Bachelor Thesis

Machine Learning for Quantum Mechanical Problems

submitted in satisfaction of the requirements for the degree of
Bachelor of Science (BSc.)
of the TU Wien, Faculty of Physics

Bachelorarbeit

Maschinelles Lernen zur Lösung von Quanten-Mechanischen Problemen

ausgeführt zum Zwecke der Erlangung des akademischen Grades eines
Bachelor of Science (BSc.)
eingereicht an der Technischen Universität Wien, Fakultät für Physik

von

Simon König

Matr.Nr.: 01234567

unter der Anleitung von

Univ.Prof. Mag.rer.nat. Dr.rer.nat. **Andreas Grüneis**

Institut für Theoretische Physik
Forschungsbereich
Technische Universität Wien
Wiedner Hauptstraße 8, 1040 Wien, Österreich

Wien, im Mai 2021

Abstract

The thesis centres around the analysis of quantum-mechanical environments. Atomic energies and forces are determined for given atomic configurations using machine learning (ML) algorithms. In physics the computation of atomic energies and forces is an important field. This work focuses on building a model, based on ML methods, able to predict energies and forces for given atomic systems. In order to use machine learning to solve problems, it is necessary to realise that these methods do not perform any complex computation describing the atomic environment. A ML model needs representative groundwork, so called, training data mirroring the system of interest. The model learns on training data and is then able to make predictions on systems similar to the ones it has been trained on.

This work first discusses the computation of training data as foundation for the machine learning algorithm. In this case the training data consists of value pairs. One value pair maps an atomic configuration to its features, the atomic energies and forces. Training data is generated using molecular dynamics (MD) simulations, making use of density functional theory. The following part explains the machine learning algorithm and its specifications. The ML model processes information and yields predictions based on the Gaussian regression process algorithm, a stochastic mechanic centred around Gaussian probability distributions. The model is then created using the quantum mechanically computed training data as foundation and a Gaussian process algorithm as mechanic.

To optimise the prediction made by the model and thus improve resemblance of reality, the construction of the model is plugged into an error minimising feedback loop. This circular optimisation process tunes the model parameters to optimise results by reduction of the offset between model predictions and reality. The refined results are presented and visualised in line with referring to future possibilities of ML mechanics describing atomic systems.

Contents

1 Introduction **6**

2 Theoretical Basis **7**
2.1 Many Body Problem of Quantum Theory . . . 7
2.1.1 Born Oppenheimer Approximation . . . 8
2.1.2 Hartree approximation . . . 9
2.2 Density Functional Theory . . . 10
2.2.1 Motivation . . . 10
2.2.2 Energy as Functional of Density . . . 10
2.3 Solutions for DFT . . . 11

3 Machine Learning **13**
3.1 Motivation . . . 13
3.1.1 From Density Functional Theory to Machine Learning . . . 13
3.1.2 Relationship of Atomic Systems and their Energy . . . 13
3.2 Gaussian Process Regression . . . 14
3.2.1 Prerequisites . . . 14
3.2.2 Building the Kernel . . . 15
3.2.3 Making Predictions . . . 15
3.3 Modifications of Gauss Process Regression for Quantum Mechanical use . . . 16
3.3.1 Total Energies and Forces . . . 16
3.3.2 Local Approximation - Cutoff Radius . . . 16
3.3.3 Sparsification - Representative Data . . . 17
3.3.4 Descriptors - Representing the Atomic Neighbourhood Environment . . . 18

4 Machine Learning Model Creation **19**
4.1 Observed Systems . . . 19
4.1.1 Water Molecule H2 . . . 19
4.1.2 Water Crystal . . . 19
4.2 Workflow . . . 20
4.2.1 Flow Chart . . . 20
4.2.2 Get Data from Molecular Dynamics Simulation . . . 20
4.2.3 Split Data into Training and Validation Data . . . 21
4.2.4 Model Training . . . 21
4.2.5 Model Prediction . . . 24
4.2.6 Model Optimisation . . . 24
4.2.7 Water Crystal Model Creation . . . 26

5 Outcome and Analysis **27**
5.1 Water Molecule H2 . . . 27
5.1.1 Outcome Visualisation . . . 27
5.1.2 Model Parameter Analysis . . . 28

5.2 Hydrogen Crystal 30
5.2.1 Outcome Visualisation 30
5.2.2 Model Parameter Analysis 31

6 Summary and Outlook **33**
6.1 Model Improvement by Descriptor Adjustment 33
6.2 Larger Systems 33

Appendix **35**
A Preparation of Training Data 35
B Software 35
C Code 35
D Optimisation 35

Bibliography **36**

Chapter 1

Introduction

One of the main challenges in modern physics is the modelling of atomic environments and the determination of related quantum mechanical features of interest, for example atomic energies and forces. These microscopic properties determine macroscopic material properties, like thermo- and electro-conductivity or elasticity. The Computation of quantum-mechanical environments is conducted using approximations, like the Born & Oppenheimer approximation, and self consistent methods. In spite of these simplification, quantum-mechanical practices describing atomic environments, in this work the density functional theory (DFT), are not suited for large systems because of high computational cost. Quantum theories, like the DFT, return correct results but lead to major computational cost.

This thesis deals with reducing the computational cost. The centre point for minimising runtime of analysis of atomic features is avoiding quantum-mechanical calculations and introducing a machine learning (ML) model to make predications instead of computations. In general a ML model takes a set of expensively computed training data as input and is then able to predict quantum-mechanical properties based on the previously gained knowledge. In this thesis the practice of earning and predicting is examined on the example atomic systems of a Hydrogen molecule H_2 and a Hydrogen crystal containing 192 atoms.

Chapter 2

Theoretical Basis

2.1 Many Body Problem of Quantum Theory

Solving the Schrödinger equation is one of the fundamental problems of modern physics. The Schrödinger equation is defined as eigen value problem, demanding a set of wave functions as solution. Applying the Hamiltonian on a set of eigen functions yields a corresponding energy.

$$\hat{H}\left|\Psi\right\rangle = E\left|\Psi\right\rangle \tag{2.1}$$

- $\hat{H}$ ... Hamiltonian
- $\left|\Psi\right\rangle$ ... wave function in Dirac notation, eigenfunction of the Hamiltonian
- E ... Energy of the system, eigen value of the Schrödinger equation

The Hamiltonian is the quantum mechanical analogy to the classical concept of mechanical energy. Equation 2.2 states that the Hamiltonian is the sum of the kinetic energy operator $\hat{T}$ and the potential energy operator $\hat{V}$.

$$\hat{H} = \hat{T} + \hat{V} \tag{2.2}$$

Solving the eigen value problem analytically is possible for a small number of objects, for example the Hydrogen atom. Real problems correspond to large systems, containing a large amount of particles. Numerical methods, such as the density functional theory (DFT) used in this work, are applied.

In atomic units, a system of N_n nuclei and N_e electrons is described by the many body Schrödinger Hamiltonian in [Lecture 4 1, p. 2]:

$$\hat{H} = -\frac{1}{2}\sum_{i}^{N_e}\nabla_i^2 - \frac{1}{2}\sum_{I}^{N_n}\nabla_I^2 - \sum_{i}^{N_e}\sum_{I}^{N_n}\frac{Z_I}{|\vec{r_i} - \vec{R_I}|} + \frac{1}{2}\sum_{i}^{N_e}\sum_{i\neq j}^{N_e}\frac{1}{|\vec{r_i} - \vec{r_j}|} + \frac{1}{2}\sum_{I}^{N_n}\sum_{I\neq J}^{N_n}\frac{Z_I Z_J}{|\vec{R_i} - \vec{R_j}|} \tag{2.3}$$

- $\hat{H}$ ... Hamiltonian
- r_i, r_j ... position of i_{th}, j_{th} electron
- R_I, R_I ... position of J_{th}, J_{th} nucleus
- Z_I, Z_J ... charge of I_{th}, J_{th} nucleus

As precursors to the DFT two approximations are discussed int the following. The ideas of the presented methods offer an approach to finding a numerical solution to the many body Schrödinger equation in equation 2.1.1 and lay the foundation for DFT.

The necessary approximations are:

- Born Oppenheimer approximation
- Hartree approximation

2.1.1 Born Oppenheimer Approximation

Born and Oppenheimer suggest, because of the significant mass difference of electron and nucleus to neglect the movement of the nucleus in respect to the electrons. The underlying idea is that the electrons, because of their mass being lower than that of the nucleus by a magnitude of 10^3, relax to an equilibrium state rapidly for any movement of the nucleus. Any movement of the nucleus does not affect the energy of the electrons as it is assumed the e^- move to equilibrium state instantly. Details to the idea of Born and Oppenheimer are to be found in [2, p. 272].

