dułak, Jesper Friis, Michael N Groves, Bjørk Hammer, Cory Hargus, Eric D Hermes, Paul C Jennings, Peter Bjerre Jensen, James Kermode, John R Kitchin, Esben Leonhard Kolsbjerg, Joseph Kubal, Kristen Kaasbjerg, Steen Lysgaard, Jón Bergmann Maronsson, Tristan Maxson, Thomas Olsen, Lars Pastewka, Andrew Peterson, Carsten Rostgaard, Jakob Schiøtz, Ole Schütt, Mikkel Strange, Kristian S Thygesen, Tejs Vegge, Lasse Vilhelmsen, Michael Walter, Zhenhua Zeng, and Karsten W Jacobsen. "The atomic simulation environment—a Python library for working with atoms". In: *Journal of Physics: Condensed Matter* 29.27 (June 2017), p. 273002. DOI: 10.1088/1361-648x/aa680e. URL: https://doi.org/10.1088/1361-648x/aa680e.

[31] Dirac Paul Adrien Maurice. "Quantum mechanics of many-electron systems". In: (1929). DOI: https://doi.org/10.1098/rspa.1929.0094.

[32] Ivan S. Novikov, Konstantin Gubaev, Evgeny V. Podryabinkin, and Alexander V. Shapeev. *The MLIP package: Moment Tensor Potentials with MPI and Active Learning*. 2020. arXiv: 2007.08555 [physics.comp-ph].

[33] I.I. Novoselov, A.V. Yanilkin, A.V. Shapeev, and E.V. Podryabinkin. "Moment tensor potentials as a promising tool to study diffusion processes". In: *Computational Materials Science* 164 (2019), pp. 46–56. ISSN: 0927-0256. DOI: https://doi.org/10.1016/j.commatsci.2019.03.049. URL: https://www.sciencedirect.com/science/article/pii/S0927025619301843.

[34] OpenAI. *OpenAI Five*. https://openai.com/projects/five/. [Online; accessed 2021-04-23]. 2021.

[35] John P. Perdew. "Density-functional approximation for the correlation energy of the inhomogeneous electron gas". In: *Phys. Rev. B* 33 (12 June 1986), pp. 8822–8824. DOI: 10.1103/PhysRevB.33.8822. URL: https://link.aps.org/doi/10.1103/PhysRevB.33.8822.

[36] S. Plimpton. "Fast Parallel Algorithms for Short-Range Molecular Dynamics". In: *J Comp Phys* 117 (1995), pp. 1–19. URL: https://cs.sandia.gov/~sjplimp/papers/jcompphys95.pdf.

[37] Evgeny V. Podryabinkin and Alexander V. Shapeev. "Active learning of linearly parametrized interatomic potentials". In: *Computational Materials Science* 140 (Dec. 2017), pp. 171–180. ISSN: 0927-0256. DOI: 10.1016/j.commatsci.2017.08.031. URL: http://dx.doi.org/10.1016/j.commatsci.2017.08.031.

[38] Matthias Rupp, Alexandre Tkatchenko, Klaus-Robert Müller, and O. Anatole von Lilienfeld. "Fast and Accurate Modeling of Molecular Atomization Energies with Machine Learning". In: *Phys. Rev. Lett.* 108 (5 Jan. 2012), p. 058301. DOI: 10.1103/PhysRevLett.108.058301. URL: https://link.aps.org/doi/10.1103/PhysRevLett.108.058301.

[39] Alexander Shapeev. "Moment Tensor Potentials: A Class of Systematically Improvable Interatomic Potentials". In: *Multiscale Modeling Simulation* 14 (Dec. 2015). DOI: 10.1137/15M1054183.

[40] Simon Stephan, Monika Thol, Jadran Vrabec, and Hans Hasse. "Thermophysical Properties of the Lennard-Jones Fluid: Database and Data Assessment". In: *Journal of Chemical Information and Modeling* 59.10 (Oct. 2019), pp. 4248–4265. ISSN: 1549-9596. DOI: 10.1021/acs.jcim.9b00620. URL: https://doi.org/10.1021/acs.jcim.9b00620.

[41] *The Vienna Ab Initio Simulation Package: atomic scale materials modelling from first principles*. 2021. URL: https://www.vasp.at/.

