universität wien

MASTERARBEIT / MASTER'S THESIS

Titel der Masterarbeit / Title of the Master's Thesis

"Neural Network Potentials with Long-range Charge Transfer and Electrostatics"

verfasst von / submitted by

Philipp Misof, BSc

angestrebter akademischer Grad / in partial fulfillment of the requirements for the degree of

Master of Science (MSc)

Wien, 2023 / Vienna, 2023

Studienkennzahl lt. Studienblatt / degree programme code as it appears on the student record sheet:	UA 066 876
Studienrichtung lt. Studienblatt / degree programme as it appears on the student record sheet:	Masterstudium Physik
Betreut von / Supervisor:	Univ.-Prof. Mag. Dr. Christoph Dellago

Abstract

Based on recent works, we implemented the next generation of high-dimensional neural network potentials (HDNNPs) into the efficient C++ package *n2p2*. In addition to the preceding version, which could compute potential energies and forces of arbitrarily sized structures with ab initio accuracy, this new method models atomic charges explicitly, based on a global charge equilibration scheme. This lifts the previous limitation of not capturing the effects induced by long-range charge transfer and thus, improves the accuracy for inhomogeneously charged systems.

In this work, the theory behind HDNNPs and their training was elaborated in detail. The Hirshfeld charges, the charge equilibration scheme and the Ewald summation, including its ambiguity, were discussed thoroughly as well. To extend the possible applications of the novel method, we made the so-called 4G HDNNP available for molecular dynamics (MD) simulations by developing an interface to the popular MD simulation package *LAMMPS*. The new functionality was showcased by computing the charge distribution and the molecular dipole moment of liquid water in an MD simulation. It was verified that the accuracy of the predicted charges is high enough for an analysis of the electrostatic properties of water. Furthermore, we showed that the Hirshfeld atomic charges noticeably underestimate the dipole moment of the molecules in liquid water, which is in agreement with the previously found underestimated dipole moment in small water clusters. Finally, we analyzed algorithm-specific properties, including the performance with respect to the system size and the parallel efficiency as well as the error of the custom developed Ewald truncation method.

Zusammenfassung

Aufbauend auf vorangegangenen Arbeiten implementieren wir die nächste Generation der hochdimensionalen neuronalen Netzwerkpotentiale (HDNNPs) in das effiziente C++ Programm *n2p2*. Zusätzlich zur Vorgängerversion, welche bereits potentielle Energien und Kräfte beliebig großer Systeme mit ab initio Genauigkeit bestimmen konnte, beinhaltet die neue Methode die explizite Modellierung atomistischer Ladungen, welche über einen globalen Ladungsausgleich bestimmt werden. Zuvor konnten Effekte, die durch einen langreichweitigen Ladungsaustausch induziert werden, konstruktionsbedingt nicht beschrieben werden, was mit dieser neuen Version allerdings behoben wird und damit die Genauigkeit des Modells für inhomogen geladene Systeme verbessert.

Im Rahmen dieser Arbeit wird die Theorie der HDNNP und deren Training detailliert ausgearbeitet. Des Weiteren werden Hirshfeldladungen, die Ladungsausgleichsmethode, sowie die Ewaldsummation inklusive ihrer Mehrdeutigkeit ausführlich besprochen. Um das Anwendungsgebiet der neuen Methode zu erweitern, haben wir die sogenannten 4G HDNNPs für die Molekulardynamik-Simulation (MD) verfügbar gemacht, indem eine Schnittstelle zu dem viel verwendetem MD Simulationsprogramm *LAMMPS* entwickelt wurde. Die gewonne Funktionalität wurde durch die Berechnung der Ladungsverteilungen sowie des molekularen Dipolmoments von flüssigem Wasser in einer MD Simulation demonstriert. Es wurde sicher gestellt, dass die Genauigkeit der vorhergesagten Ladungen ausreichend für eine Untersuchung der elektrostatischen Eigenschaften von Wasser ist. Darüber hinaus zeigten wir, dass die Hirshfeldladungen zu einer beträchtlichen Unterschätzung des molekularen Dipolmoments in flüssigem Wasser führen, was im Einklang mit vorangegangenen Untersuchungen steht, die dieses Phänomen für kleine Wassercluster zeigen konnten. Schließlich haben wir noch Algorithmus-spezifische Eigenschaften wie die Rechenleistung in Bezug auf Systemgröße und Paralleleffizienz, sowie den durch die Abbruchmethode für die Ewaldsummation verursachten Fehler analysiert.

Acknowledgements

I would like to express my deep gratitude to Prof. Christoph Dellago and Dr. Andreas Tröster, who introduced me to the broad and highly contemporary field of machine learning techniques in condensed matter physics. Not only did Prof. Christoph Dellago always provide me with valuable ideas to solve the problems I encountered, but he also offered an atmosphere in which I could easily integrate into the research group even under the restriction that the global coronavirus pandemic imposed. The enthusiasm of Dr. Andreas Tröster to the mathematical background of my field motivated me to the rigorous treatment of chapters 2 and 3, in which I have found great pleasure. Even for particularly involved questions he always had a sympathetic ear for me.

The support of Dr. Andreas Singraber made the level of professionalism in this project possible. As the creator of *n2p2*, he offered me a solid framework, in which I could develop my work and he always made time for our recurring discussions, which provided valuable insights to me. I also want to thank the other researchers in the Dellago group, in particular Lukáš Kývala, Alexander Gorfer and Salvatore Romano for their individual expertise that they have shared with me in our countless discussions. Without their hints and ideas, this thesis would not have been finished yet.

Finally, I would like to thank my family and my girlfriend Anneliese. Their encouragement and support were an integral part of reaching this point in my academic life. Although my mother and my uncle Stefan cannot experience the outcome of their support, I dedicate this work to their memory.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

Computational methods have become one of the backbones of today's condensed matter physics. They enable the investigation of materials on an atomic or subatomic level even under conditions that are technically not yet accomplishable in an experiment. Despite the exponential growth in computer performance over the last decades, there is, however, still a compromise to be made. While highly accurate and transferable methods exist, they often are computationally too expensive for the study of larger systems in a within reasonable time frame. Additionally, when those methods should be applied in molecular dynamics (MD) simulations, the physical time that is simulated is usually too short to capture rare events. Classic examples belonging to this group of methods are ab initio approaches like density functional theory (DFT) based algorithms, which solve the quantum mechanical ground state of the electrons in the system. In contrast, if large simulations become necessary, one typically has to resort to empirical models, which simplify the description of the interactions drastically for example by only considering the most prominent interaction sites inside a molecule. These models are often highly specific to the structure that is investigated and contain parameters that do not correspond directly to experimental observable quantities but are instead tweaked until the model can reproduce certain characteristics of the material. Examples of this kind of force field are the water models TIP4P [1] or SPC/E [2] and their variants. The advantage of being several orders faster than ab initio methods comes at the cost of lacking transferability and often the inability to describe the formation and breaking of bonds. Furthermore, they are only accurate in certain thermodynamic regimes.

Different approaches have been developed to find a better compromise between accuracy and computational efficiency. Recent years have shown that the branch of machine learning (ML) techniques provides promising potential in this area, especially kernel-based methods [3, 4] and models based around neural networks (NNs) are now being used extensively [5–7]. In this thesis, we focus on the latter, more specifically on the Behler-Parrinello high-dimensional neural network potentials (HDNNPs) [8]. Although Behler and Parrinello have not been the first one to successfully use NNs to construct force fields [9–11], they demonstrated how systems of arbitrary size can be described with the same neural network potential (NNP) with ab initio accuracy. This is achieved by formally partitioning the total potential energy of the system into contributions that are assigned locally to each atom and depend on the neighboring atoms inside a cutoff sphere. In an intermediate step, it is also made sure that physical symmetries are respected by the NNP. This model is still actively used, but since extensions of it have eventually been developed, it has been coined second generation or 2G HDNNP [12]. The next iteration, called 3G, introduced a second type of

NN that evaluates locally dependent atomic charges analogously to the local energy contributions and uses them to explicitly model electrostatic interactions [13]. Its advantage lies in the fact that those interactions are long-ranged and may thus even be relevant between atoms with a distance larger than the cutoff radius. Such effects cannot be captured by the 2G version by construction. Additionally, the charges can be used for further analysis, e.g. dipole moments. However, there still remains a limitation of this model. Certain structures can exhibit polarization effects over large distances, which could, in principle, be modeled by charge transfer. But since this effect can spread over a range that may be larger than any reasonable cutoff, the charges, which are predicted from the local neighborhood, will not be able to reliably describe this phenomenon. Furthermore, there is no guarantee that total charge conservation is conserved since this is a global property and cannot be imposed locally. Recently, this problem has been addressed by Ko et al. [14], who introduced the next iteration of HDNNPs, 4G. Essentially, it replaces the charge prediction method in 3G with a model that was worked out by Ghasemi et al. [15]. Instead of using NNs to predict charges from local neighborhoods directly, the idea is to replace them with electronegativities. These locally determined quantities are subsequently used in a global charge equilibration scheme that, in addition, conserves the total charge of the system. Whereas Ghasemi et al. [15] incorporated only the electrostatic interaction for the full description of the total potential energy, Ko et al. [14] used a second class of NNs to fit remaining terms in the potential energy.

In this work we will implement a 4G HDNNP into the *n2p2* [6, 16] package, which previously implemented the 2G version. It is an efficient, parallelized C++ program that offers an interface to the popular MD simulation package *LAMMPS* [17, 18]. This will enable the 4G approach to be used in MD simulations, which we will test for liquid water. A careful study of the performance in both accuracy and computational efficiency will be conducted and we will demonstrate how this generation of NNPs can be used for electrostatic analysis in MD, like the computation of molecular dipole moments. On top of that, we will present a comprehensive overview of the mathematical background that is needed to fully understand NNPs as well as the electrostatic theory that is involved.

The thesis is structured as follows. In chapter 2 the theory behind HDNNP is presented. This involves a general discussion about NNs, the statistical learning problem and the Kalman filter, which is the training algorithm that is used in this work. Subsequently, the details of the Behler-Parrinello HDNNP are presented. It then follows a comprehensive elaboration about the electrostatic theory that is necessary for 4G in chapter 3. This involves the concept of atomic charge partition, in particular the Hirshfeld method and its refinement and the charge equilibration scheme that is used. In addition, the computation of the electrostatic energy under periodic boundary conditions (PBCs) is discussed in detail, where great emphasis is put on the ambiguity of this quantity. We then continue in chapter 4 with the discussion of the computational aspects that are necessary for the implementation of 4G. This includes a truncation scheme of the Ewald summation based on a statistical error estimate, and intermediate quantities that are necessary for the training or the application of 4G HDNNPs. Following this, the methods of studying liquid water with the new NNP in the context of MD are explained in chapter 5, where we will give a concise summary of the NNP setup and the training procedure as well as the MD simulation and the subsequent analysis of the generated data. In chapter 6 the obtained results will be presented and discussed in the context. Finally, a summary of this thesis will be given in chapter 7, where we will take the opportunity to give an outlook of the near future of 4G HDNNPs.

Chapter 2

Neural Network Potentials

2.1 Feedforward Neural Networks

Artificial neural networks are a special class of functions and belong to the area of machine learning. NNs are deployed in countless applications in different areas, ranging from tasks like image processing [19], speech recognition [20], to medical problems [21] and even in physics NNs are widely used by now [8, 22, 23]. In this work we will only discuss feedforward NNs.

Definition 2.1 (Feedforward Neural Network)**.** A *feedforward NN* with $L-2$ *hidden layers* and N_i *neurons* in layer i is a function of the form

$$\begin{aligned} \Phi : \mathbb{R}^d &\to \mathbb{R}^{N_L} \\ \boldsymbol{x} &\mapsto \boldsymbol{\Phi}(\boldsymbol{x}) = \boldsymbol{W}_L \boldsymbol{\sigma}(\boldsymbol{W}_{L-1}\boldsymbol{\sigma}(\dots \boldsymbol{\sigma}(\boldsymbol{W}_1 \boldsymbol{x} + \boldsymbol{b}_1)\dots) + \boldsymbol{b}_{L-1}) + \boldsymbol{b}_L, \end{aligned} \tag{2.1}$$

where $\boldsymbol{W}_i \in \mathbb{R}^{N_{i-1}\times N_i}$, $(\boldsymbol{W}_i)_{jl} = w_{jl}$ are called the *weights*, $\boldsymbol{b}_i \in \mathbb{R}_i^N$ the *bias* vectors and $\sigma : \mathbb{R} \to \mathbb{R}$ the (non-linear) *activation function*[1] which in this case is acting component-wise on its argument. The tuple $(d, N_1, \dots, N_L)$ is called the *architecture* of the NN and describes the number of neurons in each layer. The first layer with d neurons is called the input layer and the last layer with N_L neurons is called the output layer. The number of neurons in the NN is $N_\Phi = d + \sum_{i=1}^{L} N_i$.

Note that the NN can be interpreted as a composition of layers, where each layer is composed of the affine map $\boldsymbol{A}_i(\boldsymbol{x}) = \boldsymbol{W}_i\boldsymbol{x} + \boldsymbol{b}_i$ and the activation function σ_i. Visually speaking the elements of the vector $\boldsymbol{A}_i(\boldsymbol{x})$ can be seen as the values of the neurons in layer i before the activation function is applied. One often considers the space of all NN with the same activation function σ and the same architecture $(d, N_1, \dots, N_L)$, which we denote by $\mathcal{H}^{\sigma}_{(d,N_1,\dots,N_L)}$.

It turns out that NNs are universal approximators [24–26] meaning they are able to approximate any continuous function on a compact subset.

Theorem 2.1 (Universal Approximation Theorem [26])**.** *Let* $\boldsymbol{f} : K \to \mathbb{R}^{N_L}$, $\boldsymbol{f} \in \mathcal{C}(K)$, $K \subset \mathbb{R}^d$ *compact and* $L > 1$*. If* $\sigma_i = \sigma \in \mathcal{C}(\mathbb{R})$[2] *is non-polynomial, then for any* $\varepsilon > 0$ *there exist* $N_1, \dots, N_L \in \mathbb{N}$ *and a* $\boldsymbol{\Phi} \in \mathcal{H}^{\sigma}_{(d,N_1,\dots,N_L)}$ *such that*

$$\sup_{\boldsymbol{x}\in K} |\boldsymbol{f}(\boldsymbol{x}) - \boldsymbol{\Phi}(\boldsymbol{x})| < \varepsilon.$$

[1] In practice, the activation function sometimes varies from layer to layer, but this case is not relevant for the discussion in this chapter.

[2] In [26] a less restrictive case is studied

This makes NNs appealing to regression, classification and probability density estimation problems, just to name a few.

2.2 Training of Neural Networks

2.2.1 Statistical Learning Problem

An abstract definition of the learning problem was given by Mitchell:

Definition 2.2 (Learning Problem)**.** "A computer program is said to learn from *experience* E with respect to some class of *tasks* T and *performance* measure P, if its performance at tasks in T, as measured by P, improves with experience E." [27]

In the context of NNs the task T could be the regression or classification problem. Regression consists of deducing a good approximation of an unknown function $f : \mathbb{R}^d \to \mathbb{R}^{N_L}$ only from the given data $\boldsymbol{z} = \{(\boldsymbol{x}_1, \boldsymbol{y}_1), \ldots, (\boldsymbol{x}_m, \boldsymbol{y}_m)\}$. The data is related to the unknown function by $\boldsymbol{y}_i = \boldsymbol{f}(\boldsymbol{x}_i) + \boldsymbol{\xi}$, where $\boldsymbol{\xi}$ represents noise (e.g. from a measurement process) in form of a random variable. Note that here the target values $\boldsymbol{y}_i$ are given, which makes this approach a *supervised learning* problem. In contrast, *unsupervised learning* tries to estimate the properties of the probability density $\rho(\boldsymbol{x})$ only by the given data set $\{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_m\}$ [28]. The data corresponds to the experience E and "more" experience corresponds to a larger m. In the following, we will restrict ourselves to the case $f : \mathbb{R}^d \to \mathbb{R}$ because a vector-valued function can be represented by scalar-valued functions as components. In the context of the regression problem the performance measure P usually quantifies the error of the approximation Φ at the given data points. A popular choice is the least squares method for the *empirical error* [29]

$$\mathcal{E}_{\boldsymbol{z}}(\Phi) = \frac{1}{m} \sum_{i=1}^{m} \left(\Phi(\boldsymbol{x}_i) - y_i\right)^2 .$$

This error is dependent on the data set but the goal is to minimize the true error $\mathcal{E}(\Phi)$ of Φ measured on the probability space $(\Sigma, \mathcal{G}, \mathbb{P})$ with the random vectors $X : \Sigma \to \mathbb{R}^d$ and $Y : \Sigma \to \mathbb{R}$. The pairs $(\boldsymbol{x}_i, y_i)$ are elements of $X(\Sigma) \times Y(\Sigma)$. A definition for the *true error* corresponding to the least squares method can then be written as [29]

$$\mathcal{E}(\Phi) = \int_{X(\Sigma)} \int_{Y(\Sigma)} \mathrm{d}\rho \ \left(\Phi(\boldsymbol{x}) - y\right)^2 ,$$

where ρ is the probability distribution on $X(\Sigma) \times Y(\Sigma)$. Unfortunately the probability distribution is in general unkown, otherwise the minimizing function f_ρ of the error could be calculated by [29]

$$f_\rho = \int_{Y(\Sigma)} \mathrm{d}\rho(y|x) \ y.$$

The probability distribution $\rho(y|\boldsymbol{x})$ is the conditional distribution of y given $\boldsymbol{x}$.

In practice, one needs to make an assumption about the approximation function Φ. This means that one constructs a hypothesis space $\mathcal{H}$ of which Φ should be part of. It is common that $\mathcal{H}$ is chosen as a subset of the Banach space $(\mathcal{C}(\Omega), \|\cdot\|_\infty)$, with $\|f\|_\infty = \sup_{\boldsymbol{x} \in \Omega} |f(\boldsymbol{x})|$, where $\Omega \subset \mathbb{R}^d$ is the domain on which one tries to approximate the function [29]. For example the set of polynomials up to degree n or NNs with fixed architecture are possible

hypothesis spaces. In general, $f_\rho \notin \mathcal{H}$ which motivates the *approximation error* $\mathcal{E}(\Phi_\mathcal{H})$, where [29]

$$\Phi_\mathcal{H} = \arg\min_{\Phi \in \mathcal{H}} \mathcal{E}(\Phi)$$

In the learning problem, the goal is to find the function $\Phi_{\boldsymbol{z}} \in \mathcal{H}$ that minimizes the empirical error

$$\Phi_{\boldsymbol{z}} = \arg\min_{\Phi \in \mathcal{H}} \mathcal{E}_{\boldsymbol{z}}(\Phi).$$

One can partition the true error $\mathcal{E}(\Phi_{\boldsymbol{z}})$ into two terms by defining the *sample error* $\mathcal{E}_\mathcal{H}(\Phi_{\boldsymbol{z}}) = \mathcal{E}(\Phi_{\boldsymbol{z}}) - \mathcal{E}(\Phi_\mathcal{H})$. The approximation error $\mathcal{E}(\Phi_\mathcal{H})$ does not depend on the given data. This error can be reduced by extending $\mathcal{H}$. But while fixing the size m of the data, extending $\mathcal{H}$ usually leads to an increasing sample error $\mathcal{E}_\mathcal{H}(\Phi_{\boldsymbol{z}})$ [29]. This phenomenon is often called "overfitting". In statistical learning theory, this problem is known as the *bias-variance trade-off*.

A word of caution is needed here: the existence of the minimizers $\Phi_\mathcal{H}$ and $\Phi_{\boldsymbol{z}}$ is only guaranteed if $\mathcal{H}$ is a compact subset of $\mathcal{C}(\Omega)$. However, it was proven that under relatively weak assumptions the set of NNs with fixed architecture is not closed in $\mathcal{C}([-B, B]^d), B > 0$ and therefore not a compact subspace. The assumptions apply to all commonly used activation functions except the ReLU and the parametric ReLU. When bounding all weights $\boldsymbol{W}_i$ and $\boldsymbol{b}_i$ of the NNs they do form a compact hypothesis space $\mathcal{H}$ on $\mathcal{C}(K)$ though, with $K \subset \mathbb{R}^d$ being compact [30].

The task of the training of a NN is to find the weights $\boldsymbol{W}_i$ and biases $\boldsymbol{b}_i$ that minimize the empirical error, while fixing the architecture and the activation function. In other words, the goal is to find $\Phi_{\boldsymbol{z}}$. In general, the empirical error is a high-dimensional, non-linear and non-convex function $\mathcal{E}_{\boldsymbol{z}}(\Phi)((\boldsymbol{W}_i, \boldsymbol{b}_i)_{i=1}^L)$ of the weights and biases $(\boldsymbol{W}_i, \boldsymbol{b}_i)_{i=1}^L$, thus making the minimization procedure challenging. If the empirical error is a differentiable function, gradient-based methods are widely used for optimization. One of the most prominent optimization algorithms are the *stochastic gradient descent (SGD)* and its extensions like *Adam* [31]. The gradient of the error function can be calculated efficiently by the *backpropagation* algorithm[3] [32]. However, in this work, we made use of the *multistream extended Kalman filter (EKF)* which will be explained in the next section.

In order to detect possible overfitting, a part of the training data is not used during optimization but only for the calculation of the empirical error on this subset, which is called the test set. A low sample error compared to the approximation error would mean that the empirical error on both sets should be of similar magnitude (provided that both sets are large enough) whereas a notable larger empirical error on the test set indicates overfitting.

2.2.2 Kalman Filter

Linear Kalman Filter

The *Kalman filter* is an algorithm from estimation theory that determines the least-squares solution of a linear dynamical system in a recursive manner. It was introduced by Kalman

[3]First, all neuron values before and after the application of the activation function need to be determined (forward pass). Then, starting from layer L, the derivatives with respect to the weights and biases in the ith layer can be related to intermediate results of the calculation of the derivatives in the $(i+1)$th layer.

in 1960 [33] and has since been adapted in various ways like the non-linear variants called the *extended* [34] or the *unscented* Kalman filter [35].