The kinetic energy operator $\hat{T}$ of a center of momentum frame with one nucleus and N electrons is given. Mathematically the idea can be expressed by performing the limit $M \to \infty$.

$$\hat{T} = -\frac{\hbar^2}{2(M+Nm)}\nabla^2_{CM} - \frac{\hbar}{2\mu}\sum_i^N \nabla_i^2 - \frac{\hbar^2}{M}\sum_{i>j}^N \nabla_i \nabla_j \tag{2.4}$$

Equation 2.4 is reduced following the Born Oppenheimer approximation. Mathematically the idea can be expressed by performing the limit M $\to \infty$, yielding the electronic kinetic energy operator $\hat{T}_e$.

$$\hat{T}_e = -\frac{\hbar}{2m}\sum_i^N \nabla_i^2 \tag{2.5}$$

- $\hat{T}$ ... Kinetic energy operator
- $\hat{T}_e$ ... Electronic kinetic energy operator
- M ... Mass of nucleus
- m ... Mass of electrons
- N ... Number of electrons
- ∇_{CM} ... Nabla operator with respect to the centre of mass
- ∇_i, ∇_j Nabla operator with respect to the i_{th}, j_{th} electron
- $\mu = \frac{Mm}{M+m}$ ... Reduced mass

The concept is introduced for a system of one nucleus and N electrons and is then expanded to any arbitrary Hamiltonian describing N_n nuclei and N_e electrons in equation 2.1.1. The Born Oppenheimer approximation is one of the main ideas building and instrumental to the density functional theory. The electronic Born Oppenheimer Hamiltonian is given.

$$\hat{H}_e = -\frac{1}{2}\sum_{i}^{N_e}\nabla_i^2 - \sum_{i}^{N_e}\sum_{I}^{N_n}\frac{Z_I}{|\vec{r_i}-\vec{R_I}|} + \frac{1}{2}\sum_{i}^{N_e}\sum_{i\neq j}^{N_e}\frac{1}{|\vec{r_i}-\vec{r_j}|} \tag{2.6}$$

The electronic Hamiltonian in equation 2.6 is the electronic part of the many body general Hamiltonian in equation .

2.1.2 Hartree approximation

Hartree makes an ansatz for the many electron wave function. The last term, describing the nucleus-nucleus interaction, of the many body Hamiltonian in equation 2.1.1 is assumed to be constant with the Born Oppenheimer approximation. The electronic many body Hamiltonian reads:

$$\hat{H}_e = -\frac{1}{2}\sum_{i}^{N_e}\nabla_i^2 - \sum_{i}^{N_e}\sum_{I}^{N_n}\frac{Z_I}{|\vec{r_i}-\vec{R_I}|} + \frac{1}{2}\sum_{i}^{N_e}\sum_{i\neq j}^{N_e}\frac{1}{|\vec{r_i}-\vec{r_j}|}. \tag{2.7}$$

The first two terms of equation 2.1.1 are trivial since they are equivalent to the solution of the one body Schrödinger equation. Solving the eigenvalue problem for the first and the second term leads to a separable differential equation. The first term in equation 2.1.1 is the Kinetic Energy $\hat{T}_e$ for the i_{th} electron and the second the term is the potential energy operator $\hat{V}_{en}$ for the i_{th} electron in respect to the I_{th} nucleus. The solution of the eigen functions is a product of wave functions, solving the one body Schrödinger equation. [1, p. 4]

The third term represents the potential energy $\hat{V}_e e$ and the denominator term entangles the coordinates of the i_{th} and the j_{th} electron.

$$\hat{V}_{ee} = \frac{1}{2}\sum_{i}^{N_e}\sum_{i\neq j}^{N_e}\frac{1}{|\vec{r_i}-\vec{r_j}|} \tag{2.8}$$

A possible initial eigen function to the electronic Hamiltonian in equation 2.7 is proposed by Hartee. An ansatz to initialise a self consistent solving mechanism is a product of all the N single electron wave functions. [3, p. 78]

$$\psi(\vec{r}_1, ..., \vec{r}_N) = \phi_1(\vec{r}_1)\psi_2(\vec{r}_2)...\psi_N(\vec{r}_N) \tag{2.9}$$

The potential energy operator in equation 2.8 is not separable, this is where Hartree's approximation applies. The main idea is that the potential applying to any electron is identical for every electron of the system and is caused by all the other electrons and nuclei.

With the separated eigen function $\psi(\vec{r}_1, ..., \vec{r}_N)$ an ansatz for V_{ee} is made in equation 2.10 with the potential being proportional to the proximity density of the wave functions ϕ_i. This is called the mean field approximation.

$$\hat{V}_{ee}(\vec{r}) = \sum_{i}^{N_e}\int\frac{|\phi_i|^2}{|\vec{r}-\vec{r'}|}\,dr' \tag{2.10}$$

This approximation leads to a problem in computing the potential, thus also the Hamiltonian of the one body Schrödinger equation. The eigenvalue functions to the potential energy operator $\hat{V}$ in 2.10 are part of the operator. A self consistent solving mechanism needs to be applied. [3, p. 77]

The same problem occurs in density functional theory and is solved by using a self consistent method. This means looping through an algorithm by making an educated guess for the eigen functions then computing the potential and using the potential to compute new eigen functions.

1. Ansatz for eigen functions
2. Potentials are determined by computing the probability density of the eigen functions: $|\phi_i|^2$
3. New eigenfunctions are computed with the given potential

The algorithm is an iterative process and is repeated until a self consistent solution is found.

2.2 Density Functional Theory

2.2.1 Motivation

The solution of the many Body Schrödinger equation has become feasible after applying Hartree's and Born's & Oppenheimer's approximations. With Hartree the eigenfunctions of the N body problem separate into a product of N orbitals for every spacial coordinate. One electron equals the observation of the 3 cartesian coordinates, N electrons equal the observation of $3N$ coordinates. The Hartree method provides a reduction in coordinates compared to 3^N orbitals necessary for solving the non-approximated Schrödinger equation. For real many body problems the computation in $3N$ coordinates is expensive. The computational cost of computing solutions of the Schrödinger equation with Hartree's approximation scales with N^3. This is due to matrix regularisation as the limiting factor. An CO_2 Molecule for instance, consists of 22 electrons, the related Schrödinger equation is a 66 dimensional problem.

$$CO_2 \cong 22 \text{ electrons} \tag{2.11}$$

The configuration of a lead unit cell contains about 100 Atoms. A single Pb atom possesses 14 electrons, which counts up to 1400 electrons per unit cell. The eigenvalue problem has to be solved for 4200 dimensions.

$$Pb_{\text{unitcell}} \cong 1400 \text{ electrons} \tag{2.12}$$

Density functional theory reduces computational time significantly and the many body Schrödinger equation of the lead unit cell can be solved faster than with quantum chemistry methods, like the Møller–Plesset perturbation theory. Informations on this are derived from [4].

2.2.2 Energy as Functional of Density

In the article, related to the honour of his nobel price achievement, Walter Kohn writes:

"The basic lemma of HK. The ground state density n(r) of a bound system of interacting electrons in some external potential v(r) determines this potential uniquely (Hohenberg and Kohn, 1964)" [5, p. 7]

This Lemma is the substantial concept underlying density functional theory. Kohn proposes the ground state energy of any quantum system depends solely on the electron density $n(r)$. For any electron configuration the many body Schrödinger equation is a 3 dimensional problem. The electronic Hamiltonian in Born Oppenheimer approximation is given, where $\hat{V}_{ext}$ follows comparing 2.13 with 2.7. [6, p. 6]

$$\hat{H} = \hat{T} + \hat{V}_{ext} + \hat{V}_{ee} \tag{2.13}$$

- $\hat{T}$ ... Kinetic energy operator of the free electrons
- $\hat{V}_{ext}$... External potential energy operator as functional of the electron density $\vec{n}(r)$,
- $\hat{V}_{ee}$ ... Potential energy operator describing the electron-electron-interaction

The ideas of DFT proposed by Kohn, Hohenberg & Sham in 1964 rest on two fundamental theorems:

1. $\hat{V}_{ext}$ is, apart from a constant term, determined uniquely as a functional of the electron density of the system: $\vec{n}(r)$
2. The functional V_{ext} has its Minimum in respect to a variation of the electron density $\delta\vec{n}(r)$ for a density $\vec{n}_0(r)$ for a given external potential V_{ext}.

2.3 Solutions for DFT

A general expression for the energy based on the theorems by Kohn, Hohenberg & Sham as denoted by [6, p. 6].

$$E_0 = E_V(n_0) = T_s[n_0] + \int V_{ext}(r)[n_0](r)dr + J[n_0] + E_{XC}[n_0] \tag{2.14}$$

uses the terms:

- E_0 ... Ground state energy
- n_0 ...Ground state electronic density
- $T_s[n]$ ...Kinetic energy of non-interacting electrons
- $\hat{V}_{ext}[n_0]$ ... External potential
- $J[n_0]$ ... Classical Coulomb energy
- $E_{XC}[n_0]$ ... Exchange Correlation Energy

The energy is functional of the electron density $n(r)$, hence the electron density $n(r)$ of a system has to be determined. Having Hartree in mind, the determination of the electron density $n(r)$ and further the ground state energy requires a self consistent method. Hartree's idea is applied by DFT and the electron density is determined by the sum over the density of the probability of presence:

$$n(r) = \sum_i^N |\phi_i(r)|^2 \tag{2.15}$$

The Kohn Sham equations are introduced to determine the eigen functions $\phi_r(r)$ necessary for the computation of the electron density $n(r)$. DFT introduces the concept of the Exchange correlation functional $E_{XC}[n_0]$, approximated by a functional of density. Details are in [7, p. 161].

$$\left[T_s + V(r) + \int \frac{n_0(r')}{|r - r'|}dr' + V_{XC}(n_0)\right] \psi_i = \epsilon_i \psi_i \tag{2.16}$$

The solutions of the Kohn Sham equations are the Kohn Sham orbitals and have to be found in a self consistent fashion. The eigenvalue problem is a generalised one body Schrödinger Equation, with ϵ_i as Lagrange multiplier obtaining orthogonality for the Kohn Sham orbitals. Equation 2.15 makes use of the Kohn Sham orbitals to compute the electron density of the system.

With the exchange correlation energy $E_{XC}[n]$, the total energy of the system is reduced the functional

$$E_V(n_0) = T_s[n_0] + \int V_{ext}(r)[n_0](r)dr + J[n_0] + E_{XC}[n_0] \tag{2.17}$$

where n_0 is the ground state electronic density computed by the eigen functions of the Kohn Sham equation.