# A Appendix: Recommended workflow

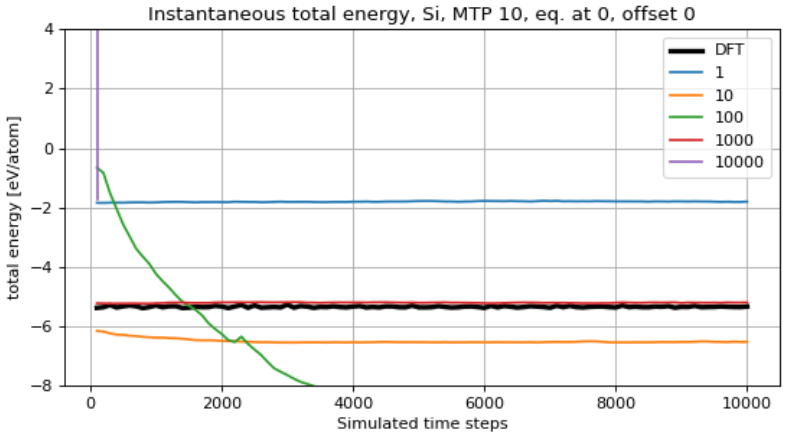
The recommended workflow for working with MTPs is, of course, to first determine the training data to be used, for example DFT data generated with VASP as in this project. The MLIP2 package has a function to convert VASP OUTCAR files to MTP appropriate .cfg-files. The repository for this master thesis project has functions to generate training and testing data from the MLIP2 converted .cfg-file[1]. Initially, train some untrained MTPs of different orders, calculate the MAEs of say the energy per atom using the in-built functions, and plot them against the amount of time steps[2]. Consider how small of an error is acceptable for the intended simulations, which depends on the purposes of the simulations. In this work the results reflect the amount of time steps needed to simulate and calculate physical properties that are comparable to DFT calculations, which may serve as a benchmark of sorts. If it is desired to accelerate properties calculations a starting point could be to train the 06 MTP for 100 time steps and compare them to previous simulations, if such are available. Important to note is that the potentials should be trained on time steps after equilibrium has been reached, to be certain that no nonphysical behaviour is learned. If acceptable accuracy is reached while doing a pilot simulation using less time than previous simulation methods attempted, it might be desirable to continue doing all calculations in high-throughput using this less expensive potential. Otherwise, if the accuracy is not acceptable training the potential for more time steps or choosing a MTP with more free parameters would be a suitable course of action.

[1] https://github.com/obsqyr/master-thesis/blob/main/MTP/cfg_parser.py
[2] https://github.com/obsqyr/master-thesis/blob/main/MTP/visualization.py

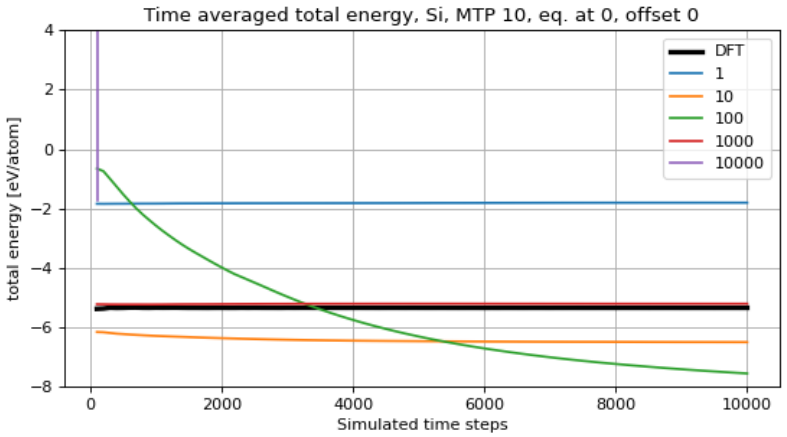
# B Appendix: Anomalies

Something to be aware of when running MD using ASE with the MTP calculators is that they need to be initially monitored to make sure that hyperparameters are set appropriately. Inappropriately chosen hyperparameters can result in diverging simulations, such as the simulations for the MTPs trained on 1 and 100 time steps in fig. 7.2. Before running MD using MTPs in high-throughput the user needs to be confident in that the choice of hyperparameters are valid.

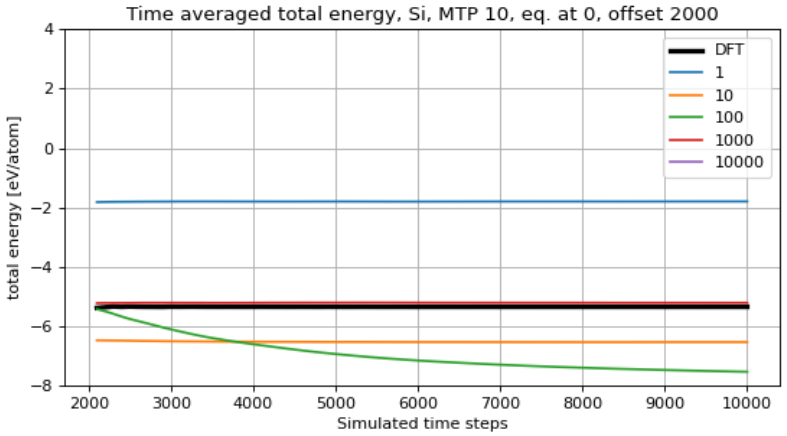
In fig. B.1 the MTP trained on 100 time steps exhibits an anomalous behaviour, it does not converge to any particular value for any of the investigated physical properties. This further illuminates the sometimes seemingly random behaviour of the MTPs during simulation.




(a) Instantaneous total energy for Silicon simulated with a 10 MTP trained on time steps starting from the first time step, with offset 0.




(b) Time averaged total energy for Silicon simulated with a 10 MTP trained on time steps starting from the first time step, with offset 0.




(c) Time averaged total energy for Silicon simulated with a 10 MTP trained on time steps starting from the first time step, with offset 2000.

Figure B.1: Instantaneous and time averaged total energy for Silicon simulated with a 10 MTP trained on time steps starting from the first time step, with offset 0 and 2000.