Consider a vector $\boldsymbol{x}_k \in \mathbb{R}^n$ that completely describes the state of a dynamical system at discrete time k. Let the change in time of the state be described by

$$\boldsymbol{x}_{k+1} = \boldsymbol{F}_{k+1,k}\boldsymbol{x}_k + \boldsymbol{w}_k,$$

where $\boldsymbol{F}_{k+1,k} \in \mathbb{R}^{n\times n}$ is called the *transition matrix* evolving the state from time k to $k+1$ and $\{\boldsymbol{w}_k\}_{k\in\mathbb{N}}$ is an n-dimensional independent gaussian random process with zero mean. Assume that the state $\boldsymbol{x}_k$ of the system cannot be observed directly so one wants to estimate the state by measuring $\boldsymbol{y}_k \in \mathbb{R}^p$ that is related to the state linearly by

$$\boldsymbol{y}_k = \boldsymbol{H}_k\boldsymbol{x}_k + \boldsymbol{v}_k.$$

$\boldsymbol{H}_k \in \mathbb{R}^{p\times n}$ $(p \leq n)$[4] and $\{\boldsymbol{v}_k\}_{k\in\mathbb{N}}$ is a p-dimensional independent gaussian random process with zero mean. The random processes $\{\boldsymbol{w}_k\}_{k\in\mathbb{N}}$ and $\{\boldsymbol{v}_k\}_{k\in\mathbb{N}}$ are further described by

$$\mathbb{E}[\boldsymbol{v}_k\boldsymbol{v}_j^\intercal] = \boldsymbol{R}_k\delta_{kj}, \quad \mathbb{E}[\boldsymbol{w}_k\boldsymbol{w}_j^\intercal] = \boldsymbol{Q}_k\delta_{kj}, \quad \mathbb{E}[\boldsymbol{v}_k\boldsymbol{w}_j^\intercal] = 0 \quad \forall k, j \in \mathbb{N},$$

where δ_{kj} is the Kronecker delta and $\boldsymbol{R}_k \in \mathbb{R}^{p\times p}$ and $\boldsymbol{Q}_k \in \mathbb{R}^{n\times n}$ are the covariance matrices at time k [37].

Given a sequence $\{\boldsymbol{y}_0, \ldots, \boldsymbol{y}_k\}$ the goal is now to find an optimal[5] estimate $\hat{\boldsymbol{x}}_{k/k}$ of $\boldsymbol{x}_k$. Here the notation $\hat{\boldsymbol{x}}_{k/j}$ indicates that $\boldsymbol{x}_k$ is estimated from the data $\{\boldsymbol{y}_0, \ldots, \boldsymbol{y}_j\}$. The Kalman filter solves the problem recursively in the sense that it computes $\hat{\boldsymbol{x}}_{k/k}$ from $\boldsymbol{y}_k$ and $\hat{\boldsymbol{x}}_{k-1/k-1}$. This approach has the advantage of being memory efficient and suitable for real-time estimates. The update step consists of the following system of equations where we introduce the *prior estimate* $\hat{\boldsymbol{x}}_{k/k-1} = \boldsymbol{F}_{k+1,k}\hat{\boldsymbol{x}}_{k-1/k-1}$ [37]:

$$\begin{aligned}
\boldsymbol{P}_{k/k-1} &= \mathbb{E}[(\boldsymbol{x}_k - \hat{\boldsymbol{x}}_{k/k-1})(\boldsymbol{x}_k - \hat{\boldsymbol{x}}_{k/k-1})^\intercal] \\
&= \boldsymbol{F}_{k,k-1}\boldsymbol{P}_{k-1/k-1}\boldsymbol{F}_{k,k-1}^\intercal + \boldsymbol{Q}_{k-1} && (2.2)\\
\boldsymbol{P}_{k/k} &= \mathbb{E}[(\boldsymbol{x}_k - \hat{\boldsymbol{x}}_{k/k})(\boldsymbol{x}_k - \hat{\boldsymbol{x}}_{k/k})^\intercal] \\
&= \boldsymbol{P}_{k/k-1} - \boldsymbol{K}_k\boldsymbol{H}_k\boldsymbol{P}_{k/k-1} && (2.3)\\
\boldsymbol{K}_k &= \boldsymbol{P}_{k/k-1}\boldsymbol{H}_k^\intercal(\boldsymbol{H}_k\boldsymbol{P}_{k/k-1}\boldsymbol{H}_k^\intercal + \boldsymbol{R}_k)^{-1} && (2.4)\\
\hat{\boldsymbol{x}}_{k/k} &= \underbrace{\boldsymbol{F}_{k,k-1}\hat{\boldsymbol{x}}_{k-1/k-1}}_{\hat{\boldsymbol{x}}_{k/k-1}} + \boldsymbol{K}_k[\boldsymbol{y}_k - \boldsymbol{H}_k\boldsymbol{F}_{k,k-1}\hat{\boldsymbol{x}}_{k-1/k-1}]. && (2.5)
\end{aligned}$$

$\boldsymbol{P}_{k/k-1}$ is the covariance matrix of the error of the prior estimate $\hat{\boldsymbol{x}}_{k/k-1}$ and $\boldsymbol{P}_{k/k}$ the one of the final estimate $\hat{\boldsymbol{x}}_{k/k}$. $\boldsymbol{K}_k$ is called the *gain matrix*. It is chosen in such a way that it minimizes the expected error

$$\mathbb{E}[(\boldsymbol{x}_k - \hat{\boldsymbol{x}}_{k/k})^\intercal(\boldsymbol{x}_k - \hat{\boldsymbol{x}}_{k/k})],$$

which is the trace of $\boldsymbol{P}_{k/k}$. The right-hand side of eq. (2.5) consists of two terms, the first one is the best prior estimate based on the previous estimate and the second term is a correction that is weighted proportional to its inducing error with respect to the new measurement. In order to start at $k = 0$ the Kalman filter needs the values of $\hat{\boldsymbol{x}}_{0/-1}$ and $\boldsymbol{P}_{0/-1}$ to be

[4] This condition is given in the original paper by Kalman [33] but deliberately removed in [36]

[5] There are various meaningful definitions

determined additionally [37]. A word of caution needs to be mentioned regarding eq. (2.4): The invertibility of the term in parentheses is not guaranteed. $\boldsymbol{H}_k \boldsymbol{P}_{k/k-1} \boldsymbol{H}_k^\intercal + \boldsymbol{R}_k$ is in general only positive semidefinite. However, if e.g. $\boldsymbol{R}_k$ is positive definite then also the sum of both terms is positive definite and that is a sufficient criterion for invertibility. As we will see later $\boldsymbol{R}_k$ is usually positive definite in the context of NN training.

Extended Kalman Filter

The *EKF* is an algorithm that adapts the Kalman filter to non-linear systems and can be described by

$$\begin{aligned} \boldsymbol{x}_{k+1} &= \boldsymbol{f}_{k+1,k}(\boldsymbol{x}_k) + \boldsymbol{w}_k \\ \boldsymbol{y}_k &= \boldsymbol{h}_k(\boldsymbol{x}_k) + \boldsymbol{v}_k \end{aligned} \tag{2.6}$$

The random processes $\{\boldsymbol{w}\}_{k\in\mathbb{N}}$ and $\{\boldsymbol{v}\}_{k\in\mathbb{N}}$ are defined in the same way as before but instead of the matrices $\boldsymbol{F}_{k+1,k}$ and $\boldsymbol{H}_k$ the non-linear functions $\boldsymbol{f}_{k+1,k}$ and $\boldsymbol{h}_k$ are introduced. Following the derivation of the Kalman filter in the non-linear case does not work since the linearity property of the (conditional) expectation cannot be used [38]. To be more precise, for a non-linear function g

$$\mathbb{E}[g(X)] = \int_{X(\Sigma)} \mathrm{d}\rho \; g(X) \neq g(\mathbb{E}[X]). \tag{2.7}$$

X, Σ and ρ are defined as in section 2.2.1. In practical applications the probability density ρ may be unkown or the integral in eq. (2.7) may be computationally too expensive. In the linear case only $\hat{\boldsymbol{x}}_{0/-1}$ and $\boldsymbol{P}_{0/-1}$ need to be known but not the actual distribution of $\boldsymbol{x}_0$.

In order to circumvent the problem the idea of the EKF is that the state $\boldsymbol{x}_k$ is hopefully close to the last estimate $\hat{\boldsymbol{x}}_{k/k}$ and one can expand $\boldsymbol{f}_{k+1,k}$ and $\boldsymbol{h}_k$ in a Taylor series up to first order around this point. Thus, one obtains a linearized approximation of the state dynamics and the measurement process and can apply the Kalman filter. By defining the matrices $\boldsymbol{F}_{k+1,k}$ and $\boldsymbol{H}_k$ with entries

$$\begin{aligned} (\boldsymbol{F}_{k+1,k})_{ij} &= \left.\frac{\partial (\boldsymbol{f}_{k+1,k})_i(\boldsymbol{x})}{\partial x_j}\right|_{\boldsymbol{x}=\hat{\boldsymbol{x}}_{k/k}} \\ (\boldsymbol{H}_k)_{ij} &= \left.\frac{\partial (\boldsymbol{h}_k)_i(\boldsymbol{x})}{\partial x_j}\right|_{\boldsymbol{x}=\hat{\boldsymbol{x}}_{k/k-1}}, \end{aligned}$$

where $(\boldsymbol{x})_i = x_i$, one obtains the extended Kalman filter equations [38, 39]

$$\boldsymbol{P}_{k/k-1} = \boldsymbol{F}_{k,k-1} \boldsymbol{P}_{k-1/k-1} \boldsymbol{F}_{k,k-1}^\intercal + \boldsymbol{Q}_{k-1} \tag{2.8}$$

$$\boldsymbol{P}_{k/k} = \boldsymbol{P}_{k/k-1} - \boldsymbol{K}_k \boldsymbol{H}_k \boldsymbol{P}_{k/k-1} \tag{2.9}$$

$$\boldsymbol{K}_k = \boldsymbol{P}_{k/k-1} \boldsymbol{H}_k^\intercal (\boldsymbol{H}_k \boldsymbol{P}_{k/k-1} \boldsymbol{H}_k^\intercal + \boldsymbol{R}_k)^{-1} \tag{2.10}$$

$$\hat{\boldsymbol{x}}_{k/k} = \underbrace{\boldsymbol{f}_{k,k-1}(\hat{\boldsymbol{x}}_{k-1/k-1})}_{\hat{\boldsymbol{x}}_{k/k-1}} + \boldsymbol{K}_k [\boldsymbol{y}_k - \boldsymbol{h}_k(\boldsymbol{f}_{k,k-1}(\hat{\boldsymbol{x}}_{k-1/k-1}))]. \tag{2.11}$$

In fact eqs. (2.8) to (2.10) of the EKF closely resemble the corresponding eqs. (2.2) to (2.4) of the classical Kalman filter. Note that because of the approximation used in the EKF, $\hat{\boldsymbol{x}}_{k/k}$ is, in general, not the optimal estimate of the current state $\boldsymbol{x}_k$. Nevertheless, this filter has been applied in various non-linear problems successfully, especially in GPS and navigation tasks.

Extended Kalman Filter as a Training Algorithm

The EKF can be applied to NN training by considering the vector of optimal weights and biases $(\boldsymbol{W}_i, \boldsymbol{b}_i)_{i=1}^{L}$ as the state of the system. Here we mostly follow the approach demonstrated by Singhal and Wu [40] and Iiguni et al. [41] among others. A comprehensive summary can be found in Haykin [39]. Out of notational convenience, we will abbreviate the weights and biases in the following by $\boldsymbol{\theta} = (\boldsymbol{W}_i, \boldsymbol{b}_i)_{i=1}^{L}$. By denoting the kth input vector of the NN by $\boldsymbol{z}_k$ and the corresponding target value by $\boldsymbol{y}_k$ we can describe the training of the NN as a special case of eq. (2.6) [39, 41, 42]

$$\begin{aligned} \boldsymbol{\theta}_{k+1} &= \boldsymbol{\theta}_k + \boldsymbol{w}_k \\ \boldsymbol{y}_k &= \boldsymbol{\Phi}(\boldsymbol{\theta}_k; \boldsymbol{z}_k) + \boldsymbol{v}_k. \end{aligned} \tag{2.12}$$

Here, $\boldsymbol{w}_k$[6] and $\boldsymbol{v}_k$ represent the same kind of noise as in eq. (2.6) and $\boldsymbol{\Phi}(\boldsymbol{\theta}_k; \boldsymbol{z}_k)$ is the NN with weights and biases $\boldsymbol{\theta}_k$ evaluated at $\boldsymbol{z}_k$. Since these equations describe a static process only the measurement process, i.e. the NN $\boldsymbol{\Phi}$, needs to be linearized

$$(\boldsymbol{H}_k)_{ij} = \left.\frac{\partial(\boldsymbol{\Phi})_i(\boldsymbol{\theta}; \boldsymbol{z}_k)}{\partial \theta_j}\right|_{\boldsymbol{\theta}=\hat{\boldsymbol{\theta}}_{\boldsymbol{k/k-1}}}.$$

This matrix can be computed efficiently by using the backpropagation algorithm [32]. Since $\boldsymbol{f}_{k+1,k}$ is the identity map we can simplify the notation by abbreviating $\hat{\boldsymbol{\theta}}_{k-1} \coloneqq \hat{\boldsymbol{\theta}}_{k-1/k-1} = \hat{\boldsymbol{\theta}}_{k/k-1}$ and $\boldsymbol{P}_k \coloneqq \boldsymbol{P}_{k/k-1}$[7]. Hence, we obtain a special case of eqs. (2.8) to (2.11) [39]

$$\boldsymbol{P}_k = \boldsymbol{P}_{k-1} - \boldsymbol{K}_{k-1}\boldsymbol{H}_{k-1}\boldsymbol{P}_{k-1} + \boldsymbol{Q}_{k-1} \tag{2.13}$$

$$\boldsymbol{K}_k = \boldsymbol{P}_k\boldsymbol{H}_k^{\intercal}(\boldsymbol{H}_k\boldsymbol{P}_k\boldsymbol{H}_k^{\intercal} + \boldsymbol{R}_k)^{-1} \tag{2.14}$$

$$\hat{\boldsymbol{\theta}}_k = \hat{\boldsymbol{\theta}}_{k-1} + \boldsymbol{K}_k[\boldsymbol{y}_k - \boldsymbol{\Phi}(\hat{\boldsymbol{\theta}}_{k-1}; \boldsymbol{z}_k)]. \tag{2.15}$$

If we consider the linearized version of eq. (2.12) the Kalman gain is chosen in such a way that the expected error with respect to the state $\mathbb{E}[(\boldsymbol{\theta}_k - \hat{\boldsymbol{\theta}}_k)^{\intercal}(\boldsymbol{\theta}_k - \hat{\boldsymbol{\theta}}_k)]$ is minimized as it was mentioned in section 2.2.2. By taking the derivative of this error with respect to $\boldsymbol{K}_k$ one obtains an expression that is not explicitly dependent on $\boldsymbol{\theta}_k$ but only on $\mathbb{E}[\boldsymbol{\theta}_k]$. Since we are considering a static system this term is independent of k and thus $\hat{\boldsymbol{\theta}}_k$ is not only the new estimate for $\boldsymbol{\theta}_k$ but for any $\boldsymbol{\theta}_i$, $i \in \mathbb{N}$ at time k. Introducing the error vector $\boldsymbol{\xi}_{ij} = \boldsymbol{y}_i - \boldsymbol{\Phi}(\hat{\boldsymbol{\theta}}_j; \boldsymbol{z}_i)$ and the matrix

$$(\boldsymbol{H}_{ij})_{kl} = \left.\frac{\partial(\boldsymbol{\Phi})_k(\boldsymbol{\theta}; \boldsymbol{z}_i)}{\partial \theta_l}\right|_{\boldsymbol{\theta}=\hat{\boldsymbol{\theta}}_{\boldsymbol{j}}}$$

one can approximate

$$\boldsymbol{\xi}_{ij} = \boldsymbol{H}_{ij}(\boldsymbol{\theta}_i - \hat{\boldsymbol{\theta}}_j) + \boldsymbol{v}_i + \mathcal{O}\left((\boldsymbol{\theta}_i - \hat{\boldsymbol{\theta}}_j)^2\right).$$

From this, it is straightforward to derive that minimizing $\mathbb{E}[(\boldsymbol{\theta}_i - \hat{\boldsymbol{\theta}}_k)^{\intercal}(\boldsymbol{\theta}_i - \hat{\boldsymbol{\theta}}_k)]$ also minimizes the expected error $\mathbb{E}[\boldsymbol{\xi}_{ik}^{\intercal}\boldsymbol{\xi}_{ik}]$ with respect to $\boldsymbol{K}_k$[8]. Thus, each update simultaneously

[6] The measurement noise is not always included in NN training, e.g. see [40, 41].

[7] This convention is chosen in such a way that it matches the notation in [39].

[8] This follows from $\boldsymbol{H}_{ik}^{\intercal}\boldsymbol{H}_{ik}$ being symmetric, $\frac{\partial \hat{\theta}_k^j}{\partial K_k^{il}} \propto \delta^{ij}$ and that the derivatives for all i, l have to be zero.

minimizes

$$\mathbb{E}\left[\frac{1}{k+1}\sum_{i=0}^{k}\boldsymbol{\xi}_{ik}^{\intercal}\boldsymbol{\xi}_{ik}\right]. \tag{2.16}$$

In section 2.2.1 we mentioned that the goal is to minimize the empirical error with respect to $\boldsymbol{\theta}$. Evidently, the EKF does not exactly do that in general. Instead, it minimizes the expectation of the empirical error by making assumptions about the noise in the state and the measurement decoded by the covariance matrices $\boldsymbol{Q}_k$ and $\boldsymbol{R}_k$, respectively. In fact, the noise in the state $\boldsymbol{\theta}_k$ is an assumption that usually does not appear in other optimization algorithms like the SGD. In practice, the minimization of eq. (2.16) is not guaranteed at all for several reasons. Firstly, the derivation only holds for the linear approximation which assumes $\|\boldsymbol{\theta}_k - \hat{\boldsymbol{\theta}}_{k-1}\|$ to be small. This leads us to the second problem. The initial estimator $\hat{\boldsymbol{\theta}}_{-1}$ already needs to be close to $\boldsymbol{\theta}_0$ and its covariance matrix $\boldsymbol{P}_0$ needs to be known. Lastly, the Gaussian noise terms introduced by the EKF are not rigorously justified but rather a rough approximation of the process leading to the question of how to choose $\boldsymbol{Q}_k$ and $\boldsymbol{R}_k$.

As a result, those variables are rather used as tuning parameters. One common way to initialize the NN weights and biases $\hat{\boldsymbol{\theta}}_{-1}$ is drawing each of them from a random uniform distribution with zero variance. The corresponding covariance matrix is then typically initialized with $\boldsymbol{P}_0 = \frac{1}{\varepsilon}\mathbb{1}$, $\varepsilon > 0$. Similarly, the covariance matrix of the measurement noise is often chosen as $\boldsymbol{R}_k = \frac{1}{\eta_k}\mathbb{1}$, $\eta_k > 0$ [39]. A larger value of η_k can be interpreted in the sense that the measurement of $\boldsymbol{y}_k$ is more certain and as a result, more emphasis is put on the update at time step k. This is why it is called *learning rate*. Note that with this choice of $\boldsymbol{R}_k$ the matrix inversion appearing in the calculation of the Kalman gain is possible as discussed before. A common scheme is to start with a small learning rate and increase it with the number of updates. Similarly, the covariance matrix describing the process noise is often set as $\boldsymbol{Q}_k = q_k\mathbb{1}$, $q_k \geq 0$. The values for q_k usually decrease with the number of updates [39]. As Haykin [39] pointed out the matrices $\boldsymbol{P}_k, \boldsymbol{R}_k$ and $\boldsymbol{Q}_k$ can be rescaled by a factor μ without changing the equations of the EKF. This means that the performance of the EKF cannot be tuned by ε, η_k and q_k independently. For a particular choice of those parameters we refer the reader to the designated sections in the work of Singraber and Morawietz [16] and Haykin [39].

Multistream Extended Kalman Filter

The aforementioned approximations of the EKF training algorithm lead to a problem called the *recency effect*. The update steps tend to be biased in favor of the errors with respect to more recent training data [43]. This is an artifact caused by the linearization procedure since the "quality" of the update steps of the EKF is heavily dependent on the accuracy of the approximation at each step. Intuitively one can think of the following: Assuming the EKF training reduces the error $\|\boldsymbol{\theta}_k - \hat{\boldsymbol{\theta}}_k\|$ with increasing k, the linear approximation underlying each successive update becomes more accurate in finding the optimal estimator. Each update should incorporate the information of the training data that is given at that time step but bad initial guesses lead to inaccurate updates in the beginning.

The *multistream EKF* is an approach that was designed to reduce the bias of the most recent update. It was introduced by Feldkamp and Puskorius [44] who were inspired by the batch update method found e.g. in the SGD algorithm. The idea is that instead of considering only a single pair $(\boldsymbol{z}, \boldsymbol{y})$ of the training data at each update step, one incorporates the information of M pairs $(\boldsymbol{z}_i, \boldsymbol{y}_i)_{i=1}^{M}$ of the training data in a single update. In order to still

be consistent with the equations of the EKF one does not consider the actual NN $\boldsymbol{\Phi}(\hat{\boldsymbol{\theta}}_{k-1}; \boldsymbol{z})$. Let $\boldsymbol{z}_i \in \mathbb{R}^d$ and $\boldsymbol{y}_i \in \mathbb{R}^{N_L}$. Then we introduce the function

$$\begin{aligned} \boldsymbol{\Psi}: \quad & \mathbb{R}^{M\times d} \to \mathbb{R}^{M\times N_L} \\ & (\boldsymbol{z}_1, \ldots, \boldsymbol{z}_M) \mapsto (\boldsymbol{\Phi}(\hat{\boldsymbol{\theta}}_{k-1}; \boldsymbol{z}_1), \ldots, \boldsymbol{\Phi}(\hat{\boldsymbol{\theta}}_{k-1}; \boldsymbol{z}_M)) \end{aligned} \tag{2.17}$$

with the corresponding target values $(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_M)$. The EKF eqs. (2.13) and (2.14) can then be applied by redefining

$$(\boldsymbol{H}_k)_{ij} = \left.\frac{\partial(\boldsymbol{\Psi})_i(\boldsymbol{\theta}; (\boldsymbol{z}_1, \ldots, \boldsymbol{z}_M))}{\partial \theta_j}\right|_{\boldsymbol{\theta}=\hat{\boldsymbol{\theta}}_{\boldsymbol{k-1}}}.$$

and eq. (2.15) needs to be replaced by

$$\hat{\boldsymbol{\theta}}_k = \hat{\boldsymbol{\theta}}_{k-1} + \boldsymbol{K}_k[(\boldsymbol{y}_1, \ldots, \boldsymbol{y}_M) - \boldsymbol{\Psi}(\hat{\boldsymbol{\theta}}_{k-1}; (\boldsymbol{z}_1, \ldots, \boldsymbol{z}_M))].$$

Several case studies [16, 43, 45] showed that the convergence rate of the optimization with respect to the number of updates improved when deploying the multistream variant of the EKF.