The introduced exchange energy correlation functional $E_{XC}[n_0]$ is related to the exchange potential correlation functional in 2.16 by

$$V_{XC}[n] = \frac{\delta E_{XC}[n]}{\delta n(r)}. \tag{2.18}$$

Chapter 3

Machine Learning

3.1 Motivation

3.1.1 From Density Functional Theory to Machine Learning

The description of atomic systems based on density functional theory, presented in chapter 2, results in significant reduction in computational cost compared to the exact solution of the many body Schrödinger equation. In general quantum mechanical systems are highly complex and diverse, in a way that exerting DFT transcends the computational resources. Further the solutions by DFT are limited to a precise system for which the calculations were conducted and need to be recalculated once the system changes. The motivation for utilising machine learning is the creation of a potential energy surface (PES), meaning a general description of a quantum mechanical system inside specific boundaries. This description is not susceptible to small changes in configuration and describes the system as a whole. General information on PES can be found in [8]. In this work machine learning is used to create potential energy surfaces, describing the energy of the atoms of a system as function of their environment. This atomic neighbourhood environment reacts sensitive to small changes of the atomic environment, for example the change of an atomic species or atomic quantity. [9, p. 1051]

The machine learning algorithm introduced in this work is not per se able to find quantum properties, like energies and forces, for atomic systems. The solutions of an ML model are based on training data the model, upon which a model makes classifications. Training data needs to be generated quantum mechanically, in this work it is computed using DFT.

The stochastic tool, underlying the ML model is the Gaussian process regression (GPR). Note that ML models, here GPR in specific, do not offer good extrapolations of problems. Valid output is generated for interpolation tasks. General informations on this are to be found in [10].

3.1.2 Relationship of Atomic Systems and their Energy

Predicting the quantum mechanical properties of atomic systems makes use of a stochastic framework. Other possibilities include utilising neuronal networks (NN). The stochastic tool used in this work is the Gaussian process regression (GPR).

To make predictions through Regression, the atomic energies of a set of atoms are introduced as a function of their configuration, this is their euclidean geometry. The hallmark of the interatomic potential is that the energy consists of a sum of energy functionals lying inside a, later introduced cutoff radius and negligible long range terms. This definition is found in [9, p. 1051]

$$E = \sum_{\alpha} \sum_{i \in \alpha} \epsilon_i^{\alpha} + lrc \tag{3.1}$$

- E ... Total Energy of atomic system
- ϵ_i^α ... Local Energy Functionals within a cutoff radius
- α ... Type of contribution to the total energy, descriptor type
- i ... Counts instances of the Local Energy Contributions
- lrc ... Long range contributions outside a cutoff radius, include polarisability, Van-der-Walls forces, etc.

The contributions ϵ_i^α range over a α, each denoting a specific type of Energy Functional. These Energy Functionals take, so called, descriptors as arguments and describe the atomic energy as functional of the atomic configuration. Descriptors characterise the given atomic constellation by mapping the cartesian coordinates of the atomic environment to a descriptor feature space. This descriptor feature space connects an atomic energy to an atomic constellation. This value pair serves as input for the ML model. Details are in [9].

3.2 Gaussian Process Regression

3.2.1 Prerequisites

Gaussian processes are used to handle regression problems. The data input consists of N data points $\mathbf{X}_n, \mathbf{t}_n = \{x^{(n)}, t_n\}_{n=1}^N$. The list of N input points $\mathbf{X}_n$ exists in the space of an H dimensional fixed set of basis functions $\{\phi(\mathbf{x})\}_{h=1}^H$ at the points $\{\mathbf{x}_n\}$ and is denoted by the matrix

$$R_{nh} \equiv \phi_h(\mathbf{x}^{(n)}) \tag{3.2}$$

the target value vector $\mathbf{y}_n$ is defined by

$$y_n \equiv \sum_h R_{nh} w_h \tag{3.3}$$

with the weights w_n being Gauss distributed with mean zero.

$$P(\mathbf{w}) = \text{Normal}(\mathbf{w}; 0; \sigma_w^2 \mathbf{I}). \tag{3.4}$$

For the sake of simplicity the following notation is employed.

- $\mathbf{y} = y_n$
- $\mathbf{R} = R_{nh}$
- $\mathbf{w} = w_h$

With [10, p. 540] the covariance matrix of $\mathbf{y}$ is introduced as:

$$\mathbf{Q} = \langle \mathbf{y}\mathbf{y}^T \rangle = \langle \mathbf{R}\mathbf{w}\mathbf{w}^T\mathbf{R}^T \rangle = \mathbf{R}\langle \mathbf{w}\mathbf{w}^T \rangle \mathbf{R}^T = \sigma_w^2 \mathbf{R}\mathbf{R}^T \tag{3.5}$$

3.2.2 Building the Kernel

The main assumption is that the weights $\mathbf{w}$ and consequently the vector $\mathbf{y}$ are Gauss distributed. This condition holds for any selection of points $\mathbf{X}_N$. The target values$t_n$are assumed to differ from the correlated function value y_n by a Gaussian noice of σ_ν^2. This determines the target value vector $\mathbf{t}$ to be Gaussian distributed.

$$P(\mathbf{w}) = \text{Normal}(\mathbf{w}; 0; \mathbf{Q} + \sigma_\nu^2 \mathbf{I}) \tag{3.6}$$

The covariance matrix of $\mathbf{t}$ is introduced as $\mathbf{C}$.

$$\mathbf{C} = \mathbf{Q} + \sigma_\nu^2 \mathbf{I} = \sigma_w^2 \mathbf{R}\mathbf{R}^T + \sigma_\nu^2 \mathbf{I} \tag{3.7}$$

Every data set $\mathbf{X}$ induces a covariance matrix, also named kernel. The covariance matrix is then extended by the other part of the training data set, the target values $\mathbf{t}$. Note that in this work the data set $\mathbf{X}$ contains the atomic configurations, more specific the descriptor values of the atomic configurations. The target values $\mathbf{t}$ are populated by atomic energies calculated by DFT. Details on the creation of the kernel are found in [10, p. 540]

In general the choice of the covariance matrix, that is the choice of the set of H basis functions is not predetermined. The only constraint is that it must not be negative definite. In this work a squared exponential kernel is used. [11, p. 83]

3.2.3 Making Predictions

The model makes predictions based on the Gaussian process algorithm using a data set consisting of N data points $\mathbf{X}_N$ and $\mathbf{t}_N$. The task is to make a prediction for value t_{N+1}. The covariance matrix $\mathbf{C}_{N+1}$ is the extended kernel for the target vector $\mathbf{t}_{N+1}$.

$$\mathbf{C}_{N+1} \equiv \begin{bmatrix} \begin{bmatrix} \mathbf{C}_N \end{bmatrix} & \begin{bmatrix} \mathbf{k} \end{bmatrix} \\ \\ \begin{bmatrix} \mathbf{k}^T \end{bmatrix} & \begin{bmatrix} \kappa \end{bmatrix} \end{bmatrix} \tag{3.8}$$

After some conversions detailed in [10, p. 543] the predicted $\hat{t}_{N+1}$ is given by

$$\hat{t}_{N+1} = \mathbf{k}^T \mathbf{C}_N^{-1} \mathbf{t}_N \tag{3.9}$$

and the error $\sigma_{\hat{t}_{N+1}}^2$ on the prediction

$$\sigma_{\hat{t}_{N+1}}^2 = \kappa - \mathbf{k}^T \mathbf{C}_N^{-1} \mathbf{k} \tag{3.10}$$

with

- $\mathbf{C}_N^{-1}$ ... Inverse kernel of kernel in equation 3.7
- $\mathbf{k}$ ... Vector containing mapping t_{N+1} to $\mathbf{t}$, part of matrix $\mathbf{C}_{N+1}$, $\mathbf{k} \equiv \langle t_{N+1}, \mathbf{t} \rangle$
- $\mathbf{t}_N$ ... Target value vector of training data
- κ ... Intrinsic value related to unknown target t_{N+1}, needed to scale the error, part of matrix $\mathbf{C}_{N+1}$, $\kappa \equiv \langle t_{N+1}, t_{N+1} \rangle$

An important advantage of the Gaussian process is that only $\mathbf{C}_N$ needs to be inverted. Omitting the inversion of $\mathbf{C}_{N+1}$ allows for the creation of a model using an arbitrary large number H of basis functions. This is because the inversion of $\mathbf{C}_N$ always scales with N^3 independent of H. Raising the number of Basis Function may improve the accuracy of the model.

3.3 Modifications of Gauss Process Regression for Quantum Mechanical use

The density functional theory in chapter 2 and the Gaussian process regression in chapter 3.2 lay the foundation for building a model and creating an interatomic potential. Data generated by DFT is input into a Gaussian regression machine and an atomic system is modelled. The machine learning fundamentals need to be augmented and further optimised for quantum mechanical use.

3.3.1 Total Energies and Forces

The first extension concerns the computation of total energies and nuclear gradients of those. In chapter 3.2 the Gaussian process regression was fit using per atom energies. This simplification does not hold, because total energies of systems are known and generated by DFT. Gaussian noise ν_E is added to the total energy of the system from equation 3.1. Procedural the computation of the kernel is identical to equation 3.7 but summarising over all N local energies of the system. [9, p. 1053]

$$E = \sum_{\alpha} \sum_{i} \epsilon_i^{\alpha} + \nu_E \tag{3.11}$$

$$\langle E_N E_{N'} \rangle = \sigma_w^2 \sum_{i \in N} \sum_{j \in N'} C(\mathbf{d}_i, \mathbf{d}_j) + \sigma_E^2 \delta_{NN'} \tag{3.12}$$

Computing forces is straightforward. The total forces relating to the total energies are the gradients with respect to any atom k of the system and the general coordinates ξ. Gaussian noise is added and denoted by ν_ξ.

$$\frac{\partial E}{\partial \xi_k} = \frac{\partial \sum_\alpha \sum_i \epsilon_i^\alpha}{\partial \xi_k} + \nu_\xi \tag{3.13}$$

The computation of the derivatives can be written as

$$\frac{\partial^2 \langle E_N E_{N'} \rangle}{\partial \xi_k \partial \chi_l} = \sigma_w^2 \sum_{i \in N} \sum_{j \in N'} \frac{\partial \mathbf{d}_i^T}{\partial \xi_k} (\nabla_{\mathbf{d}_i} C(\mathbf{d}_i, \mathbf{d}_j) \nabla_{\mathbf{d}_j}^T) \frac{\partial \mathbf{d}_j}{\partial \chi_l} + \sigma_E^2 \delta_{NN'} \delta_{\xi_k \chi_l} \tag{3.14}$$

Notice that for the all the equations 3.12, 3.13 and 3.14 a term representing Gaussian noise is added. This is outlined in chapter 3.2.2 and is due to an assumed Gaussian error of the target values $\mathbf{t}$.

3.3.2 Local Approximation - Cutoff Radius

To reduce computational cost of energies and their derivatives compact support is needed. In order to achieve this, a cutoff radius r_{cut} is introduced. The regarded local environment of every atom of a system is geometrically limited to the cutoff radius r_{cut}. The compact support for the local energies is provided by adding a cutoff function f_{cut} which smoothly changes to zero as the pairwise distance r approaches the limit r_{cut}. In this work only pairwise interatomic distances are represented by the parameter r. The local energy representation ϵ is given

$$\epsilon(\mathbf{d}_i, w, r) = f_{cut}(r) \sum_h w_h \phi_h(\mathbf{d}_i) \tag{3.15}$$

with the cutoff function f_{cut}, used by the GAP and QUIP software, referenced in Appendix 6.2

$$f_{cut} = \begin{cases} 1, & r \leq r_{cut} - d \\ \frac{\left[cos\left(\pi \frac{r - r_{cut} + d}{d}\right) + 1)\right]}{2}, & r_{cut} - d < r \leq r_{cut} \\ 0, & r > r_{cut}. \end{cases} \tag{3.16}$$

The cutoff radius r_{cut} may be changed, depending on the required precision of the model. The parameter d can also be set manually, if needed, but is typically set to 1 Å as the typical atomic length scale. Note that a larger r_cut may lead to a better fit but an increase in computation time. The concept is explained in [9].

3.3.3 Sparsification - Representative Data

Further reduction of computational cost is achieved by applying sparsification, a common machine learning concept. This practice tries to lower the dimension N of the covariance matrix by selecting a set of sparse points, the sparse configuration. If done correctly, sparsification projects all of the input data onto a subset of data points with very little loss of information. One can think of the sparse configuration as resemblance of the training data set, by selection of a highly representative set of points. Similarly to the the cutoff radius in chapter 3.3.2, the size of the sparse set is variable. A small number of sparse points reduces the computational cost of the model significantly but may also result in a loss of precision in prediction.

Let the number of sparse points be M and the number of data points in the original data set N. The covariance matrix C_{NN} is computed as detailed in chapter 3.7 for the original input configurations $\mathbf{x}_N$. [12, Appendix]

$$\mathbf{C}_{NN} = \langle E_N E_{N'} \rangle \tag{3.17}$$

Similarly, the covariance matrix C_{MM} is built for the set of sparse configurations $\mathbf{x}_M$

$$\mathbf{C}_{MM} = \langle E_M E_{M'} \rangle \tag{3.18}$$

and the covariance matrix C_{MN} mapping the sparse configuration to the original data configuration:

$$\mathbf{C}_{NM} = \langle E_N E_M \rangle \tag{3.19}$$

Putting it all together, the predicted value $t_{N+1}(x_{N+1})$ at a new atomic configuration x_{N+1} is written as

$$t_{N+1}(x_{N+1}) = \mathbf{k}^T (\mathbf{C}_{MM} + \mathbf{C}_{MN} \Lambda_{MM}^{-1} \mathbf{C}_{NM})^{-1} \mathbf{C}_{MN} \Lambda_{MM}^{-1} \mathbf{t} \tag{3.20}$$

with the final constituents

- $\mathbf{k}$ ... Vector containing mapping of t_{N+1} to $\mathbf{t}$, $\mathbf{k} \equiv \langle t_{N+1}, \mathbf{t}_M \rangle$
- $\mathbf{C}_{MM}$ ... Covariance matrix of sparse configuration
- $\mathbf{C}_{MN}$ ... Covariance matrix of sparse configuration and original configuration

- $\mathbf{t}$ ... Target Values of original configuration
- Λ_{MM}^{-1} ... Diagonal matrix, containing the Gaussian noise terms σ_E^2 for energies and σ_ζ^2 forces respectively.

The final form of the Gaussian process in equation 3.20 combines the computational cost saving mechanisms detailed in this chapter. Note that computational cost for each prediction scales with the number M of sparse points since multiplying $\mathbf{t}$ from the right in equation 3.20 has been precomputed at the training stage of the model, explained in [9, p. 1054]. The effects of sparsification and a Gauss distributed kernel allow for a fast machine learning model.

3.3.4 Descriptors - Representing the Atomic Neighbourhood Environment

The training data for the machine learning model introduced in 3.2 consists of the N data points $\mathbf{X}_n, \mathbf{t}_n = \{x^{(n)}, t_n\}_{n=1}^N$. The set $\mathbf{X}_n$ mirrors the raw atom positions of the quantum mechanical system, simulated by density functional theory. The set $\mathbf{t}_n$ are the target values for the ML process, these are atomic energies and forces.

Outlined in chapter 3.1, the machine learning model uses energy functionals for its calculations, taking descriptor values as arguments. Descriptors map the atomic configuration in cartesian coordinates to a feature space that is used as input data for building the model. The transformed data in the descriptor space is denoted by $\mathbf{X}_n$. An appropriate transformation of the raw input data to a descriptive feature space determines the success of the ML model. [9, p. 1055]

In general a descriptor is a function applied to an atom configuration and converting it to a feature space. Possible descriptions of an atomic neighbourhood environment take interatomic distances or angles into account. More complex descriptors include representing an atomic environment by neighbourhood density functions.

In this work a simple 2 body distance descriptor is utilised. This descriptor characterises atomic neighbourhood environments by the euclidean distance between atoms.

$$\mathbf{d}_{ij} = |\mathbf{r}_i - \mathbf{r}_j| \tag{3.21}$$

- $\mathbf{d}_{ij}$... interatomic distance
- $\mathbf{r}_i, \mathbf{r}_j$ ... position of the *ith*, *jth* atom

Chapter 4

Machine Learning Model Creation

4.1 Observed Systems

4.1.1 Water Molecule H2

The first observed and analysed system consists of two Hydrogen atoms forming a molecule. H_2 is pictured in cartesian coordinates in figure 4.1. The atom configurations stem from a molecular dynamics (MD) simulation computed using density functional theory. The Hydrogen atoms in the simulation are fixed in the x-y-plane and oscillate in z-direction.

With the system only changing in distance between the two hydrogen atoms, building a machine learning model using a two body distance abbreviation, presented in chapter 3.3.4, is sound. The atoms oscillate in one dimension with no changes in angle between them, making a 2 body descriptor taking only the distance between them into account suitable.

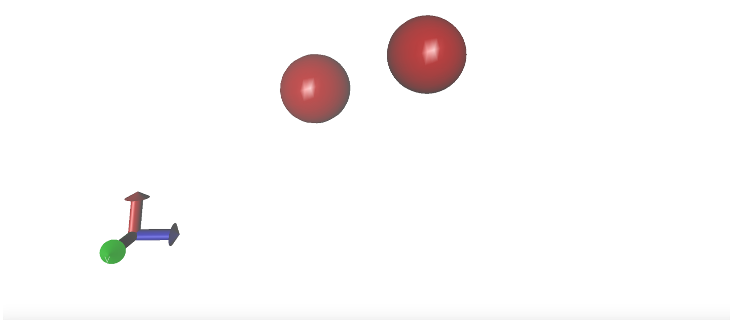

Fig. 4.1: H_2 molecule

4.1.2 Water Crystal

Second, a a larger more complex system of a Hydrogen crystal, containing 192 Atoms is introduced. Again the atom configurations are the output of a molecular dynamics (MD) simulation computed quantum mechanically making use of the Approximations of DFT. For simplicity reasons a MD is run for a bulk of atoms of just one type. The way the software, to create the ML model, is built this simplicity restriction could be lifted and the system could be updated to include multiple atom species.

The Hydrogen H in form of a crystal is depicted in figure 4.2. The MD run to generate the atomic configurations starts at a state of atoms out of ground state. The simulation generates training data for Hydrogen atoms relaxing towards an equilibrium state. The atomic configurations yielded each time step serve as as basis of the building process of the machine learning model.

To describe this system, again a pairwise distance descriptor is used. The two body descriptor oversimplifies the problem and there is a loss of information describing a 192 atom Hydrogen crystal. This is done to compare results of the same machine learning algorithm for two systems of differing complexity.