Performance Considerations

It is well known that the EKF features a superior convergence rate with respect to the number of updates than other widely used optimization algorithms that rely on gradient-based methods like the SGD or Adam [16, 40, 41]. In practice, however, it is of little interest how many update steps are needed to reach a certain error threshold. Rather one is interested in the computational time that passes. That is why the EKF may be unappealing for many scenarios since the computational complexity of a single update step is $\mathcal{O}(N_\theta^2)$[9], where

$$N_\theta = \sum_{i=1}^{L} N_i(N_{i-1} + 1) \tag{2.18}$$

denotes the number of weights and biases and $N_0 = d$. In this work the size of the NN is small enough that the EKF converges much quicker in time than other common approaches but in many areas much larger NNs are deployed which makes the EKF impracticable. There exist modifications to the EKF which try to improve the scaling of the algorithm, e.g. the decoupled EKF [46]. Those modifications come with their downsides though and have not been used in this work.

The multistream EKF offers better convergence with respect to the updates at the cost of higher computational effort for each update due to M evaluations of the NN and the computation of $\boldsymbol{H}_k$ which consists of M times more entries than the original method. However, these additional computational steps can easily be parallelized. Singraber and Morawietz [16] showed an approach that completely parallelizes the evaluations of $\boldsymbol{\Phi}(\hat{\boldsymbol{\theta}}_{k-1}; \boldsymbol{z}_i)$ and the calculation of $\boldsymbol{H}_k$. The only serial step consists of the actual evaluations of the EKF equations which scale with $\mathcal{O}(M^3)$. The convergence improvement with additional streams quickly saturates whereas the computational costs of the serial step still increase [45]. So even when ignoring the overhead caused by parallelization there is a maximum number of streams M that yields actual performance improvements.

[9] Assuming naive matrix multiplication

2.3 Potential Energy Surface

Definition 2.3 (Potential Energy Surface)**.** A potential energy surface is a function $U : M \to \mathbb{R}$ that maps each point $p \in M$ on a differentiable manifold M to a real value that is the potential energy of a (quantum) mechanical system associated with the particular configuration represented by p.

Usually, M is the configuration space or a submanifold of it. In case of a Newtonian theory $M = \mathbb{R}^{3N}$ and describes the positions of N mass points, typically corresponding to atoms [47]. Obviously the potential energy surface (PES) is only well-defined if the potential energy U is only a function of the positions of the particles. In MD one tries to model the interaction of particles by means of the particular choice of $U(\boldsymbol{r}^{3N})$, where

$$\boldsymbol{r}^{3N} = \begin{pmatrix} \boldsymbol{r}_1 \\ \boldsymbol{r}_2 \\ \vdots \\ \boldsymbol{r}_N \end{pmatrix}$$

is the column vector of all positions of the particles. In the following, we will often write this vector sloppily as $(\boldsymbol{r}_1, \ldots, \boldsymbol{r}_N)$. $U(\boldsymbol{r}^{3N})$ is then used to update the positions of each particle by integrating the equations of motion according to the forces $\boldsymbol{F}_i = -\boldsymbol{\nabla}_i U$, where $\boldsymbol{F}_i$ is the force acting on particle i. Since the forces have to be calculated for N particles during each time integration step, an efficient evaluation of U is crucial for large systems and long simulation times. This makes ab initio models, e.g. based on DFT unattractive, since the computational costs are significant for today's computational resources.

The idea of NNPs is to use NNs for the approximation of the costly potential energy surface U since NNs can in general be evaluated efficiently. Assuming a small approximation error this method allows combining the accuracy and transferability of ab initio potentials with the efficiency of NNs. In order to be applicable for MD one also needs a good approximation of the derivatives of the potential energy surface. In fact, it was shown that under relatively weak conditions, this is guaranteed.

Definition 2.4 ([48])**.** Let $l \in \mathbb{N}_{\geq 0}$ and $f \in \mathcal{C}^l(\mathbb{R})$. f is called *l-finite* if

$$0 < \int_{\mathbb{R}} \mathrm{d}x \, |f^{(l)}(x)| < \infty,$$

where $f^{(l)}$ denotes the l^{th} derivative of f.

If σ is l-finite, the NN can simultaneously approximate any smooth and rapidly decreasing function and its derivatives up to order l arbitrarily well on a compact subset [48]. And since NNs are expressed analytically as seen in eq. (2.1) one can also obtain an analytic expression for the derivative straightforwardly.

A naive approach to a NNP would be the definition

$$\begin{aligned} \Phi : \mathbb{R}^{3N} &\to \mathbb{R} \\ \boldsymbol{r}^{3N} &\mapsto \Phi(\boldsymbol{r}^{3N}) \end{aligned}$$

combined with a training algorithm that minimizes

$$\mathcal{E}_{\boldsymbol{z}}(\Phi) = \frac{1}{m}\sum_{i=1}^{m}(\Phi(\boldsymbol{r}_i^{3N}) - U(\boldsymbol{r}_i^{3N}))^2.$$

Here, the training data would be given in the form $(\boldsymbol{r}_i^{3N}, U(\boldsymbol{r}_i^{3N}))_{i=1}^m$. There are two downsides to this choice. Firstly, this NNP is limited to a maximum of N atoms since the topology of a NN is fixed so it cannot be applied to the full configuration space of a larger system. Besides that, it also is not obvious how to apply such a NNP to a system with fewer than N atoms. So in practice, one would need individual NNPs for different system sizes. The second disadvantage of this approach is that there is no guarantee that physical symmetries are preserved. This will be discussed in the next subsection.

2.3.1 Symmetries of the Potential Energy Surface

The potential energy should be independent of the choice of coordinate system. This implies that it should be invariant under a global translation $T_{\boldsymbol{a}}\boldsymbol{r}^{3N} = (\boldsymbol{r}_1 + \boldsymbol{a}, \ldots, \boldsymbol{r}_N + \boldsymbol{a})$, where $\boldsymbol{a} \in \mathbb{R}^3$ or under a global rotation $R\boldsymbol{r}^{3N} = (\boldsymbol{R}\boldsymbol{r}_1, \ldots, \boldsymbol{R}\boldsymbol{r}_N)$, where $\boldsymbol{R} \in \mathrm{O}(3)$[10]. Furthermore, if multiple atoms only differ in their position the potential energy should not depend on the order of these atom's positions in the vector $\boldsymbol{r}^{3N}$. Let $i \sim_P j \Leftrightarrow$ (atom i has same properties as atom j) be an equivalence relation. Then $[j] \subseteq \{1, \ldots N\}$ is an equivalence class of all atoms that only differ by their position. From now on we will say that these atoms are of the same type. In practice, this usually means that they are of the same element. We can define an arbitrary permutation only acting on atoms of the type $[j]$

$$\begin{aligned}\sigma_{[j]}: \quad & \mathbb{N}^N \to \mathbb{N}^N \\ & \sigma_{[j]}(i) \mapsto \begin{cases}\sigma(i) & \text{if } i \in [j] \\ i & \text{otherwise,}\end{cases}\end{aligned}$$

where σ defines an arbitrary permutation on the set $[j]$. Thus we can formally write the permutation symmetry of the potential energy for an arbitrary j like the following

$$U\left((\boldsymbol{r}_1, \ldots, \boldsymbol{r}_N)\right) = U\left((\boldsymbol{r}_{\sigma_{[j]}(1)}, \ldots, \boldsymbol{r}_{\sigma_{[j]}(N)})\right)$$

Out of notational convenience, we can combine the permutations of different types to a single map $\sigma_{\mathrm{T}} = \sigma_{i_1} \circ \cdots \circ \sigma_{i_{N_{\mathrm{Types}}}}$, where the $i_j \in \{1, \ldots, N\}/\sim_P$ enumerate the permutations of all N_{Types} atom types. It is straightforward to extend each of the aforementioned symmetry operations to a group. In fact, we can even combine these operations into a single symmetry group, in this work called CS, acting on the configuration space. We write the elements of this group as $S(\boldsymbol{R}, \boldsymbol{a}, \sigma_{\mathrm{T}})$ with the group action

$$S(\boldsymbol{R}, \boldsymbol{a}, \sigma_{\mathrm{T}})\boldsymbol{r}^{3N} = \left(\boldsymbol{R}\boldsymbol{r}_{\sigma_{\mathrm{T}}(1)} + \boldsymbol{a}, \ldots, \boldsymbol{R}\boldsymbol{r}_{\sigma_{\mathrm{T}}(N)} + \boldsymbol{a}\right)$$

and the group composition

$$S(\boldsymbol{R}', \boldsymbol{a}', \sigma_{\mathrm{T}}')S(\boldsymbol{R}, \boldsymbol{a}, \sigma_{\mathrm{T}}) = S(\boldsymbol{R}'\boldsymbol{R}, \boldsymbol{R}'\boldsymbol{a} + \boldsymbol{a}', \sigma_{\mathrm{T}}' \circ \sigma_{\mathrm{T}}).$$

[10] The discussion is often restricted to $\boldsymbol{R} \in SO(3)$, but in practice, reflection symmetry is also guaranteed as we will see later

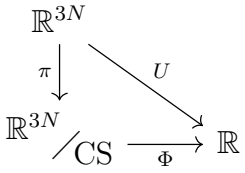
For the sake of completeness, we mention the inverse element

$$S(\boldsymbol{R}, \boldsymbol{a}, \sigma_{\mathrm{T}})^{-1} = S(\boldsymbol{R}^{\intercal}, -\boldsymbol{R}^{\intercal}\boldsymbol{a}, \sigma_{\mathrm{T}}^{-1})$$

and the identity element of this group $e = S(\mathbb{1}_3, \mathbf{0}, \mathrm{id}_{\mathbb{N}^N})$. Elements of the group CS acting on $\mathbb{R}^{3N}$ induce an equivalence relation

$$\boldsymbol{r}^{3N} \sim_{\mathrm{CS}} \boldsymbol{r}'^{3N} \Leftrightarrow \exists S(\boldsymbol{R}, \boldsymbol{a}, \sigma_{\mathrm{T}}) \in \mathrm{CS}(N_{\mathrm{Types}}) : S(\boldsymbol{R}, \boldsymbol{a}, \sigma_{\mathrm{T}})\boldsymbol{r}^{3N} = \boldsymbol{r}'^{3N}.$$

The corresponding equivalence classes are the orbits of $\mathbb{R}^{3N}$ under the action of CS, formally written as $\mathbb{R}^{3N}/_{\mathrm{CS}}$. Let π be the canonical projection $\pi : \mathbb{R}^{3N} \to \mathbb{R}^{3N}/_{\mathrm{CS}}$. Then there should exist a function $\Phi : \mathbb{R}^{3N}/_{\mathrm{CS}} \to \mathbb{R}$ such that the following diagram commutes.



2.4 High-Dimensional Neural Network Potentials

In section 2.3, we already touched lightly on the problem of defining NNs directly on the configuration space. Arguably the biggest issue is the restriction to a fixed system size. However, in many practical applications high-dimensional configuration spaces and a flexible number of atoms are of high interest. *HDNNPs* introduced by Behler and Parrinello [8] are a successful approach to overcome this limitation. This method will be explained in the next subsections in detail and we will also discuss its recent extension which was the starting point of this work.

2.4.1 Total Energy Partitioning

High-dimensional systems do not only pose a problem for the creation of NNPs but also for the ab initio electronic structure calculations which are used to generate the reference data necessary for the training of the NNs, since these methods tend to scale badly with increasing system size. Therefore, lots of effort went into developing linearly scaling algorithms which in turn led to the question whether electronic structure calculations of the complete system can be split into calculations on loosely connected smaller subsystems. In particular Prodan and Kohn [49] showed that the change of the electron density resulting from a local perturbation in the potential quickly converges to zero with increasing distance to the perturbed area. This phenomenon was coined "nearsightedness" by Kohn. Another interesting observation was made by Yang [51] who showed that instead of computing the electron density and the total energy of the system by solving the global Kohn-Sham equations [52] one can instead approximately partition the electron density into local contributions which in turn are determined by local projections of the Kohn-Sham Hamiltonian. Inspired by these results an ansatz to overcome the limitation in system size a NN defined on the configuration space would implicate, is to split the total energy of a system into local contributions.

Behler and Parrinello [8] decided to represent the potential energy as

$$U = \sum_{i=1}^{N} U_i \tag{2.19}$$

where $U_i : \mathbb{R}^{3N} \to \mathbb{R}$ is an artificial potential energy contribution associated with the local neighborhood of atom i and N represents the total number of atoms in the system. To date, there is no knowledge that this partitioning can be made sense of physically, and it should be considered merely as a rather arbitrary algorithmic choice. However, as we have discussed before the concept of local contributions is justified in electronic structure calculations which may at least motivate eq. (2.19). Further, this partitioning was also used successfully by other ML approaches [3, 53]

The locality of U_i is easy to characterize in the standard coordinates $\boldsymbol{r}^{3N} = (\boldsymbol{r}_1, \ldots, \boldsymbol{r}_N)^\intercal$.

Definition 2.5 (Local Function)**.** We say that a function $\boldsymbol{f} : \mathbb{R}^{3N} \to \mathbb{R}^l$ with coordinates $(\boldsymbol{r}_1, \ldots, \boldsymbol{r}_N)$ is *local* at $\boldsymbol{r}_i$ if there exists a $0 < R < \infty$ such that for any $\boldsymbol{r}_j, \boldsymbol{\Delta r} \in \mathbb{R}^3$ the following holds: if $\boldsymbol{r}_j, (\boldsymbol{r}_j + \boldsymbol{\Delta r}) \notin B_R(\boldsymbol{r}_i)$ then $\boldsymbol{f}(\boldsymbol{r}_1, \ldots, \boldsymbol{r}_j, \ldots, \boldsymbol{r}_N) = \boldsymbol{f}(\boldsymbol{r}_1, \ldots, \boldsymbol{r}_j + \boldsymbol{\Delta r}, \ldots, \boldsymbol{r}_N)$.

$B_R(\boldsymbol{r_i})$ denotes the open ball with radius R centered at $\boldsymbol{r}_i$. The next step in Behler and Parrinello's approach is to approximate the local potential contributions by $\tilde{U}_i = \Phi_{[i]} \circ \boldsymbol{G}_i$. Here we have introduced the functions $\boldsymbol{G}_i : \mathbb{R}^{3N} \to \mathbb{R}^d$ which map points in an arbitrary high-dimensional[11] configuration space to a d-dimensional space which is needed for the NN $\Phi_{[i]}$ with d input nodes. The advantage of this approach is that the NN needs to approximate a function that is only dependent on the local environment and does not incorporate information about the global system. Therefore the NN is not specific to the system size anymore. The locality of the approximation $\Phi_{[i]} \circ \boldsymbol{G}_i$ is satisfied if it is imposed on $\boldsymbol{G}_i$. In section 2.3.1 we have already used the subscript $[i]$ which represents the equivalence class to which atom i belongs. In this work, it will represent the element of the atom which means we will only employ N_{el} distinct NNs, where N_{el} is the number of elements in the system. The resulting NNP can then be written as

$$\tilde{U} = \sum_{i=1}^{N} \Phi_{[i]} \circ \boldsymbol{G}_i. \tag{2.20}$$

A diagram of the architecture of the 2G NNP is shown in fig. 2.1. The particular choice of $\boldsymbol{G}_i$ is non-trivial and has great impact on the performance of this approach. It will be discussed in the next subsection.

The force $\boldsymbol{F}_i$ acting on atom i can then be computed by $\boldsymbol{F}_i = -\boldsymbol{\nabla}_{\boldsymbol{r}_i} \tilde{U}$. For this, we require that $\Phi_{[i]}$ and $\boldsymbol{G}_i$ are in $\mathcal{C}^1$. Thus the popular ReLU is not a suitable activation function. Even when deciding in the implementation to impose a derivative of 0 or 1 of the ReLU at 0, like it is sometimes done, it will lead to simulating non-smooth unphysical forces.

NNs corresponding to different elements can in general have a different architecture and will most likely have different weights and parameters $\boldsymbol{\theta}_{[i]}$. Out of notational convenience we introduce the vector $\boldsymbol{\Theta} = (\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_{N_{\text{el}}})^\intercal$ that contains all weights and biases of all NNs. The extension of the training algorithm to the NNP is straightforward. The training patterns

[11] Formally it is clear that changing the number of atoms must change the form of the functions $\boldsymbol{G}_i$ but we will see that they are designed in a manner that can be adjusted straightforwardly to a particular N

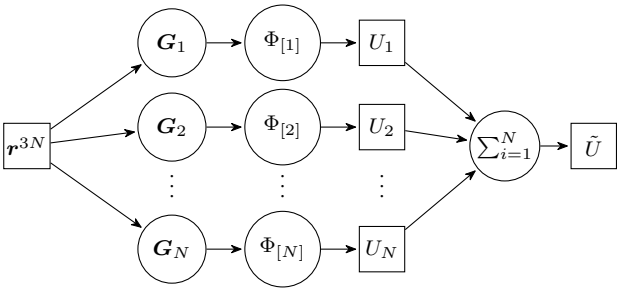


Figure 2.1: 2G HDNNP architecture. Physical quantities are drawn as rectangles, whereas maps are drawn as circles. Individual atomic energies are computed from the configuration $\boldsymbol{r}^{3N}$ and summed up in the end to the total potential energy $\tilde{U}$.

are given in the form $(\boldsymbol{r}_i^{3N}, U(\boldsymbol{r}_i^{3N}))_{i=1}^m$ where $U(\boldsymbol{r}_i^{3N})$ is computed by a suitable reference method. The derivatives with respect to the weights $\frac{\partial \tilde{U}}{\partial \theta_i}$ are easy to calculate and hence one of the usual optimization algorithms can be applied to minimize the error

$$\frac{1}{m}\sum_{i=1}^{m}(U(\boldsymbol{r}_i^{3N}) - \tilde{U}(\boldsymbol{\Theta}; \boldsymbol{r}_i^{3N}))^2$$

with respect to $\boldsymbol{\Theta}$.

However, since one can derive forces from $\tilde{U}$, those can be used as well for the training procedure and hence improve the accuracy of the forces predicted by the NNP. To fit energies and forces simultaneously they need to be coupled by a constant β with dimensions $\frac{[\text{energy}]}{[\text{force}]}$ such that the empirical error can be expressed as

$$\frac{1}{m_U}\sum_{i=1}^{m_U}(U(\boldsymbol{r}_i^{3N}) - \tilde{U}(\boldsymbol{\Theta}; \boldsymbol{r}_i^{3N}))^2 + \frac{\beta^2}{m_F}\sum_{\alpha\in I}\left(\left[\boldsymbol{F}_{\alpha_\mathrm{a}}(\boldsymbol{r}_{\alpha_\mathrm{s}}^{3N}) + \boldsymbol{\nabla}_{\alpha_\mathrm{a}}\tilde{U}(\boldsymbol{\Theta}; \boldsymbol{r}_{\alpha_\mathrm{s}}^{3N})\right]^{\alpha_\mathrm{c}}\right)^2. \quad (2.21)$$

The multi-index $\alpha = (\alpha_\mathrm{s}, \alpha_\mathrm{a}, \alpha_\mathrm{c})$ describes the corresponding structure, atom and component of the force respectively. $I \subset \mathbb{N}^3$ is the set of m_F selected force components that are used for the training while the index i in the first sum labels all m_U selected structures used for the energy training. Since there are $3N$ available forces per structure, using all of them would improve the accuracy of the forces at the expense of a degraded energy fit [16]. Different methods for the selection of the forces exist, the most popular being a random selection or a selection of the first m_F forces that yield the largest error. The balance of the error ratio between forces and energies can be tuned by changing the value of β or the ratio $\frac{m_U}{m_F}$. The effect of different values of β was studied in the work of Singraber and Morawietz [16].

2.4.2 Local Structural Descriptors

We have previously introduced the functions $\boldsymbol{G}_i : \mathbb{R}^{3N} \to \mathbb{R}^d$. There are several requirements we expect them to fulfill. First of all the $\boldsymbol{G}_i$ should have a form that is easily adjustable to different system sizes and satisfy the locality condition. Additionally, we want the approximated potential $\tilde{U}$ to be invariant under the symmetry operations discussed in section 2.3.1. Since we are thinking of the $\Phi_{[i]} \circ \boldsymbol{G}_i$ as local contributions it seems reasonable to also impose the symmetries on those terms. It can be incorporated by restricting the form of $\boldsymbol{G}_i$ adequately.

A simple solution to locality is to introduce a radial cutoff function $f_c(r)$ for which $f_c(r) = 0\ \forall r > r_c$, where r_c is the cutoff radius. Each contribution of atom $\boldsymbol{r}_j$ to $\boldsymbol{G}_i$ is then multiplied by $f_c(|\boldsymbol{r}_{ij}|)$, where $\boldsymbol{r}_{ij} = \boldsymbol{r}_i - \boldsymbol{r}_j$. We also require that $\boldsymbol{G}_i \in \mathcal{C}^1$ as we have mentioned before.