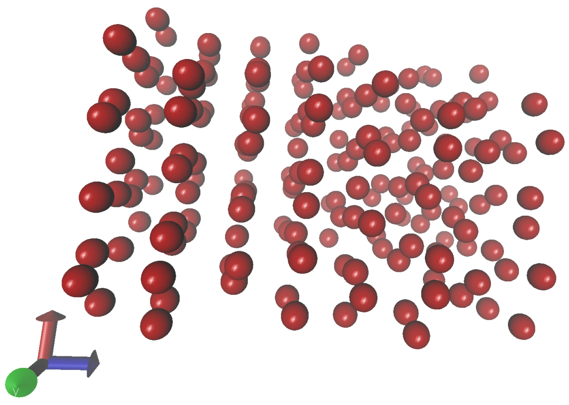

Fig. 4.2: Hydrogen Crystal

4.2 Workflow

4.2.1 Flow Chart

Creating the ML model is not a linear process, but a circular one. As can be seen in the flowchart 4.3 there are several steps building the model and turning the process into a result optimising feedback loop. The depicted Flowchart lays the foundation for creating and tuning the model and its steps are derived in the following chapters.

4.2.2 Get Data from Molecular Dynamics Simulation

Beginning at the top of the flowchart the first step on the way to creating a model is choosing which system is going to be modelled and then run a molecular dynamics simulation, briefly explained in chapter 4.1. The simulation uses DFT to yield atomic configurations for each time step, these serve as input to the model. Details on how DFT solves quantum mechanical problems are derived in chapter 2. The following chapter explain the workflow to go along with the flowchart on the example of the system containing the H_2 molecule.
The MD outputs a a set of atomic constellations with a corresponding energy per constellation and an associated force for each atom of the system per constellation. The data then needs to be transformed from an XML file coming out of the MD to a certain file format, namely an XYZ file so it can be interpreted correctly. Details are derived in Appendix 6.2

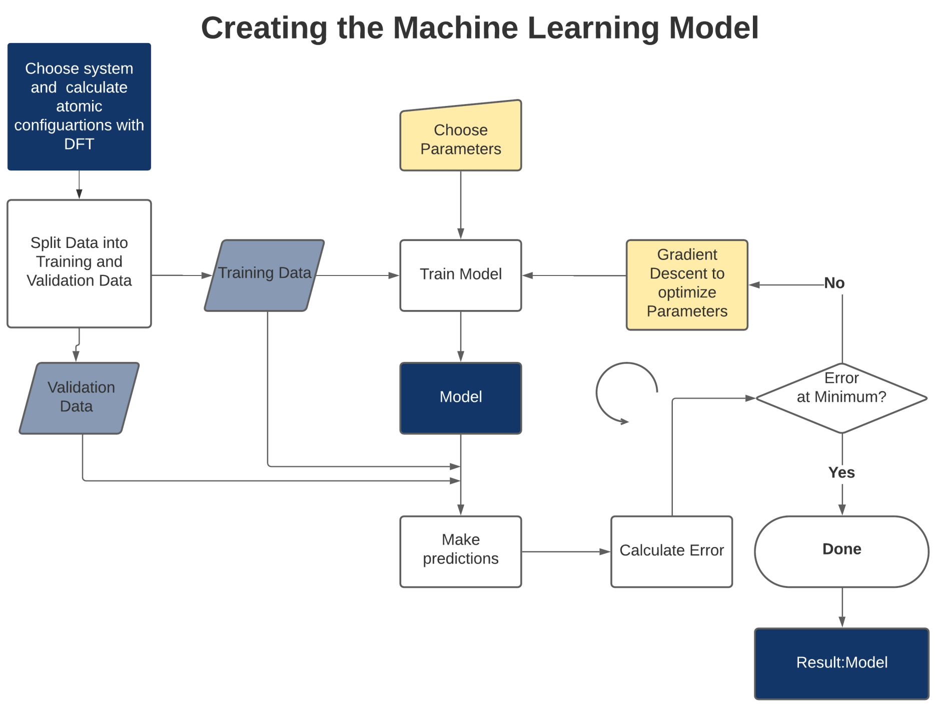
Creating the Machine Learning Model
Choose system and calculate atomic configuartions with DFT
Choose Parameters
Split Data into Training and Validation Data
Training Data
Train Model
Gradient Descent to optimize Parameters
No
Validation Data
Model
Error at Minimum?
Yes
Make predictions
Calculate Error
Done
Result:Model

Fig. 4.3: Flowchart

4.2.3 Split Data into Training and Validation Data

It is good practice in any ML process to split up the input data into two parts. [13, p. 17] One is going to be the training data used, directly used in the model building process and the other part is saved to evaluate and improve the model, that is validation data. Once the model has gone through the first iteration of its optimisation loop the validation data is used to see the quality of the model. This is done by letting the model make prediction on the validation data it has not been trained on. Saving a part of the input data for validation is done to prevent the model from overfitting, by using the validation data to refine the model. Overfitting means the model is highly accurate on reproducing the results of the training data but fails to correctly classify new input data.

In this case the split up between the training and validation data is done randomly and the ratio of the fit is set to 0.8. An 0.8 ratio delegates 80 percent of the input data for the model to training data and marks 20 percent as validation data. The ratio is by choice, but the way the code is written for this work it is possible to change split percentages and construct the model with a different split.

4.2.4 Model Training

Following the split into training data and validation data, the latter is put aside and the training data is input to the ML process which uses Gaussian Process Regression as underlying tool. Details are derived in chapter 3. The algorithm works like a black box where pairs of atomic configurations and the associated energies and forces are input to the model. In addition a set of parameters is demanded due to the algorithmic of the Gaussian Process.

There are parameters true to the nature of the Gaussian Process and further descriptor specific parameters. The chosen descriptor here is the 2 body distance. The black box model is trained, meaning in builds instructions by internally setting certain variables to certain values, which then enables the model to predict atom energies and forces for given atomic constellations.

The required parameters by the Gaussian Process represent the expected variance σ of the training data. [12, Appendix] Intuitively this can be thought of as intrinsic noise of the training data. Mathematical details can be found in chapter 3.2.

- σ_E ... Variance of training data energies
- σ_F ... Variance of training data force
- σ_S ... Variance of training data functional derivatives of energies, this is the virial stress [9, p. 1053]
- σ_H ... Variance of training data functional second derivatives of energies, this is the hessian [9, p. 1053]

The 4 σ parameters are the diagonal elements of a variance matrix.

The 2 body distance descriptor requires the following parameters to be set, details can be found in chapter 3.3:

- Cutoff radius ... sets geometrical distance for each atom up to which its environment is considered by the model, in Ångstrom
- Covariance type ... Sets how the kernel is built using the descriptor transformations of the input data as basis functions
- θ ... Goes into the computation of the covariance type as denominator. Determines the size of the classification bins by resembling the geometrical scale of the input data.
- δ ... Puts weight on the kernel for the chosen descriptors if there is more than one. The input settings determine the choice of δ.
- Sparse method ... Sets the sparsification method. The sparse data set is a representative selection of the input data
- Number of sparse points ... size of the sparse set, defines the magnitude of the reduction on the sparse point set

Of the needed parameters an educated choice is made for all but σ_E σ_F and θ, those are later optimised using a Gradient Descent Method. The predetermined parameters are set to the following values:

- Cutoff radius := 4 Ångstrom
- Covariance type := squared exponential [12, Appendix 2.1.], see equation 4.1
- $\delta := 1$, can be set to any value due to just one descriptor being used, hence no no weighting between different descriptors is needed
- Sparse method := Uniform, this means the sparse points are chosen based upon a selection of bins all of the same size. The size of the bins is determined by a uniform grid. [14, p. 2]

- Number of sparse points $:= 20$, for the H_2 system this is redundant and considers all atoms, but for the 192 atom H crystal the sparse representation is relevant
- $\sigma_S := 0.0$
- $\sigma_H := 0.0$

$$C(x_n, x_m) = \delta^2 exp\left(-\frac{1}{2}\frac{(x_n - x_m)^2}{\theta^2}\right) \tag{4.1}$$

- $C(x_n, x_m)$ ... Kernel, covariance matrix between two configurations
- δ ... Weighting between different descriptor kernels
- x_n ...Descriptor value of $n_t h$ configuration
- x_m ... Descriptor value of $m_t h$ configuration
- θ ... Hyperparameter, weighs the scale of input configurations

Note that the variance of the virial stress and the hessian in the training data are set to zero because no further deviation for the derivation is expected. This is due to the 2 body distance descriptor operating in one dimension. Choosing the fixed parameters is done based upon knowledge of the analysed atomic system. Varying the "fixed" parameters will not result in a recognisable change of the model. Thus not further optimising them is reasonable in respect to its cost.

The most dependencies on the quality of the model rely on σ_E σ_F and θ. It turns out that applying an optimisation algorithm to them returns the most precise results and converges fast to an optimum in fit. Still a guess needs to be made for σ_E σ_F and θ to then pass to the first iteration of the cyclic optimisation process. A first guess is made:

- $\sigma_E := 0.004\ eV$
- $\sigma_F := 0.08\ eV/Ångstrom$
- $\theta := 1.0$

Note that it makes sense to select the variance of force to be one order higher as the variance of energy. This is because the calculation of forces includes derivatives of energies. The variance of forces respects the error in forces and energies alike. The noise of energy values is propagated. Following the flowchart in 4.2.1 the model can enter the optimisation process with the input of

1. Training data consisting of atomic configurations with their corresponding energies and forces, and a
2. Set of fixed and variable parameters

4.2.5 Model Prediction

Using the model to make predictions requires any set of atomic configurations as input and gives a set of energies and forces as output. If the output values resemble the values of a quantum mechanical computation using, for example, density functional theory, then the model is correct. In detail the prediction yields one energy per atom configuration and forces for each atom for each spatial coordinate per atom configuration.

Going back to the flowchart in 4.2.1 the prediction is performed on the training data and additionally on the validation data. This is done to see how the model fares on input it has not been trained on. The mathematical details on how the prediction is performed based on the input data and the chosen parameters is derailed in chapter 3.

4.2.6 Model Optimisation

The model is trained and able to make predictions. To optimise the quality of the model a measure of goodness is needed. The error, in this case the root mean squared error (RMSE) is used as measure of inverse goodness. The RMSE is a measurement of how fare the predicted values are off compared to the real values. A perfect model would always predict values mirroring real values, barring any intrinsic inaccuracies in the data, leading to an RMSE of zero. Then root mean squared error is a simple measure of error taking the square root of the mean of the squared error of two values. Further computational details are explained in [15].

$$RMSE = \sqrt{\sum_{i=1}^{N} \frac{(\hat{X}_i - X_i)^2}{N}} \tag{4.2}$$

The denotations in 4.2 are

- $\hat{X}$ ... Predicted value
- X ... Real value
- N ... Number of real and predicted value pairs

Further computational details are explained in [15].
In general a model is considered valid if it performs well but not perfect on the training data. A perfect resemblance of the training data would lead to overfitting and to inaccurate predictions of new, unknown input data. The validation data comes into place as it is unknown input data for the model. For both, the training atomic configurations and validation atomic configurations, the associated energies and forces are known. In this case real means calculated by DFT, which is assumed to mirror the features of a real system precisely bewaring numerical errors.

To measure the quality of the fit, the trained model is used to make calculations on the training data and validation data and output the related energies and forces. The output is then compared to the real values of energies and forces by calculating the RMSE between them. This is done by calculating the RMSE for every real value, predicted value tuple and then added up. The summarised error is then proportional to the difference between the real energies and forces and predicted energies and forces.

A small RMSE means predicted and real values lie close to each other. This is a good fit and the model works correctly. To get a well performing model the training, predicting and calculating the error procedure is plugged into a cycling optimisation algorithm. Since the atomic configurations cannot be changed the goal of the optimisation is to minimise the RMSE by tuning the model parameters, namely σ_E σ_F and θ. The optimisation is done by a gradient descent Method. The Nelder-Mead downhill simplex algorithm is used to minimise a function, to which σ_E σ_F and θ are passed as independent arguments and and the RMSE is output as dependent variable. The dependent variable is to be minimised. Details on the Nelder-Mead downhill simplex algorithm, can be found in [16].

For this work, the built in python package *scipy* is used to minimise the RMSE . An example function call for the optimisation of H_2 can be seen in the code snippet below. Details can be found in Appendix 6.2.

```python
import scipy.optimize
initial_guess = [1,0.004,0.08]
result = scipy.optimize.minimize(RMSE_train_val,initial_guess,method='
    Nelder-Mead',
                                     options={'fatol':10e-5,'maxiter':100})
```

The following arguments are passed to the optimisation algorithm:

- RMSE_train_val is the function to be minimised
- initial_guess for the parameters σ_E σ_F and θ
- fatol is an exit condition for the algorithm, once the RMSE converges to a minimum with fatol, the process is stopped.
- maxiter is an exit condition for the algorithm. If the number of iterations of the optimisation process exceeds the value passed to maxiter it is terminated

Notice that the contribution of the RMSE of every real value, predicted value pair of energies and forces need to be scaled and cannot simply be added up and passed on to the minimiser. This is due to the ratio between the energies and forces for each configuration. Each atomic constellation hold energies for the whole system at a specific point wheres forces are attached to every atom of the constellation for every spacial coordinate at a specific point. Referring to the H_2 molecule of the MD simulation this means one energy for each time step compared to 6 forces, two atoms and three cartesian coordinates.

To scale the RMSE contributions correctly and to provide easy comparability to different atomic systems a per atom RMSE is introduced. In General, for any system, this is done by dividing both, the energies and the forces by the number of atoms ind the configurations. Additionally the Forces need to be divided by the number of spatial coordinates, for cartesian systems this means dividing by three. To prevent the optimisation process from overweighting the importance of either energies of forces, the RMSE needs to be scaled correctly.

$$RMSE = \frac{RMSE_{Energy}}{n_{Atoms}} + \frac{RMSE_{Force}}{n_{Atoms} * n_{Coordinates}} \tag{4.3}$$

The optimisation is terminated by one of the exit conditions and returns a value triple of σ_E σ_F and θ. For this value triple the RMSE is minimal for the given system and the model is optimised.

4.2.7 Water Crystal Model Creation

Having introduced the optimisation process on hand of the simple case of it can also be applied to any other system. Staying true to a single atom species but just a more complex system, the workflow as shown in the flowchart in 4.3 is applied on a system of a Hydrogen crystal containing 192 atoms. A visualisation is depicted in figure 4.2.

The workflow is identical to the one used for the H_2 molecule. First, a molecular dynamics simulation using the approximations of DFT generates atomic configurations with associated energies and forces. These configuration-value pairs are then split up into training and validation data. Second, training data is then used for building the model. The training data and the validation data is used to optimise the model by minimising the RMSE and optimising parameters. Notice that the scaling of the RMSE summands of energies and forces differs significantly from the scaling done for the H_2 molecule. The optimisation process returns an optimised parameters set for σ_E σ_F and θ different to the ones found for the molecule. The same model creating algorithm returns differentiating models for specific system.

Chapter 5

Outcome and Analysis

In chapter 4 a machine learning model able to predict atomic energies and forces was created. A model is of high quality if it resembles but also abstracts reality. The better the abstraction that is a model, the more accurate it mirrors reality. The ML model has been trained on real values and has been optimised by minimising variation from reality.

5.1 Water Molecule H2

5.1.1 Outcome Visualisation

Visualising the quality of a model can be done by plotting input against output, hence real values of energies and forces against predicted values of energies and forces. The following graphs show real values on the x-axis and predicted values on the y-axis. To get a cleaner look on the relationship between real and predicted values a help line is drawn which exactly matches real values and predicted values. If a dot, representing a predicted value lies directly on the drawn line then the prediction coincides exactly with a real value. In this case the RMSE, discussed in **??**, for this single value is zero. The further away a dot is drawn from the line, the higher the magnitude of the RMSE gets for this single value and the prediction is less accurate.

In this chapter the outcome of the ML model is analysed for the H_2 molecule. In figure 5.1 the energies before the optimisation process are plotted for training data and validation data respectively. Figure 5.2 show the forces of the training data and validation data with the initial guess for the parameters σ_E σ_F and θ.

Fixed and variable parameters are needed in addition to the training data for creating a model, referring to chapter 4.2.4. This set of parameters can be see in table 5.1 together with the corresponding RMSE.

Tab. 5.1: Parameters and RMSE before and after optimisation for H_2 molecule

	θ	σ_E	σ_F	RMSE
Before Optimisation	1	0.008	0.04	0.001276
After Optimisation	0.325703	0.007972	0.009353	0.0002264

The figures 5.1 and 5.2 depict the situation before the optimisation process, hence do not match reality yet. The initial guess for the parameters σ_E σ_F and θ and the corresponding RMSE in table 5.1 change after the model is handed to the optimisation process, explained in chapter 4.2.6. The second line of table 5.1 shows the tuned parameters together with the corresponding RMSE.

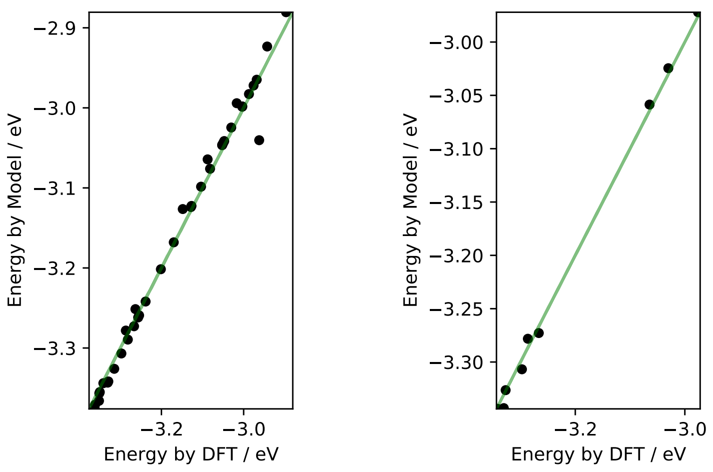

Fig. 5.1: Energies of training data (left) and validation data (right) of H_2 molecule before optimisation

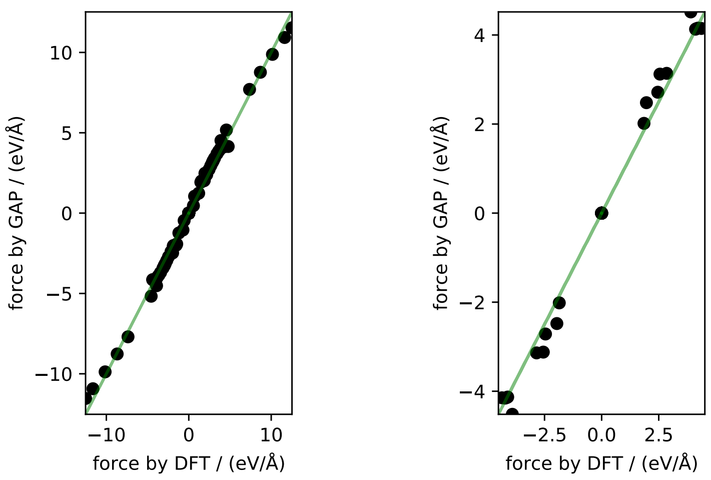

Fig. 5.2: Forces of training data (left) and validation data (right) of H_2 molecule before optimisation

The tuned parameters and minimised RMSE link to the figures 5.1 and 5.2. The energy of the training data and validation data is shown in figure 5.3. Respectively the force of the training data and the validation data can be seen in figure 5.4.