Translational invariance can be satisfied by choosing

$$\boldsymbol{G}_i(\boldsymbol{r}_1, \dots, \boldsymbol{r}_N) = \tilde{\boldsymbol{G}}_i(\boldsymbol{r}_{1i}, \dots, \boldsymbol{r}_{(i-1)i}, \boldsymbol{r}_{(i+1)i}, \dots, \boldsymbol{r}_{Ni}),$$

and we say that $\boldsymbol{G}_i$ is centered at atom i. In order to additionally enforce rotational symmetry one can further restrict the functions $\boldsymbol{G}_i$ to only be dependent on all possible inner products of the form $\boldsymbol{r}_{ij} \cdot \boldsymbol{r}_{kj}$. So a suitable function would look like

$$\boldsymbol{G}_i(\boldsymbol{r}_1, \dots, \boldsymbol{r}_N) = \overline{\boldsymbol{G}}_i(\boldsymbol{r}_{1i} \cdot \boldsymbol{r}_{1i}, \boldsymbol{r}_{1i} \cdot \boldsymbol{r}_{2i}, \dots, \boldsymbol{r}_{Ni} \cdot \boldsymbol{r}_{(N-1)i}, \boldsymbol{r}_{Ni} \cdot \boldsymbol{r}_{Ni}).$$

Permutation symmetry can be fulfilled by first requiring that for any $j, k \in \{1, \dots, N\}$, $j, k \neq i$ with $[j] = [k]$, $\boldsymbol{G}_i(\dots, \boldsymbol{r}_j, \dots, \boldsymbol{r}_k, \dots) = \boldsymbol{G}_i(\dots, \boldsymbol{r}_k, \dots, \boldsymbol{r}_j, \dots)$. For the case where $k = i$ it isn't as clear since imposing $\boldsymbol{G}_i(\dots, \boldsymbol{r}_i, \dots, \boldsymbol{r}_j, \dots) = \boldsymbol{G}_i(\dots, \boldsymbol{r}_j, \dots, \boldsymbol{r}_i, \dots)$ together with translational invariance for arbitrary $\boldsymbol{r}^{3N}$ would lead to a constant function[12]. To see what we need to impose instead, let us first enforce permutation symmetry on eq. (2.20), which leads to

$$\begin{aligned} &\boldsymbol{G}_i(\dots, \boldsymbol{r}_i, \dots, \boldsymbol{r}_j, \dots) + \boldsymbol{G}_j(\dots, \boldsymbol{r}_i, \dots, \boldsymbol{r}_j, \dots) \\ =\ &\boldsymbol{G}_i(\dots, \boldsymbol{r}_j, \dots, \boldsymbol{r}_i, \dots) + \boldsymbol{G}_j(\dots, \boldsymbol{r}_j, \dots, \boldsymbol{r}_i, \dots). \end{aligned}$$

Since this condition should hold in general and we have already ruled out one solution we are left with

$$\boldsymbol{G}_i(\dots, \boldsymbol{r}_i, \dots, \boldsymbol{r}_j, \dots) = \boldsymbol{G}_j(\dots, \boldsymbol{r}_j, \dots, \boldsymbol{r}_i, \dots).$$

This means that the functions $\boldsymbol{G}_i$ of atoms of the same element should only differ in their reference system.

Apart from the symmetry properties, the goal is to find a $\boldsymbol{G}_i$ that maps physically different local environments to distinctive points in $\mathbb{R}^d$. Depending on the number of atoms inside the cutoff sphere and the dimension d an injective map may not be possible. In that case, one can at least make sure that environments with important chemical differences are mapped to distinctive points. In ML this concept is known as feature extraction as part of the theory of dimensionality reduction.

Atom-Centered Symmetry Functions

In 2007 Behler and Parrinello [8] first introduced their set of suitable functions which they termed atom-centered symmetry functions (ACSFs). Actually, they proposed different forms that the components of $\boldsymbol{G}_i$ could have, the particular choice d and $\boldsymbol{G}_i$ then needs to be tailored to the system one is going to investigate. The first proposed symmetry function (SF)[13]

[12] Think of it as shifting the center of the SF to a different atom of the same element and enforcing the same value of $\boldsymbol{G}_i$ independent of the new environment.

[13] The wording is often sloppy whether SF refers to $\boldsymbol{G}_i$ or to one of its components

were [8, 54][14]

$$G_i^{\text{rad}}(T,\eta,R_s;\boldsymbol{r}^{3N}) = \sum_{\substack{j\neq i,\\ [j]=T}} e^{-\eta(r_{ij}-R_s)^2} f_{\text{c}}(r_{ij}) \tag{2.22}$$

$$\begin{aligned} G_i^{\text{ang.n.}}(T,T',\eta,\zeta,\lambda;\boldsymbol{r}^{3N}) = 2^{1-\zeta} \sum_{\substack{j,k\neq i,\ j<k\\ [j]=T,[k]=T'}} & (1+\lambda\cos(\theta_{ijk}))^{\zeta} \\ & \cdot e^{-\eta(r_{ij}^2+r_{ik}^2+r_{jk}^2)} f_{\text{c}}(r_{ij}) f_{\text{c}}(r_{ik}) f_{\text{c}}(r_{jk}). \end{aligned} \tag{2.23}$$

Here the parameters T and T' denote element types, $\eta,\zeta\in\mathbb{R}_{>0}$, $R_s\in\mathbb{R}_{\geq 0}$ and $\lambda\in\{-1,1\}$. The angle θ_{ijk} is measured between the atoms j and k as seen from i, so $\theta_{ijk}=\cos^{-1}\left(\frac{\boldsymbol{r}_{ji}\cdot\boldsymbol{r}_{ki}}{r_{ji}r_{ki}}\right)$. Equation (2.22) is called the radial SF and eq. (2.23) is the (narrow) angular SF. Furthermore different choices of cutoff functions are possible, typically one of the following is used

$$f_{\text{c}}(r) = \begin{cases} \frac{1}{2}\left[\cos\left(\frac{\pi r}{r_{\text{c}}}\right)+1\right], & \text{if } r\leq r_{\text{c}} \\ 0, & \text{else,} \end{cases} \tag{2.24}$$

$$f_{\text{c}}(r) = \begin{cases} \tanh^3\left(1-\frac{r}{r_{\text{c}}}\right), & \text{if } r\leq r_{\text{c}} \\ 0, & \text{else.} \end{cases} \tag{2.25}$$

Although defining the functions $\boldsymbol{G}_i$ with components of the form given in eqs. (2.22) and (2.23) have been very successful as feature maps of the local environment [5, 6, 54] the dimension d needs to be sufficiently high for an accurate description[15] And since functions like $\exp, \sin, \cos, \tanh$ are computationally expensive to evaluate, the evaluation of the SFs is often more demanding than the computation of small NNs. This problem has been addressed by Singraber [6] who minimized the number of calculations by making use of the similar form of many SF components. Also Bircher et al. [56] tackled this problem by proposing polynomial SFs with similar properties as an alternative since they are considerably cheaper to evaluate. Another problem is that the originally introduced SFs are specific to element combinations and since an accurate description of the environment needs sufficient information about all neighboring elements the number of needed SFs scales badly. As an example assume that one needs about a radial components in the SFs per element pair and b angular components per element triple. Then each $\boldsymbol{G}_i$ needs $d = aN_{\text{el}} + b\frac{1}{2}(N_{\text{el}}+1)N_{\text{el}}$ components. Obviously the dimension of the SFs is of the order $\mathcal{O}(N_{\text{el}}^2)$. This is why this method can get already computationally expensive when it is used for structures with more than ~ 4 different elements. Gastegger et al. [57] offered a solution to this problem by introducing the weighted atom-centered symmetry functions (wACSFs). As the name suggests they are based on Behler and Parrinello's original ACSFs but the weighted variants are not specific to the element pairs or triples but are combined in one function. Different element combinations are however weighted differently. Based on eqs. (2.22) and (2.23) the

[14] In many articles the sum is not taken over specific element types but over all atoms. However, the programs *RuNNer* and *n2p2*, which implement Behler and Parrinello's approach, make use of the element specific form.

[15] As an example a usable model for liquid water can be achieved with around $d = 27$ for hydrogen and $d = 30$ for oxygen [55].

components of the corresponding ACSFs are of the form [57]

$$W_i^{\mathrm{rad}}(\eta, R_s; \boldsymbol{r}^{3N}) = \sum_{j\neq i}^{N} g([j]) e^{-\eta(r_{ij}-R_s)^2} f_{\mathrm{c}}(r_{ij})$$

$$W_i^{\mathrm{ang.n.}}(\eta, R_s, \zeta, \lambda; \boldsymbol{r}^{3N}) = 2^{1-\zeta} \sum_{\substack{j,k\neq i,\\ j<k}}^{N} h([j],[k]) \left(1+\lambda\cos\left(\theta_{ijk}\right)\right)^{\zeta} \cdot e^{-\eta((r_{ij}-R_s)^2+(r_{ik}-R_s)^2+(r_{jk}-R_s)^2)} f_{\mathrm{c}}(r_{ij}) f_{\mathrm{c}}(r_{ik}) f_{\mathrm{c}}(r_{jk}),$$

where g and h are the mentioned weighting functions. The choices $g([i]) = Z_{[i]}$ and $h([i],[j]) = Z_{[j]}$ have been applied successfully. $Z_{[i]}$ is the atomic number of element $[i]$. For systems with 5 different elements Gastegger et al. showed that choosing $d = 35$ for wACSFs $\boldsymbol{W}_i$ lead to comparable accuracy as when setting $d = 220$ for the original ACSFs $\boldsymbol{G}_i$ while fixing the number of hidden layers [57].

Bispectrum of the Neighborhood Distribution

Although not used in this work a different approach than the ACSFs was developed by Bartók et al. [3] which could be utilized alternatively. In their work they started with the distribution

$$\rho_i(\boldsymbol{r}) = \delta(\boldsymbol{r}-\boldsymbol{r}_i) + \sum_{j\neq i} w_{[j]} \delta(\boldsymbol{r}-\boldsymbol{r}_j) f_{\mathrm{c}}(r_j),$$

which describes the local environment of atom i. The $w_{[j]}$ encode the element type. Next, this density is mapped from the local environment with cutoff r_{c} onto the 3-sphere S^3 and then this new density ρ_i' is expanded as a linear combination of hyperspherical harmonics $U^j_{m'm}$ [4]

$$\rho_i' = \sum_{j=0}^{\infty} \sum_{m,m'=-j}^{j} c^j_{m'm} U^j_{m'm}. \tag{2.26}$$

Obviously, this series must be truncated at some point and will affect the accuracy of the description of the environment. The expansion coefficients $c^j_{m'm}$ are then used to compute the bispectrum [4]

$$B_{j_1,j_2,j_3} = \sum_{m_1',m_1=-j_1}^{j_1} \sum_{m_2',m_2=-j_2}^{j_2} \sum_{m_3',m_3=-j_3}^{j_3} (c^{j_3}_{m_3'm_3})^* C^{j_3m_3}_{j_1m_1j_2m_2} C^{j_3m_3'}_{j_1m_1'j_2m_2'} c^{j_1}_{m_1'm_1} c^{j_2}_{m_2'm_2},$$

which is invariant under permutation and rotation. $C^{j_3m_3}_{j_1m_1j_2m_2}$ denote the Clebsch-Gordan coefficients. The bispectrum B_{j_1,j_2,j_3} or a subset of it is then used as the input for the NN.

2.4.3 Incorporate Electrostatic Interactions

Recently, the stages of development of NNPs have been classified into different generations [12, 14]. The particular form of the HDNNPs discussed in sections 2.4.1 and 2.4.2 correspond to the second generation (2G). Although this kind of NNP has been applied very successfully, it cannot model, by construction, any interaction between atoms that are further apart than

the diameter of the largest cutoff sphere. Specifically, it is impossible to explicitly model long-range interactions like Coulomb or dipolar forces beyond the cutoff sphere.

To overcome this limitation Artrith and Behler [58] introduced the third generation (3G) of NNPs, which introduce a second class of NNs that compute partial atomic charges as a function of the local environment of the respective atom. The charges can then be used to calculate the Coulomb potential energy with standard methods and the non-Coulombic part of the potential energy is then modelled as in eq. (2.20). But since the charge is independent of the configuration outside the cutoff sphere, charge transfer beyond this domain cannot be described. This can become a problem when modeling a region of a material close to an interface or if one considers long, polarized molecules [14]. Simply increasing the cutoff radius does not solve the issue in general. Although in practice all simulations are limited to a box of finite size, introducing a cutoff sphere that can describe charge transfer along the complete domain of the structure defeats the purpose of the cutoff method and would increase the computational effort immensely.

Recently a fourth generation (4G) of the NNPs by Behler and Parrinello have been introduced by Ko et al. [14] that is capable of describing charge transfer and long-range interactions. The approach is inspired by the charge equilibration neural network technique (CENT) [15] which implements a global charge equilibration scheme (Qeq) based on the concept of electronegativity. This is explained in more detail in section 3.2.

The charge equilibration distributes the atomic charges in a way that minimizes the potential energy of the system. For this an expression of the charge-dependent part of the energy is necessary. Including only the electrostatic interaction between the charges would not take effects inside the atom into account, which is why quantities such as the atom-specific electronegativity and the element-specific atomic hardness are introduced. Their particular physical meaning will be elaborated in section 3.2. The electronegativity χ_i of atom i is constructed to be environment dependent whereas the atomic hardness $J_{[i]}$ is fixed for each element, hence the subscript $[i]$. Both in the CENT and the 4G NNP, χ_i is computed by a NN $\Phi_{\chi,[i]}$ as a function of the local environment that is again described by a separate set of ACSFs, so

$$\chi_i = \Phi_{\chi,[i]} \circ \boldsymbol{G}_{\chi,i}.$$

The $\Phi_{\chi,[i]}$ are again the same for atoms of the same element. Subsequently, the electronegativities are used to obtain the atomic charges q_i with the help of the equilibration scheme discussed in section 3.2.

In order to compute the total potential energy of the structure a different approach is used compared to eq. (2.19). The potential is now divided into a long- and a short-ranged contribution

$$U = U^{\text{elec}} + \sum_{i=1}^{N} U_i^{\text{short}}. \tag{2.27}$$

The U_i^{short} represent local contributions of short-ranged interactions similar to the 2G potential with a minor difference. Namely, not only $\boldsymbol{G}_{U,i}$[16] is used as the input information but also the charge q_i of atom i

$$\tilde{U}_i^{\text{short}} = \Phi_{U,[i]}(\boldsymbol{G}_{U,i}, \tilde{q}_i).$$

A diagram of the complete architecture of the 4G HDNNP is shown in fig. 2.2.

[16] In principle, $\boldsymbol{G}_i$ used as argument for $\Phi_{[i]}$ and $\boldsymbol{G}'_i$ used for $\Psi_{[i]}$ do not need to be the same but in practice they often are.

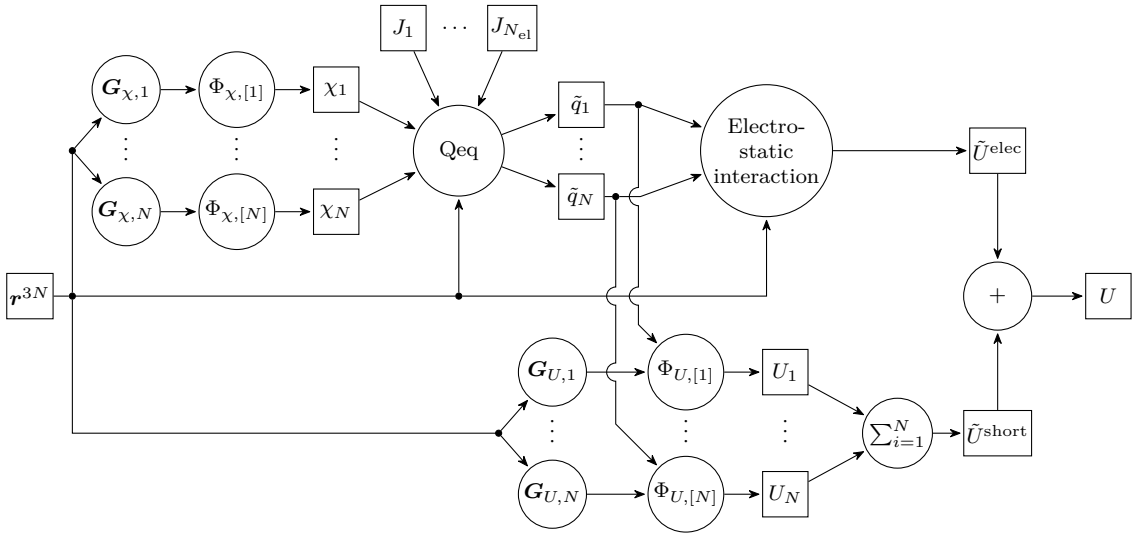


Figure 2.2: 4G HDNNP architecture. Physical quantities are drawn as rectangles, whereas maps are drawn as circles. Going from left to right, the architecture splits into an electrostatic part (upper half) and a short-ranged part (lower half). The individual contributions $\tilde{U}^{\mathrm{elec}}$ and $\tilde{U}^{\mathrm{short}}$ are then added together in the final step.

U^{elec} is roughly speaking the potential energy contributed by electrostatic interactions that occur over distances larger than the cutoff distance r_{c}. To be more precise, let u_{ij} be the pair potential between atoms i and j. Then we introduce a screening function f^{screen} and write [59]

$$U^{\mathrm{elec}} = \sum_{i<j} u_{ij}\ f_{\mathrm{screen}}(r_{ij}).$$

The screening function $f_{\mathrm{screen}} : \mathbb{R}_{\geq 0} \to [0,1]$ maps distances greater than the cutoff radius to one and distances smaller than an inner cutoff radius $r_{\mathrm{i,c}}$ to 0 but increases monotonically and smoothly from zero to one on the interval $[r_{\mathrm{i,c}}, r_{\mathrm{c}}]$. Thus $f_{\mathrm{screen}}(0) = 0$ which screens any self-interaction completely. Multiple definitions are possible but here we settled with

$$f_{\mathrm{screen}} = \begin{cases} 0 & \text{if } r < r_{\mathrm{i,c}} \\ \frac{1}{2}\left(1 - \cos\left(\pi \frac{r - r_{\mathrm{i,c}}}{r_{\mathrm{c}} - r_{\mathrm{i,c}}}\right)\right) & \text{if } r_{\mathrm{i,c}} < r < r_{\mathrm{c}} \\ 1 & \text{else.} \end{cases} \tag{2.28}$$

By dampening the electrostatic potential in U^{elec} between closely located atoms one basically changes the form of U_i^{short} that has to be approached by the second class of NNs. Apparently, this method improves the fit quality [59].

In contrast to 2G the training of the 4G NNP becomes a two-step procedure. Firstly, the NNs $\Phi_{\chi,[i]}$ computing the electronegativities need to be optimized. The training data consists of vectors of atomic charges $\boldsymbol{q}_i$ that are calculated from ab initio methods with an appropriate charge partition scheme like the Hirshfeld charges introduced in section 3.1.1. The vector contains all charges belonging to the same structure i. The charge vector $\tilde{\boldsymbol{q}}_i = \tilde{\boldsymbol{q}}_i(\boldsymbol{\chi}(\boldsymbol{\Theta}_\chi; \boldsymbol{r}_i^{3N}), \boldsymbol{J}, \boldsymbol{r}_i^{3N})$ computed by the NNs is differentiable with respect to $\boldsymbol{\Theta}_\chi$ and $\boldsymbol{J}$ and hence can be optimized with one of the common training algorithms. $\boldsymbol{\chi}$ and $\boldsymbol{J}$ are again vectors containing all χ_j and J_j belonging to the same structure, respectively and the label

in $\boldsymbol{\Theta}_\chi$ indicates that the weights and biases of all $\Phi_{\chi,[i]}$ are meant. It is noteworthy that Ko et al. [14] decided to optimize $\boldsymbol{J}$ also during training under the constraint $J_i \geq 0$[17] for all i.

The second step of the training consists of training the NNs $\Phi_{U,[i]}$. This is very similar to the training procedure of the 2G NNP with the only difference that for each training pattern $\tilde{\boldsymbol{q}}_i$ and $\tilde{U}^{\text{elec}}$ needs to be computed first by utilizing the optimized NNs $\Phi_{\chi,[i]}$. Afterwards the empirical error between $U(\boldsymbol{r}^{3N})$ and

$$\tilde{U} = \tilde{U}^{\text{elec}} + \sum_{i=1}^{N} \Phi_{U,[i]}(\boldsymbol{\theta}_{U,[i]}; \boldsymbol{r}^{3N}, \tilde{q}_i), \tag{2.29}$$

combined with the error between the computed and the reference forces is minimized with respect to $\boldsymbol{\Theta}_U$ during training.

There are several choices for the atomic charge partition scheme and also for the particular charge density of each atomic charge. In this work we adopted the choice of Ko et al. [14] and thus we have used the Hirshfeld method combined with a Gaussian charge distribution of atom i

$$\rho_i(\boldsymbol{r}) = \frac{q_i}{(2\pi\sigma_{[i]})^{\frac{3}{2}}} e^{-\frac{(\boldsymbol{r}-\boldsymbol{r}_i)^2}{2\sigma_{[i]}^2}}. \tag{2.30}$$

The width of the charge $\sigma_{[i]}$ is the covalent radius of the respective element.

[17] This ensures that the charge equilibration is well-defined

Chapter 3

Electrostatics

In the following, we assume a unit system where $4\pi\varepsilon_0 = 1$ such that the Coulomb potential between two charges q_1, q_2 takes the form

$$U = \frac{q_1 q_2}{\|\boldsymbol{r}_1 - \boldsymbol{r}_2\|}.$$

3.1 Charge Partition

In order to include electrostatic interactions in atomistic models a simple method is to assign a charge to each atom. In context of DFT computations, a natural approach is to partition the total electron density $\rho^{\mathrm{e}}_{\mathrm{tot}}$ into individual electron densities ρ^{e}_i that are contributed by the atoms. Although the total electron density is well-defined, the atomic contributions are not in the sense that a unique partition scheme does not exist as can easily be seen by comparing different methods that all produce the same total electron density [60–62]. Here we will introduce the Hirshfeld charges [61] that are mainly used in this work and its successor the Hirshfeld-I method also called iterative Hirshfeld method. The latter eliminates the need of an arbitrary choice in the original method and is overall in better agreement with other atomic charge models.

3.1.1 Hirshfeld Method

An often-used charge partition scheme is the one introduced by Hirshfeld [61]. In contrast to other methods like the Bader charges [63, 64], the electron density at each point is written as a sum of multiple atomic contributions $\rho^{\mathrm{e}}_{\mathrm{tot}}(\boldsymbol{r}) = \sum_i \hat{\rho}^{\mathrm{e}}_i(\boldsymbol{r})$. At each point $\boldsymbol{r}$ the atomic densities are written as a fraction of the total density

$$\hat{\rho}^{\mathrm{e}}_i(\boldsymbol{r}) = w_i(\boldsymbol{r})\rho^{\mathrm{e}}_{\mathrm{tot}}(\boldsymbol{r}),$$

with $\sum_i w_i(\boldsymbol{r}) = 1,\ w_i(\boldsymbol{r}) \in [0,1]$ everywhere inside the structure. Hirshfeld chose to relate the weights w_i to the electron densities $\rho^{\mathrm{e,n}}_i$ of isolated neutrally charged atoms. More precisely the spherically averaged electron densities

$$\bar{\rho}^{\mathrm{e,n}}_i(\boldsymbol{r}) = \begin{cases} \frac{1}{\Omega(\|\boldsymbol{r}-\boldsymbol{r}_i\|)} \int_{\partial B_{\|\boldsymbol{r}-\boldsymbol{r}_i\|}(\boldsymbol{r}_i)} \rho^{\mathrm{e,n}}_i \,\mathrm{d}S & \|\boldsymbol{r}-\boldsymbol{r}_i\| > 0 \\ \rho^{\mathrm{e,n}}_i(\boldsymbol{r}_i) & \text{else} \end{cases}$$

of atoms located at positions $\boldsymbol{r}_i$ are used. Here, $\Omega(r)$ is the surface area of a ball with radius r and $\partial B_r(\boldsymbol{x})$ is the boundary of a ball with radius r centered at $\boldsymbol{x}$. The weights are then defined as

$$w_i(\boldsymbol{r}) = \frac{\bar{\rho}_i^{\mathrm{e,n}}}{\sum_j \bar{\rho}_j^{\mathrm{e,n}}}(\boldsymbol{r}).$$

The atomic charge q_i can then be written as

$$q_i = \left(Z_i - \int_{\mathbb{R}^3} \hat{\rho}_i^{\mathrm{e}}(\boldsymbol{r}) \, \mathrm{d}^3\boldsymbol{r} \right) e,$$

where e is the elementary charge and Z_i the atomic number of atom i.

The Hirshfeld charges are appealing because of their simplicity and can be calculated efficiently compared to other charge models like the Bader charges. In contrast to Mulliken charges [60], the model is not explicitly dependent on the particular basis set that is used to compute the wave function of the system [65]. Furthermore when describing properties of a molecule in terms of properties of the corresponding atoms, the Hirshfeld charges yield a "transferable" model whereas e.g. Bader charges lack this feature [66].

However, since every atomic charge model is a rough approximation, it comes with its downsides. One of them is that the charges tend to be much smaller in magnitude compared to other atomic charge models [62, 67]. Consequently, dipole moments of molecules computed from the constituent atomic charges tend to be underestimated, e.g. in the case of water [68]. Another point that is often criticized is the arbitrary choice of relating the weighting functions w_i to the electron densities of the isolated neutral atoms. It was shown that when using different reference electron densities the computed charges change notably and can even lead to polarized symmetric molecules like N_2 when using electron densities of the reference system N^+N^- [67].

3.1.2 Hirshfeld-I Method

Bultinck et al. [62] extended the Hirshfeld method to an iterative scheme which they called *Hirshfeld-I*. In the following, we will drop the labels "e" and "n" from $\hat{\rho}_i^{\mathrm{e}}$ and $\bar{\rho}_i^{\mathrm{e,n}}$. Let $\bar{\rho}_i^j$ be the spherically averaged density of atom i at iteration j. The weighting function at iteration j is then defined as

$$w_i^j(\boldsymbol{r}) = \frac{\bar{\rho}_i^{j-1}}{\sum_k \bar{\rho}_k^{j-1}}(\boldsymbol{r}),$$

and the atomic electron density at iteration j is then

$$\hat{\rho}_i^j(\boldsymbol{r}) = w_i^j(\boldsymbol{r}) \rho_{\mathrm{tot}}(\boldsymbol{r}),$$

with the corresponding charge

$$q_i^j = \left(Z_i - \int_{\mathbb{R}^3} \hat{\rho}_i^j(\boldsymbol{r}) \, \mathrm{d}^3\boldsymbol{r} \right) e.$$

The important step is now that loosely speaking ρ_i^j (the reference density before averaging) is related to the electron density of the isolated atom i with charge q_i^j, so we write $\rho_i^j(q_i^j; \boldsymbol{r})$. In general, q_i^j is not an integer multiple of the elementary charge e which is why the following solution was suggested by Bultinck et al.

$$\rho_i^j(q_i^j; \boldsymbol{r}) = \rho_i^j(\lfloor q_i^j \rfloor; \boldsymbol{r})(\lceil q_i^j \rceil - q_i^j) + \rho_i^j(\lceil q_i^j \rceil; \boldsymbol{r})(q_i^j - \lfloor q_i^j \rfloor),$$

where $\lfloor . \rfloor$ and $\lceil . \rceil$ are the floor and ceiling function, respectively.

It was numerically shown that for a wide range of different initial reference densities the algorithm converged to the same atomic charges indicating that this method is not dependent on the initial reference densities anymore. Additionally the obtained charges were in general larger in magnitude compared to the original method [62]. An extension to periodic materials has been treated in [69]. A more efficient algorithm for computing the Hirshfeld-I charges is explained in [70]. A case study of the molecular dipole moments in liquid water have shown that the Hirshfeld-I method delivered better results concerning the experimental dipole moment compared to other commonly used atomic charge models [68].

3.2 Charge Equilibration

The first step is to express the total potential energy U as a function of the atomic charges q_i. Rappe and Goddard [71] provide a simple approximation. The total potential energy is written as a sum of atomic contributions that describe the inner-atomic interactions plus a second sum that incorporates the inter-atomic electrostatic interactions

$$U^{\mathrm{Qeq}} = \sum_{i=1}^{N} \left(U^0_{[i]} + \left.\frac{\partial U_i}{\partial q_i}\right|_{q_i=0} q_i + \frac{1}{2} \left.\frac{\partial^2 U_i}{\partial q_i^2}\right|_{q_i=0} q_i^2 \right) + U^{\mathrm{elec}}.$$

Here "Qeq" stands for charge equilibration since this particular form of the total potential energy is only used for the charge equilibration scheme. The terms in the first sum are approximations of the atomic energy by making a Taylor expansion up to second order of the energy of an isolated atom i with respect to its charge around the neutral state $q_i = 0$. $U^0_{[i]}$ is the reference energy of an isolated atom of element $[i]$. U_{elec} is the potential caused by the electrostatic interaction of the charge distribution. The first derivative $\chi_i = \left.\frac{\partial U_i}{\partial q_i}\right|_{q_i=0}$ is the *electronegativity* and $J_{[i]} = \left.\frac{\partial^2 U_i}{\partial q_i^2}\right|_{q_i=0}$ is called the *atomic hardness* [15] or sometimes the *idempotential* [71] of atom i.

The particular form of U^{elec} depends on the charge distribution of each atom. If the atomic charges are modeled as point charges the potential is well-known from any introductory course on electrostatics

$$U^{\mathrm{elec}} = \sum_{i<j} \frac{q_i q_j}{r_{ij}},$$

where $r_{ij} = \|\boldsymbol{r}_i - \boldsymbol{r}_j\|$ is the distance between atom i and j.

Another popular choice are smeared charges like the Gaussian distributed charges, see eq. (2.30). The charge distribution of the system is written as

$$\rho = \sum_{i=1}^{N} \rho_i,$$

where ρ_i are Gaussian charge distributions located at $\boldsymbol{r}_i$. Since we are not dealing with point

charges anymore we can compute the resulting potential via[1]

$$\begin{aligned} U^{\text{elec}} &= \frac{1}{2}\int_{\mathbb{R}3} \mathrm{d}^3\boldsymbol{x} \int_{\mathbb{R}^3} \mathrm{d}^3\boldsymbol{y}\, \frac{\rho(\boldsymbol{x})\rho(\boldsymbol{y})}{\|\boldsymbol{x}-\boldsymbol{y}\|} \\ &= \frac{1}{2}\int_{\mathbb{R}3} \mathrm{d}^3\boldsymbol{x} \int_{\mathbb{R}^3} \mathrm{d}^3\boldsymbol{y}\, \sum_{i,j} \frac{\rho_i(\boldsymbol{x})\rho_j(\boldsymbol{y})}{\|\boldsymbol{x}-\boldsymbol{y}\|} \\ &= \frac{1}{2}\sum_{i,j}\int_{\mathbb{R}3} \mathrm{d}^3\boldsymbol{x} \int_{\mathbb{R}^3} \mathrm{d}^3\boldsymbol{y}\, \frac{\rho_i(\boldsymbol{x})\rho_j(\boldsymbol{y})}{\|\boldsymbol{x}-\boldsymbol{y}\|}. \end{aligned}$$

The integral inside the sum can be expressed in terms of the *error function* $\operatorname{erf}(x)$. We compute the pair potential as [72]

$$u_{ij}(r_{ij}) = \int_{\mathbb{R}3} \mathrm{d}^3\boldsymbol{x} \int_{\mathbb{R}^3} \mathrm{d}^3\boldsymbol{y}\, \frac{\rho_i(\boldsymbol{x})\rho_j(\boldsymbol{y})}{\|\boldsymbol{x}-\boldsymbol{y}\|} = \frac{q_i q_j}{r_{ij}} \operatorname{erf}\left(\frac{r_{ij}}{\sqrt{2(\sigma_i^2+\sigma_j^2)}}\right). \tag{3.1}$$

A derivation of the solution can be found in appendix A.1. The case $r_{ij} = 0$ can be computed by taking the limit $r_{ij} \to 0$ and applying the rule of L'Hôpital

$$u_{ij}(0) = \lim_{r_{ij}\to 0} \frac{q_i q_j}{r_{ij}} \operatorname{erf}\left(\frac{r_{ij}}{\sqrt{2(\sigma_i^2+\sigma_j^2)}}\right) = \frac{2q_i q_j}{\sqrt{2\pi(\sigma_i^2+\sigma_j^2)}}.$$

The case $i = j$ represents the self-interaction but $u_{ii}(0)$ counts the interaction energy twice. That is why we distinguish the actual self-energy by

$$u_{\text{self},i} = u_{ii}(0) = \frac{q_i^2}{2\sqrt{\pi}\sigma_i}. \tag{3.2}$$

Under the condition $\boldsymbol{r}_i \neq \boldsymbol{r}_j$ if $i \neq j$ for all $i, j \in \{1, \ldots, N\}$ we can write

$$U^{\text{elec}} = \sum_{i<j} \frac{q_i q_j}{r_{ij}} \operatorname{erf}\left(\frac{r_{ij}}{\sqrt{2}\gamma_{ij}}\right) + \sum_{i=1}^{N} \frac{q_i^2}{2\sqrt{\pi}\sigma_i},$$

where we have introduced the constants

$$\gamma_{ij} = \sqrt{\sigma_i^2 + \sigma_j^2}. \tag{3.3}$$

The second sum represents all self-interactions of the individual Gaussian charge distributions whereas the first sum describes the interactions between distinct charge distributions.

Putting everything together, the potential used for the charge equilibration in the case of Gaussian distributed charges is

$$\begin{aligned} U^{\text{Qeq}} &= \sum_{i=1}^{N} \left[U_{[i]}^0 + \chi_i q_i + \frac{1}{2} J_i q_i^2 + \frac{q_i^2}{2\sqrt{\pi}\sigma_i} + \frac{1}{2} \sum_{\substack{j \\ j\neq i}} \frac{q_i q_j}{r_{ij}} \operatorname{erf}\left(\frac{r_{ij}}{\sqrt{2}\gamma_{ij}}\right) \right] \\ &= \sum_{i=1}^{N} U_{[i]}^0 + \boldsymbol{\chi}^\intercal \boldsymbol{q} + \frac{1}{2}\boldsymbol{q}^\intercal \boldsymbol{A} \boldsymbol{q}. \end{aligned}$$

[1] For point charges described by delta distributions the integral diverges. This artefact is known as infinite "self-energy" in the theory of classical electrodynamics.

Here we have introduced the symmetric matrix [14, 15]

$$(\boldsymbol{A})_{ij} = \begin{cases} J_i + \frac{1}{\sqrt{\pi}\sigma_i} & \text{if } i = j \\ \frac{1}{r_{ij}} \operatorname{erf}\left(\frac{r_{ij}}{\sqrt{2}\gamma_{ij}}\right) & \text{else.} \end{cases} \tag{3.4}$$

The column vectors $\boldsymbol{q}$ and $\boldsymbol{\chi}$ contain all q_i and χ_i, respectively, as seen in section 2.4.3.

The charge equilibration is now conducted by minimizing U^{Qeq} with respect to all charges q_i. There is however the constraint that the total charge of the system $q^{\mathrm{tot}} = \sum_i q_i$ is fixed. Differently formulated, we want to minimize U^{Qeq} on the hypersurface defined by $g(\boldsymbol{q}) = \sum_i q_i - q^{\mathrm{tot}} = 0$. This problem can be solved by making use of the method of Lagrange multipliers. Instead of minimizing U^{Qeq} we introduce the Lagrangian function

$$\mathcal{L}(\boldsymbol{q}, \lambda) = \boldsymbol{\chi}\boldsymbol{q} + \frac{1}{2}\boldsymbol{q}^{\intercal}\boldsymbol{A}\boldsymbol{q} + \lambda(q_1 + \dots q_N - q^{\mathrm{tot}})$$

By using the notation

$$\bar{\boldsymbol{q}} = \begin{pmatrix} \boldsymbol{q} \\ \lambda \end{pmatrix}$$

we can establish the necessary minimization condition under constraint as [15]

$$\boldsymbol{\nabla}_{\bar{q}}\mathcal{L} = \boldsymbol{0} \Leftrightarrow \begin{cases} \sum_j A_{ij}q_j + \lambda + \chi_i = 0 \quad \forall i \in \{1, \dots, N\} \\ q^{\mathrm{tot}} = \sum_i q_i. \end{cases} \tag{3.5}$$

By introducing

$$\bar{\boldsymbol{A}} = \begin{pmatrix} & & & 1 \\ & \boldsymbol{A} & & \vdots \\ & & & 1 \\ 1 & \dots & 1 & 0 \end{pmatrix} \quad \text{and} \quad \bar{\boldsymbol{\chi}} = \begin{pmatrix} \boldsymbol{\chi} \\ -q_{\mathrm{tot}} \end{pmatrix}$$

we can rewrite eq. (3.5) as a linear system of equations

$$\bar{\boldsymbol{A}}\bar{\boldsymbol{q}} = -\bar{\boldsymbol{\chi}}. \tag{3.6}$$

This system of equations has a unique solution because of the following.

Proposition 3.1. *If* $J_i \geq 0$ *for all* $i \in \{1, \dots, N\}$, $(\sigma_i, \boldsymbol{r}_i) \neq (\sigma_j, \boldsymbol{r}_j)$ *if* $i \neq j$, *and* $\boldsymbol{q} \neq \boldsymbol{0}$ *then* $\det(\bar{\boldsymbol{A}}) \neq 0$

Proof. First, we will show that the matrix $\boldsymbol{A}$ is positive definite if $J_i \geq 0\ \forall i$ and $\boldsymbol{q} \neq 0$. The matrix can be written as the sum $\boldsymbol{A} = \boldsymbol{B} + \mathrm{diag}(\boldsymbol{J})$, where $\mathrm{diag}(\boldsymbol{J})$ is a diagonal matrix with $\boldsymbol{J}$ as its entries. $\boldsymbol{B}$ describes the complete electrostatic interaction caused by the Gaussian charge distributions. For non-vanishing charge distributions ρ with non-diverging integral

$$U^{\mathrm{elec}} = \frac{1}{2}\int_{\mathbb{R}^3} \mathrm{d}^3\boldsymbol{x} \int_{\mathbb{R}^3} \mathrm{d}^3\boldsymbol{y}\, \frac{\rho(\boldsymbol{x})\rho(\boldsymbol{y})}{\|\boldsymbol{x} - \boldsymbol{y}\|},$$

and strong enough decay when $\|\boldsymbol{x}\| \to \infty$, the potential is strictly positive. For more details see appendix A.2 where this is proven for a charge density consisting of Gaussian charges.

Hence, $U^{\text{elec}} = \boldsymbol{q}^\intercal \boldsymbol{B} \boldsymbol{q} \geq 0$ for any $\boldsymbol{q}$. If additionally $i \neq j$ implies $(\sigma_i, \boldsymbol{r}_i) \neq (\sigma_j, \boldsymbol{r}_j)$[2], then $\boldsymbol{q}^\intercal \boldsymbol{B} \boldsymbol{q} > 0$ for any $\boldsymbol{q} \neq \boldsymbol{0}$, which means $\boldsymbol{B}$ is positive definite and thus $\boldsymbol{A}$ is too. Since any positive definite matrix is invertible with positive definite inverse it follows that $\bar{\boldsymbol{A}}$ is invertible. This can be seen by using

$$\det \begin{pmatrix} \boldsymbol{C} & \boldsymbol{d} \\ \boldsymbol{d}^\intercal & 0 \end{pmatrix} = -\det(\boldsymbol{C}) \boldsymbol{d}^\intercal \boldsymbol{C}^{-1} \boldsymbol{d},$$

where $\boldsymbol{C}$ is a $(n-1) \times (n-1)$ matrix and $\boldsymbol{d}$ is a $(n-1)$-dimensional vector. Applied on $\bar{\boldsymbol{A}}$ we obtain

$$\det(\bar{\boldsymbol{A}}) = -\det(\boldsymbol{A}) (1, \ldots, 1) \boldsymbol{A}^{-1} (1, \ldots, 1)^\intercal \neq 0,$$

since $\boldsymbol{A}$ and $\boldsymbol{A}^{-1}$ are positive definite. □

What is left to show is that $U^{\text{Qeq}}(\boldsymbol{q}^*)$ is a local minimum along the constraint of charge conservation. Here, $\boldsymbol{q}^*$ is the charge vector satisfying eq. (3.5). For this, we expand it up to second order at the solution $\boldsymbol{q}^*$

$$U^{\text{Qeq}}(\boldsymbol{q}^* + \boldsymbol{h}) \approx U^{\text{Qeq}}(\boldsymbol{q}^*) + \boldsymbol{\nabla} U^{\text{Qeq}}\big|_{\boldsymbol{q}=\boldsymbol{q}^*} \cdot \boldsymbol{h} + \frac{1}{2} \boldsymbol{h}^\intercal \, \boldsymbol{H}_{U^{\text{Qeq}}}\big|_{\boldsymbol{q}=\boldsymbol{q}^*} \boldsymbol{h}.$$

The matrix $\boldsymbol{H}_{U^{\text{Qeq}}} = \boldsymbol{A}$ is the Hessian matrix of U^{Qeq}. Equation (3.5) implies that $\boldsymbol{\nabla} U^{\text{Qeq}}\big|_{\boldsymbol{q}=\boldsymbol{q}^*} = -\lambda \, \boldsymbol{\nabla} g\big|_{\boldsymbol{q}=\boldsymbol{q}^*}$. This means that any tangent vector $\boldsymbol{h}$ of the hypersurface $g(\boldsymbol{q}) = 0$ at point $\boldsymbol{q}^*$ is orthogonal to $\boldsymbol{\nabla} U^{\text{Qeq}}\big|_{\boldsymbol{q}=\boldsymbol{q}^*}$. Hence,

$$U^{\text{Qeq}}(\boldsymbol{q}^* + \boldsymbol{h}) \approx U^{\text{Qeq}}(\boldsymbol{q}^*) + \frac{1}{2} \boldsymbol{h}^\intercal \boldsymbol{A} \boldsymbol{h} > U^{\text{Qeq}}(\boldsymbol{q}*)$$

for any infinitesimal, non-vanishing $\boldsymbol{h}$ of the tangent space of $g(\boldsymbol{q}) = 0$ at $\boldsymbol{q}^*$.

3.3 Long-Range Interactions

In MD electrostatics is considered to be a *long-range* interaction. This term is typically used to classify interactions which potential energy contribution U can be written as a sum of pair potentials $U = \frac{1}{2} \sum_{i,j} u_{ij}(r_{ij})$, where $i, j \in \{1, \ldots, N\}$. The definition of long-range and *short-range* interaction often varies depending on the field of study and may not always be formulated precisely. In this work we use the following definition.

Definition 3.1. An interaction described by a pair-potential u_{ij} is called *short-ranged* if for any $i, j \in \{1, \ldots, N\}$ there exists an $R \geq 0$ such that the integral

$$\int_R^\infty \mathrm{d}r \; |u_{ij}(r)| r^{d-1}$$

is convergent. Here, d is the spatial dimension of the system. Conversely, if the interaction is not short-ranged, it is called *long-ranged*.

[2] Otherwise Gaussian charges which only differ in their sign could cancel out

This definition can be motivated by the fact that in MD interactions are often only included up to a certain cutoff r_c. Consider an infinitely large system where we want to compute the potential energy associated with a subdomain consisting of N atoms[3]. The cutoff introduces an error

$$\Delta U = \frac{1}{2}\sum_{i=1}^{N}\sum_{\substack{j,\\ r_{ij}>r_c}} u_{ij}(r_{ij}).$$

For simplicity assume $u_{ij}(r) = a_{ij}f(r) \leq af(r)$ where the a_{ij} are chosen from a finite set[4] such that we can define $a = \max_{ij}|a_{ij}|$. Further, assume a uniform neighbor distribution $n(\boldsymbol{x}) = n_0$ for $\|\boldsymbol{x}\| \geq r_c$. Then a rough estimate for the upper limit of ΔU can be written as [73]

$$\Delta U \lessapprox \frac{N}{2} 4\pi n_0 \int_{r_c}^{\infty} \mathrm{d}r\; a|f(r)|r^2.$$

By definition there exists a pair (i,j) with $|u_{ij}(r)| = a|f(r)|$. If the pair potential is short-ranged, the integral converges whereas if it is long-ranged the cutoff error could diverge under the here introduced assumptions. Thus, in many applications a more careful treatment of this kind of interactions needs to be done since the cutoff method can introduce major systematic errors.

An example of a short-range interaction is the Lennard-Jones potential whereas the electrostatic potential is long-ranged. Note that in the literature it is sometimes found that any pair potential decaying faster than $\frac{1}{r^d}$ is considered short-ranged. However, this is not equivalent to our definition, as can be seen from the example $u_{ij}(r) = \frac{1}{r^d \ln r}$

3.4 Electrostatics Under Periodic Boundary Conditions

3.4.1 Periodic Boundary Conditions

When physical simulations are employed to compute the properties of a material it is most often done in a cell spanned by the linearly independent vectors $\boldsymbol{a}_1$, $\boldsymbol{a}_2$ and $\boldsymbol{a}_3$ (assuming a three-dimensional structure). Particularly, if one is interested in the properties of the bulk, then PBCs are enforced. The particles in the simulation cell are treated as if they are under the influence of infinitely many identical images of the cell stacked up along the dimensions the PBC is applied to. In the case of PBCs along each dimension, this leads to a lattice with the lattice vectors $\boldsymbol{a}_i$. In the following, we assume a three-dimensional cell ($d = 3$) with PBCs along each dimension. We are now going to make this more precise. Let Ω be the simulation cell. It is the set

$$\Omega = \left\{\boldsymbol{x} \in \mathbb{R}^3 | \boldsymbol{x} = c_1\boldsymbol{a}_1 + c_2\boldsymbol{a}_2 + c_3\boldsymbol{a}_3;\; c_1, c_2, c_3 \in [0,1]\right\}$$

Using the $\boldsymbol{a}_i$ as basis vectors we can describe each point $\boldsymbol{x}$ by the vector $\boldsymbol{c} = (c_1, c_2, c_3)^\intercal$, known as the *fractional* coordinates. PBCs along the dimension spanned by $\boldsymbol{a}_1$ means that we identify the following points

$$(0, c_2, c_3) \sim (1, c_2, c_3) \quad \forall c_2, c_3 \in [0,1].$$

[3] E.g. the energy of a unit cell under periodic boundary conditions.