The optimised model leads to a smaller RMSE, see table 5.1 and the visual comparison of the graphs in figures 5.1 and 5.2 to the graphs in figures 5.3 and 5.4 shows dots lying closer to the optimal drawn line for the optimised model.

5.1.2 Model Parameter Analysis

Performing the optimisation and tuning the parameters σ_E σ_F and θ leads to an almost exact mirroring of real energies and forces by the model. For the H_2 molecule the gradient descent algorithm, introduced in chapter 4.2.6, terminated after 67 iterations and 137 function evaluations.

The almost exact reproduction or real energies and forces is due to the elementariness of the examined system. The considered H_2 molecule oscillates along the z-coordinate, hence is moving one dimensionally. The descriptor underlying the ML model takes 2 body distances into account. Actual atomic configurations are transformed to a feature space.The 2 body distances descriptor feature space allows an exact representation of the H_2 molecule oscillating in the distance between the Hydrogen Atoms. One to one mapping of the geometric atomic configuration to the feature space of the ML model leads to a small RMSE and classifications of high quality by the model.

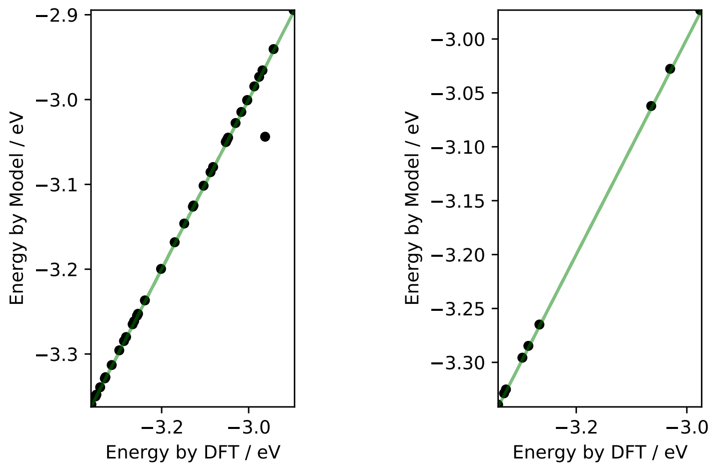

Fig. 5.3: Energies of training data (left) and validation data (right) of H_2 molecule after optimisation

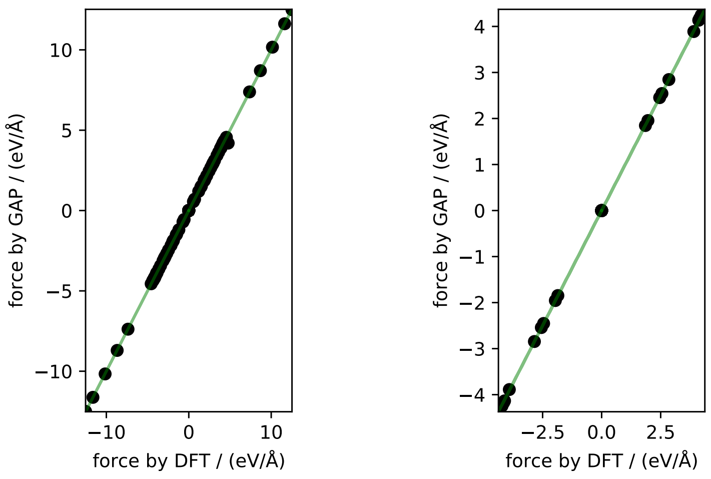

Fig. 5.4: Forces of training data (left) and validation data (right) of H_2 molecule after optimisation

5.2 Hydrogen Crystal

5.2.1 Outcome Visualisation

Similarly to the H_2 molecule the Hydrogen crystal is examined. The system is more sophisticated due to the greater number of atoms, the Hydrogen Crystal consists of 192 atoms compared to just two for H_2, but more so due to the complexity of the arrangement of the atomic constellation. The molecular dynamics simulation (MD) starts at an unordered state. During the simulation the Hydrogen atoms relax towards an equilibrium state.

To train and refine the model the same set of parameters is for the H_2 molecule and the Hydrogen crystal alike. In this chapter the goodness of the ML model of the Hydrogen crystal is analysed. A visualisation on how the model fares to reproduce the quantum mechanically calculated values, before the start of the optimisation process can be found in figure 5.5 for the energies of the training data and the validation data. The forces of the training data and for the validation data can be found in figure 5.6. The green diagonal line serves as target line for the predicted values. If on the line the predicted values match the real values. By looking at the figures 5.5 and 5.6 and checking the positions of the dots, representing the classifications of the model, one can see that the initial model does not mirror reality. Additionally, the values of the parameters and the RMSE can be seen in table 5.2.

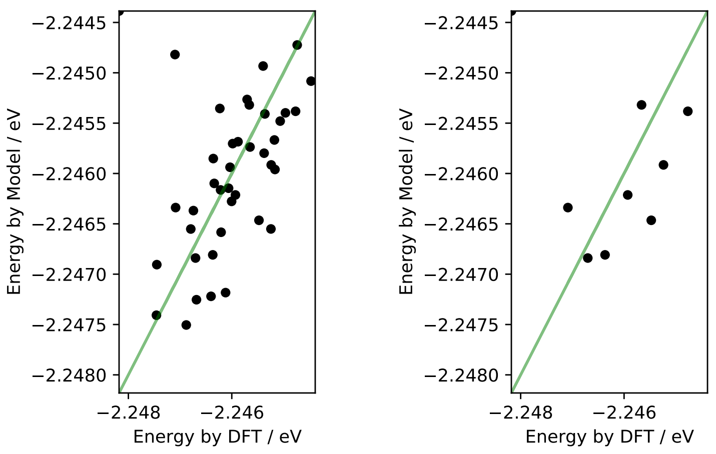

Fig. 5.5: Energies of training data (left) and validation data (right) of Hydrogen crystal before optimisation

Tab. 5.2: Parameters and RMSE before and after optimisation for Hydrogen crystal

	θ	σ_E	σ_F	RMSE
Before Optimisation	1	0.008	0.04	0.000743
After Optimisation	0.384155	0.065248	0.052590	0.000259

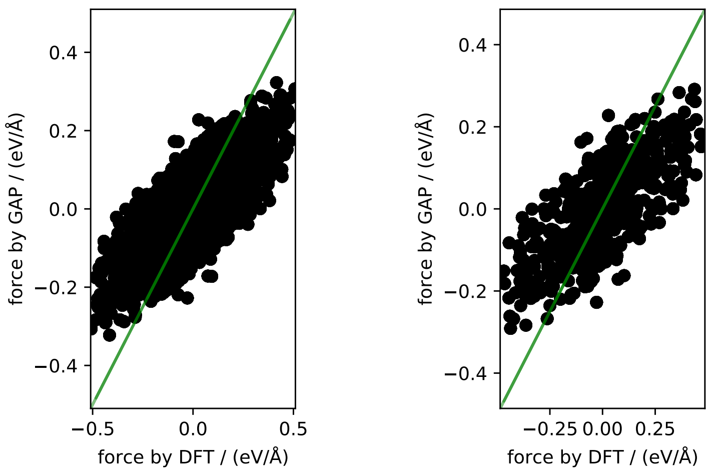

Fig. 5.6: Forces of training data (left) and validation data (right) Hydrogen crystal before optimisation

Starting out with the same set of parameters for the H_2 molecule, in table 5.1 and the Hydrogen crystal, in table 5.2, σ_E σ_F and θ change significantly after the optimisation process is run. The values for the variable parameters vary from the ones found in table 5.1. A visualisation of the predictions of the optimised values for the energies can be found in figure 5.7 for the training data and the validation data. Predictions on forces of the training data and the validation data are shown in figure 5.8.

5.2.2 Model Parameter Analysis

Two main conclusions can be drawn from the optimisation process of the model of the Hydrogen crystal. First, the RMSE error is reduced significantly throughout the optimisation process. Looking at the plots before the appliance of the optimising gradient descent algorithm in figures 5.5 and 5.6 and after in figures 5.7 and 5.8, one can see that the dots resembling the predicted values of energies and forces in respect to their real values have moved drastically towards the optimal green line. The reduction of the RMSE can be seen in table 5.2. The optimisation process terminated after 46 iterations and 104 function evaluations.

Second, the plots in figures 5.7 and 5.8, together with the RMSE in table 5.2 show the relatively low quality of the model compared to the H_2 molecule. The predictions for the Hydrogen crystal feature a larger variance in respect to the corresponding real values than the predictions of the model for the H_2 molecule. Looking at the predictions of the model in the figures 5.7 and 5.8 one cannot see the that predicted and real values coincide on every occasion, but a strong trend is visible.

The reason the model does not classify energies and forces for the Hydrogen crystal as well as it does for the H_2 molecule is the choice of descriptor. Both systems are represented by a 2 body distance descriptor. A one dimensional distance suffices the requirements of the oscillating H_2 molecule but does not for a 192 Atom Hydrogen crystal relaxing to an equilibrium state. The descriptor fails to precisely map the complex 192 Atom constellation to a feature space serving as input for the ML model. The loss of information by choice of a 2 body distance descriptor determines the worse prediction performance of the model compared to the precise predictions of the model for the H_2 molecule. The interatomic interference of the larger system exceeds the sphere of the simple 2 body distance descriptor.