[4] The a_{ij} could be element specific

The definition is analogous for $\boldsymbol{a}_2$ and $\boldsymbol{a}_3$. This is in fact the definition of the three-dimensional torus $\mathbb{T}^3 = {}^{\mathbb{R}^3}\!/_{\mathbb{Z}^3}$. In Riemannian geometry it is well-known that this torus is flat when equipped with the unique Riemannian metric such that the covering map

$$\begin{aligned} p : \mathbb{R}^3 &\to \mathbb{T}^3 \\ \boldsymbol{x} &\mapsto [\boldsymbol{x}], \end{aligned} \tag{3.7}$$

becomes a Riemannian covering[5]. For more details, see appendix B.2.1. Here $[\boldsymbol{x}]$ are the equivalence classes induced by $\mathbb{Z}^3$. For a reference of Riemannian covering maps see [74].

To simplify the notation we will now restrict the discussion to the special case $\boldsymbol{a}_i = \boldsymbol{e}_i$, where the latter are the basis vector of the Cartesian coordinate system. A generalization is straightforward. The canonical way to project points from $\mathbb{R}^3$ to $\mathbb{T}^3$ is via the map p. For any periodic function $g : \mathbb{R}^3 \to \mathbb{C}$, $g(\boldsymbol{x} + \boldsymbol{n}) = g(\boldsymbol{x})$, where $\boldsymbol{n} \in \mathbb{Z}^3$, we can define a corresponding function $f(\boldsymbol{x}) : \mathbb{T} \to \mathbb{C}$ by setting [75]

$$g(\boldsymbol{x}) = f([\boldsymbol{x}]).$$

Since the identification between g and f is so straightforward, they are often denoted as the same function.

In order to include electrostatic interactions under PBCs we have to find the electrostatic potential ϕ as a solution of the Laplace equation

$$\Delta\phi(\boldsymbol{x}) = -4\pi\bar{\rho}(\boldsymbol{x}),$$

where $\bar{\rho}(\boldsymbol{x})$ is the charge distribution on the torus. Here, Δ is the Laplace-Beltrami operator on the torus. However, when using the natural choice of local coordinates, it becomes the Laplace operator in $\mathbb{R}^3$. For more details, see appendix B.2.2. The charge distribution on $\mathbb{T}^3$ can be constructed via the periodization [76]

$$\bar{\rho}(\boldsymbol{x}) = \sum_{\boldsymbol{n}\in\mathbb{Z}^3} \rho(\boldsymbol{x} + \boldsymbol{n})$$

Whether $\bar{\rho}$ diverges or not depends on the properties of ρ. If $\rho \in L^1(\mathbb{R}^3)$ then $\bar{\rho}$ is finite a.e. on $\mathbb{T}^3$

$$\begin{aligned} \int_{\mathbb{T}^3} \mathrm{d}\boldsymbol{x} \sum_{\boldsymbol{n}\in\mathbb{Z}^3} |\rho(\boldsymbol{x} + \boldsymbol{n})| &= \sum_{\boldsymbol{n}} \int_{\mathbb{T}^3} \mathrm{d}\boldsymbol{x}\, |\rho(\boldsymbol{x} + \boldsymbol{n})| \\ &= \sum_{\boldsymbol{n}} \int_{\boldsymbol{n}+\mathbb{T}^3} \mathrm{d}\boldsymbol{x}\, |\rho(\boldsymbol{x})| \\ &= \int_{\mathbb{R}^3} \mathrm{d}\boldsymbol{x}\, |\rho(\boldsymbol{x})| < \infty. \end{aligned}$$

There has been some abuse of notation since $\mathbb{T}^3$ is not a subset of $\mathbb{R}^3$. We do not make this rigorous here since this kind of notation also appears in mathematical literature like [76]. For now one can interpret $\int_{\mathbb{T}^3} \sim \int_{(0,1)^3}$ and argue that the set of points on $\mathbb{T}^3$ that cannot be identified with a subset of $(0,1)^3$ has measure zero. The interchange of the sum with the integral is allowed by the monotone convergence theorem for the Lebesgue integral.

[5] Assuming $\mathbb{R}^3$ is equipped with the Euclidean metric

3.4.2 Conditional Convergence of the Electrostatic Energy

When conducting simulations of particles that interact electrostatically it is often necessary to compute the electrostatic energy of the simulation box. If PBCs are applied this energy is often written in the following form

$$U = \frac{1}{2} \sum_{\boldsymbol{n} \in \mathbb{L}}{}' \sum_{i,j=1}^{N} u_{ij}(\|\boldsymbol{r_i} - \boldsymbol{r_j} + \boldsymbol{n}\|). \tag{3.8}$$

Here, u_{ij} is the pair potential between two charge distributions centered at $\boldsymbol{r}_i$, $\boldsymbol{r}_j$, e.g. for two point charges q_i, q_j it becomes $u_{ij}(r_{ij}) = \frac{q_i q_j}{r_{ij}}$, where r_{ij} is the distance between charge i and j. We have also generalized the unit cell to an arbitrary triclinic cell, thus the lattice points are not elements of $\mathbb{Z}^3$ anymore but are elements of

$$\mathbb{L} = \left\{ \boldsymbol{x} \in \mathbb{R}^3 | \boldsymbol{x} = n_1 \boldsymbol{a}_1 + n_2 \boldsymbol{a}_2 + n_3 \boldsymbol{a}_3,\ n_i \in \mathbb{Z} \right\}.$$

The prime at the sum indicates that if $\boldsymbol{n} = \boldsymbol{0}$ then the term $i = j$ is not included. This is typically done when u_{ij} is the potential between two point charges since the self-interaction diverges. If one considers a u_{ij} for which $u_{ij}(0) < \infty$ then one does not need to exclude the term, e.g. Gaussian distributed charges. Whether the series in eq. (3.8) converges or not depends on u_{ij}. In the following, we will only discuss the convergence properties of the expression in eq. (3.8) for point charges. However, up to technical details, the discussion for Gaussian charges can be conducted analogously.

For point charges, the electrostatic energy is often written as

$$U = \frac{1}{2} \sum_{\boldsymbol{n} \in \mathbb{L}}{}' \sum_{i,j=1}^{N} \frac{q_i q_j}{\|\boldsymbol{r}_i - \boldsymbol{r}_j + \boldsymbol{n}\|}.$$

Since $\mathbb{L} \sim \mathbb{Z}^3$ is countable, there exists a bijection $f : \mathbb{L} \to \mathbb{N}$ which defines an order of summation $f(\boldsymbol{n}) = k$. Thus we can label the points in $\mathbb{L}$ by k and write $\boldsymbol{n}_k$ to specify an order. Then we can use this notation to express the electrostatic energy under this order of summation as a series

$$U = \frac{1}{2} \sum_{k=1}^{\infty}{}' \sum_{i,j=1}^{N} \frac{q_i q_j}{\|\boldsymbol{r}_i - \boldsymbol{r}_j + \boldsymbol{n}_k\|}. \tag{3.9}$$

Physically there is no reason for choosing a specific order so we would expect that this series is independent under a permutation of the labels k. One could go even further and specify an order over all charges individually and not only over the lattice points $\boldsymbol{n}$

$$U = \frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{\infty}{}' \frac{q_i q'_k}{\|\boldsymbol{r}_i - \boldsymbol{r}'_k\|}.$$

Here q'_k denotes the charge at an arbitrary position $\boldsymbol{r}'_k$ in one of the periodic copies. However, this series is definitely not invariant under permutation of the indices k since it is not absolutely convergent as can be seen as from this simple estimate

$$\frac{1}{2} \sum_{i=1}^{N} \sum_{k=1}^{\infty}{}' \frac{|q_i q'_k|}{\|\boldsymbol{r}_i - \boldsymbol{r}'_k\|} \geq \frac{1}{2\|\boldsymbol{a}_1\|} \sum_{n=1}^{\infty} \frac{q_i^2}{n} = \infty \quad \forall q_i \neq 0,$$

where we have used that the set $\{n\boldsymbol{a}_1\}_{n\in\mathbb{N}}$ is part of $\mathbb{L}$ and for each $\boldsymbol{n} \neq \boldsymbol{0}$ there exists a term for which $q_i = q'_k$. This problem is also known when defining Madelung's constant for crystals [77]. Since the series is not absolutely convergent it follows from the Riemann rearrangement theorem that even if the series is still conditionally convergent, there exists a permutation of k for each real number such that the series has this number as its limit.

Indeed, the series in eq. (3.9) can be conditionally convergent if charge neutrality $\sum_i q_i = 0$ is satisfied [78–80]. This is shown by rewriting the summands in eq. (3.9) as a product of the charge and the potential $\phi_{\boldsymbol{n}_k}$ of each periodic image it is interacting with

$$U = \frac{1}{2}{\sum_{k=1}^{\infty}}' \sum_{i=1}^{N} q_i \phi_{\boldsymbol{n}_k}(\boldsymbol{r}_i) = U_0 + \frac{1}{2}\sum_{k=2}^{\infty}\sum_{i=1}^{N} q_i \phi_{\boldsymbol{n}_k}(\boldsymbol{r}_i). \tag{3.10}$$

in the second equality we have identified $\boldsymbol{n}_1 = \boldsymbol{0}$ and written the summand separately by U_0. The potential $\phi_{\boldsymbol{n}_k}(\boldsymbol{x})$ is generated by the charges in the lattice cell $\boldsymbol{n}_k$ and can be expanded in spherical harmonics for $\boldsymbol{x}$ outside the cell[6]. $\phi_{\boldsymbol{n}_k}(\boldsymbol{r}_i)$ can be easy expressed by moving to spherical coordinates (r', θ', φ') with origin at $\boldsymbol{n}_k - \boldsymbol{r}_i$. In these coordinates

$$\phi'_{\boldsymbol{n}_k}(r', \theta', \varphi') = \sum_{l=0}^{\infty}\sum_{m=-l}^{l} \frac{4\pi}{2l+1} q_{lm} \frac{Y_{lm}(\theta', \varphi')}{r'^{l+1}} = \phi'(r', \theta', \varphi'). \tag{3.11}$$

The spherical harmonics Y_{lm} are chosen as in [81] and the multipole moments are

$$q_{lm} = \sum_{j=1}^{N} q_j (r'_{ji})^l Y^*_{lm}(\theta'_{ji}, \varphi'_{ji})$$

with $(r'_{ji}, \theta'_{ji}, \varphi'_{ji})$ being the coordinates of the charge q_j in the box $\boldsymbol{n}_k$ in the shifted coordinates, so they are the spherical coordinate components of the vector $\boldsymbol{r}_j - \boldsymbol{r}_i$.

Moving back to the original coordinates (r, θ, φ) we obtain

$$\begin{aligned} \phi_{\boldsymbol{n}_k}(r_i, \theta_i, \varphi_i) &= \phi'(r_{\boldsymbol{n}_k}, \pi - \theta_{\boldsymbol{n}_k}, \pi + \phi_{\boldsymbol{n}_k}) \\ \phi_{-\boldsymbol{n}_k}(r_i, \theta_i, \varphi_i) &= \phi'(r_{\boldsymbol{n}_k}, \theta_{\boldsymbol{n}_k}, \phi_{\boldsymbol{n}_k}) \end{aligned}$$

where $(r_{\boldsymbol{n}_k}, \theta_{\boldsymbol{n}_k}, \varphi_{\boldsymbol{n}_k})$ are the spherical coordinate components of $\boldsymbol{n}_k$.

Out of notational convenience, we change the order of the sum in eq. (3.10) s.t. $\boldsymbol{n}_k \mapsto -\boldsymbol{n}_k$[7]. Thus, we obtain

$$U = U_0 + \frac{1}{2}\sum_{k=2}^{\infty}\sum_{i=1}^{N} q_i \sum_{l=0}^{\infty}\sum_{m=-l}^{l} \frac{4\pi}{2l+1} \left(\sum_{j=1}^{N} q_j r'^{l}_{ji} Y^*_{lm}(\theta'_{ji}, \varphi'_{ji}) \right) \frac{Y_{lm}(\theta_{\boldsymbol{n}_k}, \varphi_{\boldsymbol{n}_k})}{r^{l+1}_{\boldsymbol{n}_k}},$$

where we have used $r'_{ij} = r_{ij} = \|\boldsymbol{r}_{ij}\|$ with $\boldsymbol{r}_{ij} = \boldsymbol{r}_i - \boldsymbol{r}_j$ Using the *addition theorem* for spherical harmonics this can be rewritten as [81]

$$U = U_0 + \frac{1}{2}\sum_{k=2}^{\infty}\sum_{i=1}^{N} q_i \sum_{l=0}^{\infty}\sum_{j=1}^{N} q_j r^l_{ji} \frac{P_l(\cos\gamma_{ji,\boldsymbol{n}_k})}{r^{l+1}_{\boldsymbol{n}_k}}. \tag{3.12}$$

[6] This is why we have separated the U_0 term.

[7] Reordering the sum can change the result as argued before but we haven't specified an order yet explicitly.

P_l are the Legendre polynomials and $\gamma_{ji,\boldsymbol{n}_k}$ is the angle between $\boldsymbol{r}_j - \boldsymbol{r}_i$ and $\boldsymbol{n}_k$. This expression is useful for discussing the convergence of U since one can restrict the discussion to the terms where $l \leq 2$. This follows from the following facts. First, note that $P_l(x) \in [-1, 1]$ for $x \in [-1, 1]$ and all $l \in \mathbb{N}_0$ [82]. So for all $r_{\boldsymbol{n}_k}$ with $\frac{r_{ij}}{r_{\boldsymbol{n}_k}} < 1$[8]

$$\sum_{l=3}^{\infty}\sum_{j=1}^{N} q_j r_{ji}^l \frac{P_l(\cos\gamma_{ji,\boldsymbol{n}_k})}{r_{\boldsymbol{n}_k}^{l+1}} \leq \frac{C_1}{r_{\boldsymbol{n}_k}}\sum_{l=3}^{\infty}\left(\frac{r_{ij}}{r_{\boldsymbol{n}_k}}\right)^l = \frac{C_1 r_{ij}^3}{r_{\boldsymbol{n}_k}^3(r_{\boldsymbol{n}_k} - r_{ij})}$$

where $C_1 \in \mathbb{R}$ is finite. The lattice sum over this result is then also absolutely convergent. To see this, recall the well-known fact[9] that

$$\sum_{\boldsymbol{n}\in\mathbb{Z}^3} \frac{1}{\|\boldsymbol{n}\|^s} < \infty \quad \forall s > 3.$$

For a more general lattice $\mathbb{L}$ we can write a similar expression by using the fact that $\boldsymbol{n} \in \mathbb{L}$ can be written as $\boldsymbol{A}\boldsymbol{m}$, where $\boldsymbol{A}$ is a matrix with the vectors $\boldsymbol{a}_1, \boldsymbol{a}_2$ and $\boldsymbol{a}_3$ as its columns and $\boldsymbol{m} \in \mathbb{Z}^3$

$$\sum_{\boldsymbol{n}\in\mathbb{L}} \frac{1}{\|\boldsymbol{n}\|^s} = \sum_{\boldsymbol{m}\in\mathbb{Z}^3} \frac{1}{\|\boldsymbol{A}\boldsymbol{m}\|^s}$$

It turns out that this sum is also convergent for $s > 3$ and can even be analytically continued to complex values of $s \neq 3$ [83].

The $l = 0$ term in eq. (3.12) vanishes because of charge neutrality and $P_0(x) = 1$. In order to finish the discussion about the convergence of eq. (3.12) we are left with the terms

$$\frac{1}{2}\sum_{k=2}^{\infty}\sum_{i=1}^{N} q_i \sum_{l=1}^{2}\sum_{j=1}^{N} q_j \|\boldsymbol{r}_{ji}\|^l \frac{P_l(\cos\gamma_{ji,\boldsymbol{n}_k})}{r_{\boldsymbol{n}_k}^{l+1}}.$$

This expression is not absolutely convergent anymore so the order of summation is relevant. We now choose an order where we treat $l = 1$ and $l = 2$ separately, so

$$\frac{1}{2}\sum_{i,j=1}^{N} q_i \sum_{l=1}^{2}\sum_{k=2}^{\infty} q_j \|\boldsymbol{r}_{ji}\|^l \frac{P_l(\cos\gamma_{ji,\boldsymbol{n}_k})}{r_{\boldsymbol{n}_k}^{l+1}}.$$

For $l = 1$ choose an order where for each even k, $\boldsymbol{n}_{k+1} = -\boldsymbol{n}_k$ and where for $k < l$ it holds $\|\boldsymbol{n}_k\| \leq \|\boldsymbol{n}_l\|$. Since $\cos\gamma_{ji,\boldsymbol{n}_k} = -\cos\gamma_{ji,-\boldsymbol{n}_k}$, and $P_l(-x) = (-1)^l P_l(x)$, the terms for $\boldsymbol{n}_k$ and $\boldsymbol{n}_{k+1}$ cancel in this order. Using once again that the terms $q_j\|\boldsymbol{r}_{ji}\|^l P_l(\cos\gamma_{ji,\boldsymbol{n}_k})$ are bounded and that $r_{\boldsymbol{n}_k}^{-2}$ is a monotonically decreasing null sequence one can use Dirichlet's test to show that this particular series converges.

According to Makov and Payne [80] Dirichlet's test can also be used to argue that the $l = 2$ terms are conditionally convergent, although this is not demonstrated explicitly. Thus, for a system with charge neutrality, the series in eq. (3.9) is conditionally convergent. A crucial feature of eq. (3.9) is that one sums over all charges in one lattice cell at once, i.e.

[8] This condition is not satisfied only by a finite number of lattice points $\boldsymbol{n}$.

[9] This can be shown by expressing the sum as an integral over a corresponding function on $\mathbb{R}^3 \setminus (-\frac{1}{2}, \frac{1}{2})^3$ that takes the value $\frac{1}{\|\boldsymbol{n}\|^s}$ on the subset $\boldsymbol{n} + (-\frac{1}{2}, \frac{1}{2})^3$ and bound it for $\|\boldsymbol{x}\| > \frac{\sqrt{3}}{2}$ by $\frac{1}{(\|\boldsymbol{x}\| - \frac{\sqrt{3}}{2})^s}$ which has a finite integral.

one groups the summands in such a way that they have a net charge of zero. If a different order is chosen, then even a neutrally charged system can be associated with a divergent series as it is demonstrated in [77].

We have only discussed the conditional convergence of the electrostatic energy for point charges in a cubic unit cell with length one. However, the arguments can easily be extended to more general charge distributions and also to non-cubic lattices since only the inversion symmetry of the lattice is used, which is even found in a triclinic cell.

The physical interpretation of the ambiguity of the electrostatic energy is that for any material with finite size, all dipole moments of all cells have a non-negligible effect on the potential in the original cell, including the cells at the surface. The construction of a system under PBCs like it is done in eq. (3.9) can then be considered as the limit of making a finite system infinitely large. By choosing a particular order of summation one selects a specific shape that the material corresponds to while making it infinitely large [79, 80, 84]. This will be discussed in more detail in the next subsection. Another perspective to argue physically about the ambiguity of the potential energy is that for a system that consists of finitely many copies of the unit cell, it is clear which choice of the unit cell was made by looking at the boundary of the material. If infinitely many copies exist, then there are infinitely many equivalent choices of the unit cell that can describe the system. However, the naive way of defining the dipole moment of a unit cell

$$\boldsymbol{p} = \int_{\Omega} \mathrm{d}^3\boldsymbol{x}\ \boldsymbol{x}\rho(\boldsymbol{x}) \tag{3.13}$$

is dependent on the choice of Ω. When choosing a specific order of summation (which also implies a choice of unit cell) then at least for spherical ordering of the summation produces a term containing $\boldsymbol{p}$ [79, 80]. This shows again the ambiguity of the electrostatic energy.

3.4.3 Ewald Summation

The Ewald summation is a method for computing the potential energy associated with a unit cell under PBCs. So it is a technique to evaluate an expression as given in eq. (3.8). Initially, it was introduced by Ewald in 1921 to compute the electrostatic potential energy associated with the unit cell of a crystal [85]. Later it started to be used in atomistic calculations with long-range interactions under PBCs. Most often one encounters the Ewald summation in combination with the pair potential of two point charges q_i, q_j with $u_{ij}(r_{ij}) = \frac{q_i q_j}{r_{ij}}$. However, the Ewald summation can also be extended to other potentials like the ones describing charge-dipole, dipole-dipole and other interactions [79]. It transforms the electrostatic energy into an absolutely and especially fast converging sum. This is of course only possible if an order of summation is implicitly imposed. The fast convergence makes the Ewald summation interesting for numerical calculations since a truncation of the sum after a few summands can still achieve relatively high accuracy.

Point Charges

Particularly insightful studies about the Ewald summation have been conducted by de Leeuw et al. [79] and Smith and Rowlinson [86]. It is noteworthy that the former have only considered cubic lattices whereas the latter studied the Ewald summation for general lattices. A comprehensible summary of the work of Smith and Rowlinson was written by Darden [87].