Fine tuning the parameters leads to the best achievable results, visualised in the figures 5.7 and 5.8 with the values off the optimal predictions but a visible slope and a trend towards correct classification.

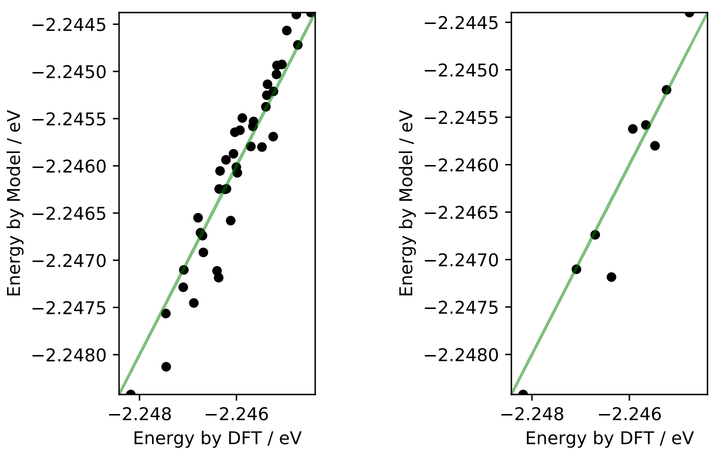

Fig. 5.7: Energies of training data (left) and validation data (right) of Hydrogen crystal after optimisation

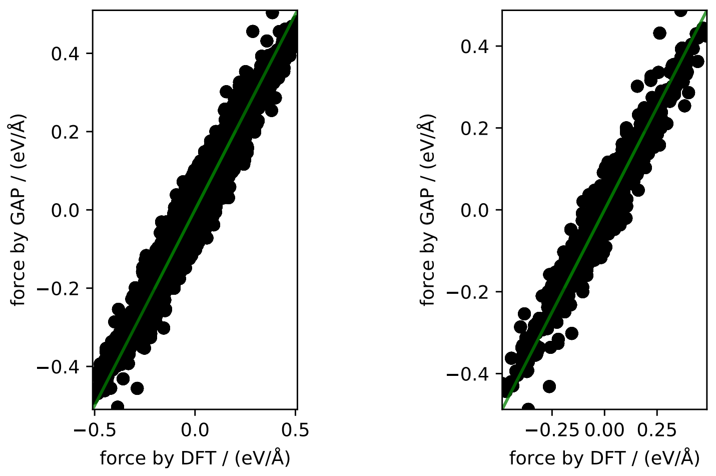

Fig. 5.8: Forces of training data (left) and validation data (right) of Hydrogen crystal after optimisation

Chapter 6

Summary and Outlook

6.1 Model Improvement by Descriptor Adjustment

Analysing the choice of parameters in chapter 5 the optimal selection of θ, σ_E and σ_F leads to great results for the H_2 molecule and to satisfying results for the Hydrogen crystal system. Using the same descriptor as feature space for both atomic structures the deficiencies are outlined modelling the 192 atom Hydrogen crystal.

While, as pointed out in chapter 5.1.2, the 2 body distance suffices the H_2 molecule oscillating in one dimension, it does fail to express the interactions present in the Hydrogen crystal. One possible solution can be to expand the feature space used to describe the atomic system by adding or changing descriptors. Other descriptors may be used to resemble the atomic configurations more accurately in case of larger systems, as is the Hydrogen crystal. A descriptor taking angles between objects into account would be a potential expansion of the feature space. This would upgrade the geometrically mapped feature space from one dimension to three dimensions. That is the case for a 3 body angle descriptor taking angles between 3 atoms into account. The software utilised for this work in Appendix 6.2 offers a 3 body angle descriptor amongst others. Making use of more complex descriptors could have positive impact to the modelling process and result in an improved fit.

6.2 Larger Systems

Predicting energies and forces built on appropriate descriptor feature spaces and parameters opens up machine learning solutions for larger and more complex systems than the H_2 molecule in chapter 4.1.1 and the Hydrogen crystal in chapter 4.1.2. The suboptimal choice of the 2 body distance descriptor for a 192 atom problem leads to classifications trending in the right direction, although not precise results. The correctly reproduced trend shows the possibilities of using machine learning algorithms for the prognosis of the behaviour of quantum-mechanical systems. As pointed out in chapter 6.1 a fitting descriptor choice can lead to more exact precision in predictions.

Larger, more diverse systems may require combinations of different descriptors than the 2 body distance descriptor utilised in this work. Gaining a model of high quality demands a sound choice of descriptors and refined parameter values. Additionally, the goodness of the model depends on the quality of the Input Data, computed with DFT. The input data, providing the model with training data and validation data, is generated prior to building the model. The importance of the input data becomes significant for diverse systems, possibly containing multiple atomic species.

Note that the creation of the model does not require any quantum-mechanical calculations and depends solely on the input data. This is pointed out in chapter 4.2.2. Skipping quantum-mechanical calculations, for example by DFT, and making prognoses based on a trained machine learning model reduces runtime and cost of computation. This makes machine learning essential for the challenge of describing complex atomic systems.

Appendix

A Preparation of Training Data

The atomic environments used in this work are yielded by a *Molecular Dynamics (MD) Simulation.* The data is computed using *Density Functional Theory (DFT)* and each time-step of the *MD* simulation corresponds to one atomic configuration.

The *MD Simulation* output is primordially written to the *XML* file. The atomic configurations with their related energies and forces need to be extracted from the *XML* file using a shell script. The script can be found in (TODO: Simon Repository) The machine learning algorithm demands input data in format of an *XYZ* file. In respect to the needed output file format, the shell script transforms the data by extracting it from the *XML* file and writing it to an *XYZ* file.

B Software

The *Machine Learning* software packages used in this work are *GAP* ,*QUIP* by *Albert Bartok-Partay, Noam Bernstein, Gabor Csanyi* and *James Kermode.* [17] The software is mostly constructed in *Fortran*, with a *Python* user interface. The source code can be found in [18]. Also the python package *Atomic Simulation Environment (ASE)* is used. [17]

C Code

Preparing the *Training Data*, performing a *Split*, training the *ML* model, optimising the model parameters and generating plots is done using modules and methods written in *Python.* The entirety of the code is stored in (TODO: Simon Repository).

D Optimisation

To optimise the *Machine Learning* model its parameters are optimised. This is done by minimising the *Root Mean Squared Error (RMSE)* between the predictions of the model and computations by *DFT*. The minimisation is done using a python module, that is *scipy.py* [19]. The minimisation performs a *Gradient Descent* method, namely the *Nelder-Mead-Algorithm (NM).* The *NM* method is a *Downhill Simplex* algorithm for minimising non-linear functions in respect to multiple parameters.

The process, introduced by *John Nelder* and *Roger Mead* in 1965, does not implicitly compute derivations of the function to be minimised. Such an algorithm is needed, since the *ML training* and *prediction* process does not offer a function on which algebraic operations can be performed on. The creation of the model is a *Black-Box* for the optimisation algorithm. Specifics on the *Nelder-Mead* algorithm can be found in [16]

Bibliography

[1]T. Hande, *Lecture notes on principles of density functional theory*, `http://www.physics.metu.edu.tr/~hande/teaching/741.html`, accessed: 05.12.2020.

[2]M. Hjorth-Jensen, *Computational physics*, 2008.

[3]M. Pitschmann, *Atom-, kern-, und teilchenphysik i* (2020).

[4]*Lecture 3: from many-body to single-particle: quantum modeling of molecules*, `https://ocw.mit.edu/courses/materials-science-and-engineering/3-021j-introduction-to-modeling-and-simulation-spring-2012/part-ii-lectures-videos-and-notes/lecture-3/`, accessed: 12.04.2021.

[5]W. Kohn, *Nobel lecture: electronic structure of matter—wave functions and density functionals*, 1999.

[6]J. Hutter, *Dichtefunktionaltheorie*, 2004.

[7]S. B. Paschen and P. Mohn, *Festkörperphysik ii*, 2020.

[8]J. C. C. J. Espinosa-Garcia M. Monge-Palacios, "Constructing potential energy surfaces for polyatomic systems: recent progress and new problems", (2012).

[9]B. A. P. and S. G., "Gaussian approximation potentials: a brief tutorial introduction", J. Quantum Chem, 115, 1051–1057, `10.1002/qua.24927` (2014).

[10]D. J. MacKay, *Information theory, inference, and learning algorithms* (Cambridge University Press, 2003).

[11]D. K. Duvenaud, *Automatic model construction with gaussian processes* (2014).

[12]A. P. Bartók, M. C. Payne, R. Kondor, and G. Csanyi, "Gaussian approximation potentials: the accuracy of quantum mechanics, without the electrons", (2009).

[13]A. C. Müller and S. Guido, *Introduction to machine learning with python* (O´Reilly Media, Inc. 1005 Gravenstein Highway North, Sebastopol, CA 95472, 2017).

[14]C. G. Vladimir Blinovsky, "Asymptotic enumeration of sparse uniform linear hypergraphs with given degrees", (2016).

[15]*Root-mean-squared-error an overview*, `https://www.sciencedirect.com/topics/engeneering/root-mean-squared-error`, accessed: 07.04.2021.

[16]W. H. Charles Audet, *Derivative-free and blackbox optimisation* (Springer, 2017).

[17]*Gnu general public license, version 2*, `https://github.com/libAtoms/QUIP`, Last retrieved 2021-31-01, 2006.

[18]*Libatoms git repository*, `https://github.com/libAtoms`, accessed: 19.04.2021.

[19]E. Jones, T. Oliphant, P. Peterson, et al., *SciPy: open source scientific tools for Python*, 2001.