De Leeuw et al. start by choosing a particular order of summation over $\mathbb{Z}^3$. The following discussion can easily be generalized to cubic lattices $\mathbb{L}_\mathrm{c}$ with a different length than 1. For a general sum over $\mathbb{Z}^3$

$$\sum_{\boldsymbol{n}\in\mathbb{Z}^3} a_{\boldsymbol{n}}$$

a summation over spherical shells may be written as

$$\sum_{m=0}^{\infty} \sum_{\substack{\boldsymbol{n}\in\mathbb{Z}^3:\\ \|\boldsymbol{n}\|=r_m}} a_{\boldsymbol{n}},$$

where the r_m are an ordered sequence[10] of all radii that can be written as $\|\boldsymbol{n}\|$. He then argues that one can introduce so-called convergence factors f with $f(m,0)=1$ for all m that make

$$\sum_{\boldsymbol{n}\in\mathbb{Z}^3} a_{\boldsymbol{n}} f(m(\boldsymbol{n}), s)$$

absolutely convergent for $s>0$. Here, $m(\boldsymbol{n})\in\mathbb{N}$. For their properties, we refer the reader to the original work [79]. In particular, if $f(m,s)=\tilde{f}(r_m,s)\in\mathbb{R}$ it is then shown that

$$\sum_{m=0}^{\infty} \sum_{\substack{\boldsymbol{n}\in\mathbb{Z}^3:\\ \|\boldsymbol{n}\|=r_m}} a_{\boldsymbol{n}} = \lim_{s\to\infty} \sum_{\boldsymbol{n}\in\mathbb{Z}^3}^{\infty} a_{\boldsymbol{n}} f(m(\boldsymbol{n}), s).$$

Using $f(m,s)=e^{-sm(\boldsymbol{n})^2}$ for the series in eq. (3.9) the result is then computed explicitly and corresponds to the Ewald summation for spherical ordering[11] [79, 86]

$$\begin{aligned} U^{\mathrm{pc}} = &\frac{1}{2}{\sum_{\boldsymbol{n}\in\mathbb{L}}}' \sum_{i,j=1}^{N} q_i q_j \frac{\operatorname{erfc}\left(\frac{\|\boldsymbol{r}_{ij}+\boldsymbol{n}\|}{\sqrt{2}\eta}\right)}{\|\boldsymbol{r}_{ij}+\boldsymbol{n}\|} \\ &+ \frac{1}{2\pi V} \sum_{\boldsymbol{k}\neq\boldsymbol{0}} \frac{\exp\left(-2\pi^2\eta^2\|\boldsymbol{k}\|^2\right)}{\|\boldsymbol{k}\|^2} |S(\boldsymbol{k})|^2 \\ &- \frac{1}{\sqrt{2\pi}\eta} \sum_{i=1}^{N} q_i^2 + \frac{2\pi}{3V} \underbrace{\left\| \sum_{i}^{N} q_i \boldsymbol{r}_i \right\|^2}_{=\|\boldsymbol{p}\|^2}. \end{aligned} \tag{3.14}$$

Here we have stated the result for a general cubic lattice $\mathbb{L}_\mathrm{c}$ with side length L and unit cell volume $V=L^3$. The vectors $\boldsymbol{k}$ are the lattice vectors of the reciprocal lattice defined by $\tilde{\boldsymbol{a}}_1, \tilde{\boldsymbol{a}}_2$ and $\tilde{\boldsymbol{a}}_3$ which satisfy $\boldsymbol{a}_i \cdot \tilde{\boldsymbol{a}}_j = \delta_{ij}$. The parameter $\eta\in\mathbb{R}_{>0}$ can be chosen freely and affects the rate of convergence of the sums over $\boldsymbol{n}$ and $\boldsymbol{k}$ in opposite ways. Thus, it is often used to tune the computational performance of the Ewald algorithm. Finally, $S(\boldsymbol{k})$ is the structure factor defined by

$$S(\boldsymbol{k}) = \sum_{i=1}^{N} q_i e^{i2\pi\boldsymbol{k}\cdot\boldsymbol{r}_i}.$$

[10] The value of this series is now uniquely determined since the inner sum is finite for any r_m and the order of $\{r_m\}_{m\in\mathbb{N}_{\geq 0}}$ is specified.

[11] We write this result for a general lattice $\mathbb{L}$ since this was later generalized by [86]. Additionally, they even derived the result for ellipsoidal ordering.

Note that because of the structure factor the reciprocal space sum scales only with $\mathcal{O}(N)$ compared to the real space sum that scales with $\mathcal{O}(N^2)$.

The approach by Smith and Rowlinson starts by defining a region in $\mathbb{R}^3$ bounded by a hypersurface defined via

$$P(\boldsymbol{r}) = 0.$$

For any $m \in \mathbb{N}$

$$P_m(\boldsymbol{r}) = P(\boldsymbol{r}/m) = 0$$

is the surface of a, in general, larger region of the same shape. Those regions are notated as P and P_m respectively. One can construct a large but finite system by considering replicas of the unit cell at all points $\boldsymbol{n} \in \mathbb{L} \cap P_m$. It is then shown that the electrostatic energy associated with the unit cell in this system

$$U^{\mathrm{pc}}(P_m) = \frac{1}{2} {\sum_{\boldsymbol{n}\in\mathbb{L}\cap P_m}}' \sum_{i,j=1}^{N} \frac{q_i q_j}{\|\boldsymbol{r}_i - \boldsymbol{r}_j + \boldsymbol{n}\|}$$

can be written as a sum of two terms [86],

$$U^{\mathrm{pc}}(P_m) = U_0 + J(\boldsymbol{p}, P_m). \tag{3.15}$$

Here, U_0 is not dependent on P_m whereas J is dependent on the dipole moment $\boldsymbol{p}$ defined in eq. (3.13) and the shape of the system P_m. Also, $J(\boldsymbol{0}, P_m) = 0$ for any P_m. As mentioned before $\boldsymbol{p}$ is dependent on the choice of the unit cell. This is not a contradiction to eq. (3.15) since for finitely many copies of the unit cell, different choices of the cell lead to different systems[12]. Finally, provided that $\lim_{m\to\infty} U^{\mathrm{pc}}(P_m)$ exists, the electrostatic energy under PBCs with spherical summation order takes the form

$$U^{\mathrm{pc}}(P) = \lim_{m\to\infty} U^{\mathrm{pc}}(P_m) = U_0 + J(\boldsymbol{p}, P).$$

Smith and Rowlinson [86] showed explicitly that for $P = S^2$ being the unit sphere one obtains

$$J(\boldsymbol{p}, S^2) = \frac{2\pi}{3V}\|\boldsymbol{p}\|^2.$$

Comparing this with eq. (3.14) it becomes clear that the last term is precisely the shape-dependent part of U^{pc} whereas the leading terms hold for any order of summation.

For an explicit derivation of eq. (3.14) we refer to [79, 86, 87]. However, we mention how the parameter η and the summation over the reciprocal lattice are introduced. Starting from the definition of the gamma function

$$\Gamma(u) = \int_0^\infty \mathrm{d}t\; t^{u-1}e^{-t} = \lambda^u \int_0^\infty \mathrm{d}z\; z^{u-1}e^{-\lambda z}$$

it follows for $\lambda = \|\boldsymbol{r}\|^2$ and $u = \frac{1}{2}$

$$\frac{1}{\|\boldsymbol{r}\|} = \frac{1}{\sqrt{\pi}} \int_0^\infty \mathrm{d}t\; \frac{1}{\sqrt{t}} e^{-t\|\boldsymbol{r}\|^2}, \tag{3.16}$$

[12] This can easily be seen by considering the outermost charges that appear at the surface.

where $\Gamma(\frac{1}{2}) = \sqrt{\pi}$ is used. The integral is then split into two parts

$$\underbrace{\left(\int_0^{\frac{1}{2\eta^2}} \dots\right)}_{I_1(\|\boldsymbol{r}\|)} + \underbrace{\left(\int_{\frac{1}{2\eta^2}}^{\infty} \dots\right)}_{I_2(\|\boldsymbol{r}\|)}.$$

With the simple substitution $z = \sqrt{t}\|\boldsymbol{r}\|$, I_2 can be rewritten directly in terms of the complementary error function

$$I_2(\|\boldsymbol{r}\|) = \int_{\frac{1}{2\eta^2}}^{\infty} \mathrm{d}t \, \frac{1}{\sqrt{t}} e^{-t\|\boldsymbol{r}\|^2} = \frac{\operatorname{erfc}\left(\frac{\|\boldsymbol{r}\|}{\sqrt{2}\eta}\right)}{\|\boldsymbol{r}\|}.$$

In order to bring the sum $\sum_{\boldsymbol{n}} I_1(\|\boldsymbol{r}+\boldsymbol{n}\|)$ into a manageable shape a little more work is needed. One starts by expressing the Gaussian inside the integral as the Fourier transform of the Gaussian in reciprocal space

$$e^{-t\|\boldsymbol{r}+\boldsymbol{n}\|^2} = \left(\frac{\pi}{t}\right)^{\frac{3}{2}} \int_{\mathbb{R}^3} \mathrm{d}^3\boldsymbol{u} \; e^{-\frac{\pi^2\boldsymbol{u}^2}{t} - 2\pi i \boldsymbol{u}\cdot(\boldsymbol{r}+\boldsymbol{n})}$$

It is then used that instead of integrating $\boldsymbol{u}$ over the complete reciprocal space one can reduce the integral only over the reciprocal unit cell and sum over all reciprocal lattice points $\boldsymbol{k}$. The exact details are rather technical so we refer the reader to Smith and Rowlinson [86] and Darden [87]. Lastly, it is noted that the sum over $I_1(\|\boldsymbol{r}+\boldsymbol{n}\|)$ manifests the shape-dependency.

In practice the shape-dependent term $J(\boldsymbol{p}, S)$ is not always included. The reasons for that vary. There exist derivations of the Ewald summation that are physically more intuitive by adding Gaussian screening charges with width η on top of the actual charges in the system. The additional interaction energy is removed afterwards. For more details, see Frenkel and Smit [73]. However, this approach fails to describe the shape-dependent part and thus, the extra term is often deliberately set to zero. Interestingly, in case of the spherical summation order, it was shown by Smith [88] that when deriving the electrostatic energy under PBCs by taking the limit $\lim_{m\to\infty} U^{\mathrm{pc}}(P_m)$, one can describe the large but finite spherical system P_m to be embedded inside a dielectric with dielectric constant ε. If the dipole moment $\boldsymbol{p}$ of the unit cell Ω does not vanish, the electrostatic field of the dipole moment of the system polarizes the surrounding dielectric and thus induces a reaction potential ϕ_{Pol}. Including this potential in the computation of the electrostatic energy, the shape-dependent term takes the form [87]

$$J_{\mathrm{Pol}}(\boldsymbol{p}, S) = \frac{1}{2\varepsilon+1}\frac{2\pi}{V}\|\boldsymbol{p}\|^2.$$

In the limit $\varepsilon \to \infty$ this contribution vanishes and we are left with eq. (3.14) without the last term. This limit is sometimes called "tinfoil" boundary condition [87].

Gaussian Charges

In this work, the atomic charge distributions are modeled as Gaussian charges with width σ_i for atom i, see eq. (2.30). Thus it is necessary to adjust the Ewald summation. Since the pair potential between Gaussian charges behaves asymptotically like the pair potential between point charges, see eq. (3.1), we expect that the derivations by de Leeuw et al. [79] and

Smith and Rowlinson [86] only need small modifications to account for the smeared charge distribution. The biggest difference is only that for Gaussian charges, the self-interaction is finite and can thus be included. Fortunately, this has already been worked out by Gingrich and Wilson [89], who followed the derivation by de Leeuw et al. [79]. The formula for Gaussian charges with different widths has been worked out by Kiss et al. [72]. The electrostatic energy takes the form[13]

$$\begin{aligned} U^{\mathrm{G}} &= \frac{1}{2}\sum_{\boldsymbol{n}\in\mathbb{L}}{}' \sum_{i,j=1}^{N} q_i q_j \frac{\operatorname{erfc}\left(\frac{\|\boldsymbol{r}_{ij}+\boldsymbol{n}\|}{\sqrt{2}\eta}\right) - \operatorname{erfc}\left(\frac{\|\boldsymbol{r}_{ij}+\boldsymbol{n}\|}{\sqrt{2}\gamma_{ij}}\right)}{\|\boldsymbol{r}_{ij}+\boldsymbol{n}\|} \\ &\quad + \frac{1}{2\pi V}\sum_{\boldsymbol{k}\neq\boldsymbol{0}} \frac{\exp\left(-2\pi^2\eta^2\|\boldsymbol{k}\|^2\right)}{\|\boldsymbol{k}\|^2}|S(\boldsymbol{k})|^2 \\ &\quad + \left[\sum_{i=1}^{N} q_i^2\left(\frac{1}{2\sqrt{\pi}\sigma_i} - \frac{1}{\sqrt{2\pi}\eta}\right)\right] + \frac{2\pi}{3V}\left\|\sum_{i=1}^{N} q_i\boldsymbol{r}_i\right\|^2 \\ &= U^{\mathrm{pc}} - \frac{1}{2}\sum_{\boldsymbol{n}\in\mathbb{L}}{}' \sum_{i,j=1}^{N} \frac{\operatorname{erfc}\left(\frac{\|\boldsymbol{r}_{ij}+\boldsymbol{n}\|}{\sqrt{2}\gamma_{ij}}\right)}{\|\boldsymbol{r}_{ij}+\boldsymbol{n}\|} + \sum_{i=1}^{N}\frac{q_i^2}{2\sqrt{\pi}\sigma_i} \end{aligned} \tag{3.17}$$

where we have used the definition for γ_{ij} as in eq. (3.3) again. Note that for Gaussian charges

$$\boldsymbol{p} = \int_{\mathbb{R}^3} \mathrm{d}^3\boldsymbol{x}\ \sum_{i=1}^{N}\rho_i(\boldsymbol{x})\boldsymbol{x} = \sum_{i=1}^{N} q_i\boldsymbol{r}_i.$$

Thus the potential U^{G} only differs from the energy of the corresponding point charges by a real space correction and the self-interaction contribution. Already in the derivation it becomes clear that the replacement of the point charges with Gaussian ones solely affects the real space sum [89]. Since the shape-dependent term $J(\boldsymbol{p}, P)$ only appears in evaluating the reciprocal space sum, the results are also valid for Gaussian distributed charges. In the original derivation of Gingrich and Wilson [89] and when generalizing it to different Gaussian charge widths, the Ewald parameter η is most often chosen such that $\eta > \sigma$ or $\eta > \max_{ij}\gamma_{ij}$ for different Gaussian widths, because in contrast to eq. (3.16) the integral ranges from 0 to $\frac{1}{2\gamma_{ij}^2}$ and usually one splits the integral somewhere inside this interval. However, there is no problem in choosing $\eta < \max_{ij}\gamma_{ij}$ provided it is still positive.

Using Ewald Summation in Charge Equilibration

In section 3.2 we discussed how minimizing the potential energy with respect to charges can be transformed into a linear system of equations. Note that in this discussion we did not consider PBCs. With eqs. (3.14) and (3.17) now at hand, one can also express U^{Qeq} under PBCs as a quadratic equation in $\boldsymbol{q}$. This becomes obvious when rewriting

$$\begin{aligned} |S(\boldsymbol{k})|^2 &= \sum_{i,j=1}^{N} q_i q_j e^{i2\pi\boldsymbol{k}\cdot\boldsymbol{r}_i} e^{-i2\pi\boldsymbol{k}\cdot\boldsymbol{r}_j} \\ &= \sum_{i,j=1}^{N} q_i q_j \cos\left(2\pi\boldsymbol{k}\cdot\boldsymbol{r}_{ij}\right) \end{aligned}$$

[13] In [89] there is a wrong sign at the shape-dependent term

Plugging this in eq. (3.17) we can write the electrostatic energy for Gaussian charges under PBCs for spherical summation order as a matrix equation

$$U^{\mathrm{G}} = \frac{1}{2}\boldsymbol{q}^{\intercal}\boldsymbol{B}\boldsymbol{q},$$

with

$$\begin{aligned}(\boldsymbol{B})_{ij} = {} & \frac{1}{\pi V}\sum_{\boldsymbol{k}\neq\boldsymbol{0}} \frac{\exp\left(-2\pi^2\eta^2\|\boldsymbol{k}\|^2\right)}{\|\boldsymbol{k}\|^2}\cos\left(2\pi\boldsymbol{k}\cdot\boldsymbol{r}_{ij}\right) \\ & + {\sum_{\boldsymbol{n}\in\mathbb{L}}}' \frac{\operatorname{erfc}\left(\frac{\|\boldsymbol{r}_{ij}+\boldsymbol{n}\|}{\sqrt{2}\eta}\right) - \operatorname{erfc}\left(\frac{\|\boldsymbol{r}_{ij}+\boldsymbol{n}\|}{\sqrt{2}\gamma_{ij}}\right)}{\|\boldsymbol{r}_{ij}+\boldsymbol{n}\|} + \frac{4\pi}{3V}\boldsymbol{r}_i\cdot\boldsymbol{r}_j \\ & + \delta_{ij}\left(\frac{1}{\sqrt{\pi}\sigma_i} - \sqrt{\frac{2}{\pi\eta^2}}\right).\end{aligned} \tag{3.18}$$

3.4.4 Electrostatic Energy on the Torus

In section 3.4.1 we have already talked about the periodization of the charge distribution in order to obtain the corresponding function on the torus $\mathbb{T}^3$. The Ewald summation of the electrostatic energy between Gaussian distributed charges transforms the expression[14]

$$U^{\mathrm{G}} = \frac{1}{2}\sum_{\boldsymbol{n}\in\mathbb{L}}\sum_{i,j=1}^{N}\frac{q_i q_j}{\|\boldsymbol{r}_{ij}+\boldsymbol{n}\|}\operatorname{erf}\left(\frac{\|\boldsymbol{r}_{ij}+\boldsymbol{n}\|}{\sqrt{2}\gamma_{ij}}\right) + \sum_{i=1}^{N}\frac{q_i^2}{2\sqrt{\pi}\sigma_i}$$

into an absolutely and rapidly converging sum by choosing an order of summation as discussed before. This can be interpreted as computing

$$U^{\mathrm{G}} = \frac{1}{2}\int_{\mathbb{R}^3}\mathrm{d}^3\boldsymbol{x}\ \rho(\boldsymbol{x})\sum_{\boldsymbol{n}\in\mathbb{L}}\phi(\boldsymbol{x}+\boldsymbol{n}),$$

where ρ is the non-periodic charge distribution generated in $\mathbb{R}^3$ by the Gaussian charges in the unit cell Ω, and ϕ is the potential generated by those charges. The sum can be seen as the periodization of ϕ. However, as we have seen this sum is in general not absolutely convergent. For Gaussian charges, the periodization is absolutely convergent though (on any triclinic lattice $\mathbb{L}$), as it is demonstrated in appendix C.

For simplicity we will restrict the following discussion again to the case $\mathbb{L} = \mathbb{Z}^3$. Note that any sum $\sum_{\boldsymbol{n}\in\mathbb{L}}$ can be rewritten as a sum $\sum_{\boldsymbol{m}\in\mathbb{Z}^3}$ when substituting $\boldsymbol{n} = \boldsymbol{A}\boldsymbol{m}$. The matrix $\boldsymbol{A}\in\mathbb{R}^{3\times3}$ has the lattice vectors $\boldsymbol{a}_1, \boldsymbol{a}_2$ and $\boldsymbol{a}_3$ as its columns.

In section 3.4.1 we discussed that PBCs give the unit cell the topology of the torus $\mathbb{T}^3$. This insight can be used to attempt a different derivation of the electrostatic energy. Since ρ can be periodized without any problem compared to ϕ, a different approach would be to obtain a potential $\bar{\phi}$ defined on the torus by solving the Poisson equation

$$\Delta\bar{\phi} = -4\pi\bar{\rho}$$

[14] U^{G} is not strictly the same as in the previous subsection because here we do not specify the order of summation.

on the torus, where $\bar{\rho}$ is the periodized charge distribution. As mentioned before the Laplace-Beltrami operator becomes the usual Laplace operator when using appropriate coordinates. Mamode [90] showed how to construct the fundamental solution of the Laplace equation on the flat two-dimensional torus. Here we choose a different way because of the particular simplicity that arises when dealing with Gaussian charges. As it is often done, this partial differential equation can be transformed into an algebraic equation when switching to Fourier space. The toroidal Fourier transform is the map [75]

$$\begin{aligned}\mathcal{F}_{\mathbb{T}^3}:\ \mathcal{C}^\infty &\to \mathcal{S}(\mathbb{Z}^3)\\ f(\boldsymbol{x}) &\mapsto \hat{f}(\boldsymbol{k}) = \int_{\mathbb{T}^3} \mathrm{d}^3\boldsymbol{x}\ e^{-i2\pi\boldsymbol{x}\cdot\boldsymbol{k}} f(\boldsymbol{x}).\end{aligned}$$

$\mathcal{S}(\mathbb{Z}^3)$ is the Schwartz space on $\mathbb{Z}^3$ consisting of rapidly decaying functions that map from $\mathbb{Z}^3 \to \mathbb{C}$. A precise definition can be found in the book by Ruzhansky and Turunen [75]. The inverse transformation is then [75]

$$\mathcal{F}^{-1}_{\mathbb{T}^3}[\hat{f}](\boldsymbol{x}) = \sum_{\boldsymbol{k}\in\mathbb{Z}^3} e^{i2\pi\boldsymbol{k}\cdot\boldsymbol{x}} \hat{f}(\boldsymbol{k}).$$

The Fourier transform of $\bar{\rho}$ corresponding to the Gaussian distributed charges can easily be computed:

$$\begin{aligned}\hat{\rho}(\boldsymbol{k}) &= \int_{\mathbb{T}^3} \mathrm{d}^3\boldsymbol{x}\ e^{-i2\pi\boldsymbol{k}\cdot\boldsymbol{x}} \sum_{\boldsymbol{n}\in\mathbb{Z}^3}\sum_{i=1}^{N} \frac{q_i}{(2\pi\sigma_i^2)^{\frac{3}{2}}} e^{-\frac{\|\boldsymbol{x}+\boldsymbol{n}+\boldsymbol{r}_i\|^2}{2\sigma_i^2}}\\ &= \sum_{i=1}^{N} \int_{\mathbb{R}^3} \mathrm{d}^3\boldsymbol{x}\ e^{-i2\pi\boldsymbol{k}\cdot\boldsymbol{x}} \frac{q_i}{(2\pi\sigma_i^2)^{\frac{3}{2}}} e^{-\frac{\|\boldsymbol{x}+\boldsymbol{n}+\boldsymbol{r}_i\|^2}{2\sigma_i^2}}\\ &\ \ \vdots\\ &= \sum_{i=1}^{N} q_i e^{-2\pi^2\sigma_i^2\|\boldsymbol{k}\|^2 - i2\pi\boldsymbol{k}\cdot\boldsymbol{r}_i}.\end{aligned}$$

The Laplace equation translates to the following equation in Fourier space

$$\pi\|\boldsymbol{k}\|^2\hat{\phi}(\boldsymbol{k}) = \hat{\rho}(\boldsymbol{k}).$$

For $\boldsymbol{k} = \boldsymbol{0}$ this equation does not specify $\hat{\phi}(\boldsymbol{0})$, which is a constant C_0. It only encodes the charge neutrality requirement since $\hat{\rho}(\boldsymbol{0}) = \int_{\mathbb{R}^3}\mathrm{d}^3\boldsymbol{x}\ \rho(\boldsymbol{x})$. So we obtain

$$\begin{aligned}\bar{\phi}(\boldsymbol{x}) &= \left(\sum_{\boldsymbol{k}\neq\boldsymbol{0}} e^{i2\pi\boldsymbol{k}\cdot\boldsymbol{x}} \frac{\hat{\rho}(\boldsymbol{k})}{\pi\|\boldsymbol{k}\|^2}\right) + C_0\\ &= \left(\sum_{\boldsymbol{k}\neq\boldsymbol{0}}\sum_{i=1}^{N} \frac{q_i}{\pi\|\boldsymbol{k}\|^2} e^{-2\pi^2\sigma_i^2\|\boldsymbol{k}\|^2 + i2\pi\boldsymbol{k}\cdot(\boldsymbol{x}-\boldsymbol{r}_i)}\right) + C_0.\end{aligned}$$

Analogously to the computation in $\mathbb{R}^3$ we compute the electrostatic energy via

$$\begin{aligned}
U &= \frac{1}{2}\int_{\mathbb{T}^3} \mathrm{d}^3\boldsymbol{x}\ \bar{\rho}(\boldsymbol{x})\bar{\phi}(\boldsymbol{x}) \\
&= \frac{1}{2}\int_{\mathbb{R}^3} \mathrm{d}^3\boldsymbol{x}\ \rho(\boldsymbol{x})\bar{\phi} \\
&= \frac{1}{2}\int_{\mathbb{R}^3} \mathrm{d}^3\boldsymbol{x}\ \left(\sum_{i=1}^{N} \frac{q_i}{(2\pi\sigma_i^2)^{\frac{3}{2}}} e^{-\frac{\|\boldsymbol{x}-\boldsymbol{r}_i\|^2}{2\sigma_i^2}}\right) \\
&\quad \cdot \left(\left[\sum_{\boldsymbol{k}\neq\boldsymbol{0}}\sum_{j=1}^{N} \frac{q_i}{\pi\|\boldsymbol{k}\|^2} e^{-2\pi^2\sigma_i^2\|\boldsymbol{k}\|^2 + i2\pi\boldsymbol{k}\cdot(\boldsymbol{x}-\boldsymbol{r}_i)}\right] + C_0\right) \\
&= \frac{1}{2}\sum_{i,j=1}^{N} \frac{q_i q_j}{(2\pi\sigma_i^2)^{\frac{3}{2}}} \int_{\mathbb{R}^3} \mathrm{d}^3\boldsymbol{x}\ e^{-\frac{\|\boldsymbol{x}-\boldsymbol{r}_i\|^2}{2\sigma_j^2}} \sum_{\boldsymbol{k}\neq\boldsymbol{0}} \frac{1}{\pi\|\boldsymbol{k}\|^2} e^{-2\pi^2\sigma_j^2\|\boldsymbol{k}\|^2 + i2\pi\boldsymbol{k}\cdot(\boldsymbol{x}-\boldsymbol{r}_j)}.
\end{aligned}$$

Note that the undetermined constant C_0 drops out at the end as a consequence of charge neutrality. For further computation it is necessary to interchange the integral and the sum over $\boldsymbol{k}$. A rigorous treatment of this step can be found in appendix C. Evaluating the integral one arrives at

$$U = \frac{1}{2}\sum_{i,j=1}^{N} q_i q_j \sum_{\boldsymbol{k}\neq\boldsymbol{0}} \frac{1}{\pi\|\boldsymbol{k}\|^2} e^{-2\pi^2\|\boldsymbol{k}\|^2(\sigma_i^2+\sigma_j^2) - i2\pi\boldsymbol{k}\cdot(\boldsymbol{r}_j-\boldsymbol{r}_i)}.$$

Interestingly, this result is absolutely convergent[15] without any manipulations like the ones that appear in the Ewald summation. To the author's knowledge, it is not clear how this result can be related to eq. (3.17). The discussion in section 3.4.3 about defining a surface P_m in real space and taking the limit $m \to \infty$ is not transferable to this approach here since everything is formulated on the torus $\mathbb{T}^3$, which is a compact manifold without a boundary. Thus, a discussion about surface terms and surrounding dielectrics is impossible. The same computation for point charges is not straightforward because in that case the self-interaction, which is automatically included when formulating the electrostatic interaction on the torus, leads to a divergent expression for the energy.

[15] In appendix C it is shown that the lattice sum over a Gaussian function is convergent which can be used for a comparison test.

Chapter 4

Implementation Details

4.1 Truncation of the Ewald Sum

When investigating a system without PBCs, the charge equilibration described in section 3.1 is straightforward. One constructs the $\boldsymbol{A}$ matrix as stated in eq. (3.4) and solves eq. (3.6). In contrast, if PBCs are applied, it is necessary to approximate the $\boldsymbol{A}$ matrix in eq. (3.4) by truncating the real-space and Fourier space sums. For this it is common to introduce spherical cutoffs $r_{r,\mathrm{c}}$, $r_{k,\mathrm{c}}$ such that all real-space terms with $\|\boldsymbol{r}_{ij} + \boldsymbol{n}\| > r_{r,\mathrm{c}}$ or Fourier space terms satisfying $\|\boldsymbol{k}\| > r_{k,\mathrm{c}}$ are not included. The truncation error propagates to the computation of the electrostatic energy and forces acting on the atoms. It is not only dependent on $r_{r,\mathrm{c}}$ and $r_{k,\mathrm{c}}$ but also on the Ewald summation parameter η. From eq. (3.17) it is clear that increasing η allows smaller values of $r_{k,\mathrm{c}}$ but requires larger values of $r_{r,\mathrm{c}}$ in order to prevent an increase in the error. In section 4.1.2 we will discuss a particular scheme for determining a good choice of η. But first, we will see η as given and decide how to choose the cutoffs as a function of it.

4.1.1 Estimating the Truncation Error

The problem of choosing the cutoffs of the real-space and Fourier space sums has been discussed several times. Since this work is strongly related to the work of Ko et al. [14] who also had to implement a solution to this problem, we mention here that they followed the approach published by Jackson and Catlow [91]. The idea is to discard all terms in the sum for which the expression without the charges is smaller in magnitude than a certain threshold $\frac{A}{r_0}$. Here, A is a dimensionless parameter that specifies the accuracy and r_0 is a characteristic length of the lattice. This approach has the advantage that it leads to the simple formulas [91]

$$r_{r,\mathrm{c}} = \eta\sqrt{-2\ln A} \quad \text{and} \quad r_{k,\mathrm{c}} = \frac{1}{2\pi\eta}\sqrt{-2\ln A}.$$

It relies on assumptions like a similar magnitude of the charges and a homogenous distribution of the charges which may not satisfied in the system of interest. The biggest disadvantage is, however, that it is not obvious how the accuracy parameter A is related to physical quantities like the electrostatic energy or forces. Furthermore, it is a pessimistic estimate since it does not account for canceling contributions caused by alternating signs of the charges. Also, note that the derivation of this result assumes point charges instead of Gaussian charges.

This is why in this work we have chosen to use the method presented by Kolafa and Perram [92] with a minor modification included by Petersen [93]. A very similar method is presented by Wells and Chaffee [94]. The idea is to derive estimates of the standard deviation of either the energy that is contributed by an individual charge or alternatively of the force acting on a charge. It can be derived by assuming that each charge is distributed uniformly in the simulation box [93]. Depending on the system this can be a crude estimate, so verifying the error estimate by checking the convergence of the forces for different precision values is recommended. An advantage compared to method [91] is that it respects the cancellation effects of the alternating signs of the charges. Originally, this method was derived for a system of point charges for which the charges are known at the time when the Ewald sum is evaluated. However, in this work, we make use of Gaussian charges and need to compute the Ewald sum in order to get an expression of the electrostatic energy as a function of the charges. A rough estimate for the upper limit of the charges q_i is $q_{\max}$ which is the largest value in magnitude of all charges that appear in the training data. This quantity is easy to determine. Since a good training data set should cover enough configurations such that in a real-world simulation no extrapolations[1] occur, this is a quite robust estimate.

In this work, we define the accuracy of the Ewald sum for charge i as the error

$$\Delta \boldsymbol{F}_i = \boldsymbol{F}_{i,\text{trunc}} - \boldsymbol{F}_{i,\text{exact}},$$

where $\boldsymbol{F}_{i,\text{trunc}}$ is the force on charge i resulting from the truncated sum and $\boldsymbol{F}_{i,\text{exact}}$ is the one from the complete sum. The standard deviation of this quantity is then written as [91]

$$\sigma^2(\Delta \boldsymbol{F}_i) = \frac{1}{N} \sum_{i=1}^{N} \Delta \boldsymbol{F}_i^2.$$

It is argued by [91] that this is often a good error estimate. As already mentioned, we are going to bound $|q_i|$ by $q_{\max}$. Thus, we bound all estimates $\sigma(\Delta \boldsymbol{F}_i)$ by a single one that is independent of the index i

$$\sigma_F \geq \sigma(\Delta \boldsymbol{F}_i).$$

This can be divided into the real-space error $\sigma_{F,r}$ and the Fourier space error $\sigma_{F,k}$. For the Fourier space error we can use directly[2] the result of [92] or [93] and bound it from above by using $q_{\max}$. Following the result of the latter work, which has an additional factor of $\sqrt{\frac{\pi}{2}}$ one obtains

$$\sigma_{F,k} = \frac{2q_{\max}^2}{\eta} \sqrt{\frac{N}{2\pi r_{k,\text{c}} V}} e^{-\frac{(2\pi r_{k,\text{c}} \eta)^2}{2}}, \tag{4.1}$$

where V is the volume of the unit cell Ω. As pointed out in [92], this approximation is only valid for $2\pi\eta^2 r_{k,\text{c}} < \frac{L}{2}$ for a cubic unit cell with length L. For a general triclinic cell this would translate to $2\pi\eta^2 r_{k,\text{c}} < \frac{\|\boldsymbol{A}\|_2}{2}$, where $\|\boldsymbol{A}\|_2$ is the spectral norm of the matrix consisting of the column vectors $\boldsymbol{a}_i$.

Considering the real-space contribution, we can show that the error estimate for point charges is also an upper bound for the error estimate of Gaussian charges, provided $\eta \geq$

[1] Technically speaking, we say that a NN is extrapolating, if the configuration it is evaluating is not contained in the convex hull of the training data set. This is not easy to determine in practice though.

[2] This follows from the fact that the Ewald summation of Gaussian charges only differs in the real space part.

$\max_{ij}\gamma_{ij}$. The force on charge i associated with the real-space part can be written as

$$\boldsymbol{F}_{i,\mathrm{r}} = \frac{q_i}{2}\sum_{\boldsymbol{n}}{}' \sum_{j=1}^{N} q_j \frac{\boldsymbol{r}_{ij}+\boldsymbol{n}}{\|\boldsymbol{r}_{ij}+\boldsymbol{n}\|^2} \underbrace{\left(a'(\|\boldsymbol{r}_{ij}+\boldsymbol{n}\|,\gamma_{ij}) - \frac{a(\|\boldsymbol{r}_{ij}+\boldsymbol{n}\|,\gamma_{ij})}{\|\boldsymbol{r}_{ij}+\boldsymbol{n}\|}\right)}_{=:b(\|\boldsymbol{r}_{ij}+\boldsymbol{n}\|,\gamma_{ij})},$$

where we have introduced

$$\begin{aligned} a(r,\gamma_{ij}) &= \operatorname{erfc}\left(\frac{r}{\sqrt{2}\eta}\right) - \operatorname{erfc}\left(\frac{r}{\sqrt{2}\gamma_{ij}}\right) \\ a'(r,\gamma_{ij}) &= \frac{\partial a(r,\gamma_{ij})}{\partial r}. \end{aligned}$$

Following the notation of [92], we can write the real-space error as

$$\Delta\boldsymbol{F}_{i,r}(\{q_j,\boldsymbol{r}_j\}_j) = \sum_j q_j f_i(\boldsymbol{r}_{ij},\gamma_{ij}),$$

where

$$f_i(\boldsymbol{r},\gamma) = \frac{q_i}{2}\sum_{\boldsymbol{n}}{}' \frac{\boldsymbol{r}+\boldsymbol{n}}{\|\boldsymbol{r}+\boldsymbol{n}\|^2} b(\|\boldsymbol{r}+\boldsymbol{n}\|,\gamma)\theta(\|\boldsymbol{r}+\boldsymbol{n}\|-r_{r,\mathrm{c}}),$$

with $\theta(x)$ being the Heaviside function. As in [52] by assuming independently $3d$-Gaussian distributed charge positions beyond the cutoff, we conclude

$$\begin{aligned} \langle(\Delta F_{i,r})^2\rangle &= \sum_j q_j^2 \langle f^2(\boldsymbol{r}_j,\gamma_{ij})\rangle \\ &= \sum_j q_j^2 \left\langle \frac{q_i^2}{4}\sum_{\boldsymbol{n}}{}' \frac{1}{\|\boldsymbol{r}_{ij}+\boldsymbol{n}\|^2} b^2(\|\boldsymbol{r}_{ij}+\boldsymbol{n}\|,\gamma_{ij})\theta(\|\boldsymbol{r}_{ij}+\boldsymbol{n}\|-r_{r,\mathrm{c}})\right\rangle. \end{aligned}$$

Note that the single lattice sum results from the approximation that all charge positions beyond the cutoff are independent. The expected value inside the sum is of the form $\langle f(\boldsymbol{r}_j)\rangle$. Since the set of all different states of $\boldsymbol{r}_j$ is precisely the unit cell Ω and a uniform distribution is assumed, the expected value is computed as

$$\langle f(\boldsymbol{r}_j)\rangle \approx \int_\Omega \mathrm{d}^3\boldsymbol{x}\ \frac{1}{V} f(\boldsymbol{x}).$$

Using this one obtains

$$\begin{aligned} &\left\langle \sum_{\boldsymbol{n}}{}' \frac{1}{\|\boldsymbol{r}_{ij}+\boldsymbol{n}\|^2} b^2(\|\boldsymbol{r}_{ij}+\boldsymbol{n}\|,\gamma_{ij})\theta(\|\boldsymbol{r}_{ij}+\boldsymbol{n}\|-r_{r,\mathrm{c}})\right\rangle \\ &\approx \frac{1}{V}\int_\Omega \mathrm{d}^3\boldsymbol{x}\ \sum_{\boldsymbol{n}} \frac{1}{\|\boldsymbol{x}+\boldsymbol{n}\|^2} b^2(\|\boldsymbol{x}+\boldsymbol{n}\|,\gamma_{ij})\theta(\|\boldsymbol{x}+\boldsymbol{n}\|) \\ &= \frac{4\pi}{V}\int_{r_{r,\mathrm{c}}}^{\infty} \mathrm{d}r\ b^2(r,\gamma_{ij}). \end{aligned}$$

One can then verify that the last expression can be bound from above by replacing $b(r, \gamma_{ij})$ with

$$\frac{\partial}{\partial r}\left(\frac{1}{r}\mathrm{erfc}\left(\frac{r}{\sqrt{2}\eta}\right)\right),$$

provided that $\eta \geq \max_{ij}\gamma_{ij}$. With this upper bound one arrives at the real-space error estimate for point charges as given in [92],

$$\sigma_{F,r} = 2q_{\max}^2\sqrt{\frac{N}{r_{r,\mathrm{c}}V}}e^{-\frac{r_{r,\mathrm{c}}^2}{2\eta^2}}. \tag{4.2}$$

By introducing $C \coloneqq 2^{\frac{3}{4}}q_{\max}^2\sqrt{\frac{N}{V}}$, $s_r \coloneqq \frac{r_{r,\mathrm{c}}}{\sqrt{2}\eta}$ and $s_k \coloneqq \frac{2\pi r_{k,\mathrm{c}}\eta}{\sqrt{2}}$, eqs. (4.1) and (4.2) can be written as

$$\sigma_{F,r} = C\frac{e^{-s_r^2}}{\sqrt{\eta s_r}} \quad \text{and} \quad \sigma_{F,k} = C\frac{e^{-s_k^2}}{\sqrt{\eta s_k}}.$$

Assuming independence of the real-space and Fourier space error [93] the total error can be estimated by

$$\sigma_F = \sqrt{\sigma_{F,r}^2 + \sigma_{F,k}^2}.$$

4.1.2 Choosing η and the Cutoffs

If we want the total error to be less than ε and in addition similar error contributions in both real- and Fourier space, there are now different options. For fixed ε, one of $\eta, r_{k,\mathrm{c}}, r_{r,\mathrm{c}}$ can be chosen freely. Practically, only two of them are interesting to tune. The real-space sum is a sum over all neighbors inside the cutoff sphere, which is usually implemented by iterating over a previously constructed neighbor list [73]. In many MD codes the minimum image convention (MIC) is used, which means that the cutoff sphere must be small enough to fit into the unit cell. Moreover, having only one real-space cutoff enables other performance improvements like having a single, minimal neighbor list for different operations. On the other hand, it was shown by [91] that under an optimal choice of η the Ewald summation scales with $\mathcal{O}(N^{\frac{3}{2}})$ instead of the general quadratic scaling with the number of charges N. Conceptionally speaking, this can be seen by first observing that the real-space sum scales quadratically compared to the linear scaling of the Fourier space sum. By tuning η one can shift the number of necessary summands between real- and Fourier space while keeping the error ε bounded. Thus, for truncated sums, one can use η to find a compromise between the quadratic scaling in the real- space and the linear scaling in Fourier space. However, this discussion does not apply when using the Ewald summation for the charge equilibration scheme discussed in section 3.2. Here, the Fourier space sum was also expressed in terms of the charge pairs q_i, q_j in order to formulate the total energy as a matrix equation, leading to quadratic scaling in Fourier space as well. In the following, we will discuss the two most interesting methods of choosing η and the cutoffs r_{rc}, $r_{k,\mathrm{c}}$.

Performance-Optimised η

The performance impact of the cutoffs r_{rc}, $r_{k,\mathrm{c}}$ for the Ewald summation can be estimated straightforwardly. This enables tuning η in such a way that the computational effort is minimized, as we are going to see. Since in this work we implemented the Ewald summation

into *n2p2* [6], which is in general not restricted to the MIC, the cutoffs can be chosen arbitrarily. This makes optimizing η appealing whenever *n2p2* is used as a standalone tool (e.g. training).

First, we start from the equation that arises if one requires the real- and Fourier space error estimates to be equal,

$$\frac{e^{-s^2}}{\sqrt{s}} = \sqrt{\frac{\eta}{2}}\frac{\varepsilon}{C}, \tag{4.3}$$

where $s > 0$, which can later be interpreted as s_r or s_k. Since the function defined on the left-hand side is continuous, strictly monotonically decreasing, approaches $+\infty$ as $s \to 0$ and 0 as $s \to \infty$, it is bijective on $s \in \mathbb{R}_{>0}$ and we can formally write $s = s(\eta)$. Now we need to estimate the total computation time of the Ewald sum. For that, we assume that the computation of a single summand is a constant τ_r or τ_k for a real- or Fourier space contribution, respectively. By further assuming a uniform distribution of charges inside the cutoff sphere, we can estimate the total computation time as[3]

$$\begin{aligned}\tau &\approx \frac{4\pi}{3} r_{r,\mathrm{c}}^3 \frac{N^2}{V}\tau_r + \frac{4\pi}{3} r_{k,\mathrm{c}}^3 V N^2 \tau_k \\ &= \frac{4\pi}{3}\left(\sqrt{2}\eta s\right)^3 \frac{N^2}{V}\tau_r + \frac{4\pi}{3}\left(\frac{\sqrt{2}s}{2\pi\eta}\right)^3 V N^2 \tau_k\end{aligned}$$

Next, minimization implies $\frac{\mathrm{d}\tau}{\mathrm{d}\eta} \stackrel{!}{=} 0$, where it is important to remember that $s = s(\eta)$. One finds the implicit relation

$$\eta^6 = \frac{\tau_k}{\tau_r}\frac{V^2}{(2\pi)^3}\left(1 + \frac{1}{2s(\eta)^2}\right) \tag{4.4}$$

In order to compute η we can use eq. (4.4) in combination with eq. (4.3). We start with an initial guess of η and afterwards use eq. (4.3) to determine s. Since the solution to s cannot be written in terms of elementary functions we use Newton's method to find s iteratively by

$$s_{n+1} = s_n + \left(\frac{2s_n}{4s_n^2 + 1}\right)\left(1 - \sqrt{s_n}e^{s_n^2}\sqrt{\frac{\eta}{2}}\frac{\varepsilon}{C}\right).$$

One has to take care that the starting value s_0 leads to a positive s_1. It is easy to verify that such a starting value always exists. Using the converged value for s in eq. (4.4) will in general lead to a new value of η. One then repeats the procedure until self-consistency is achieved. The convergence of this self-consistent cycle has not been shown analytically in this work. However, it was tested numerically for many different orders of magnitude of the initial guess of $\eta > 0$ without any problem. Additionally, for reasonable guesses one obtained a self-consistent η after only a few iterations. The ratio $\frac{\tau_k}{\tau_r}$ depends on the implementation and the underlying hardware. In *n2p2* we measured a value of $\frac{1}{3.676}$ on an Intel Core i7-7700HQ CPU at 2.80 GHz. We call this $\eta = \eta_{\mathrm{opt}}$. Note, however, that in favor of a valid error estimate, we take $\max_{ij}\{\eta_{\mathrm{opt}}, \gamma_{ij}\}$ at the end.

Finally, the cutoffs are computed by

$$r_{r,\mathrm{c}} = \sqrt{2}\eta s \quad r_{k,\mathrm{c}} = \frac{\sqrt{2}s}{2\pi\eta}.$$

[3] This is just a minor modification of the result in [73, 91].

Fixating the Real-Space Cutoff

For practical reasons, it is often preferred to set the real-space cutoff independently and adjust the Ewald parameter and the Fourier space cutoff accordingly. In this case we start again from eq. (4.2) and impose $\sigma_{F,r} \leq \frac{\varepsilon}{\sqrt{2}}$. Introducing $\tilde{C} = 2q_{\max}^2\sqrt{\frac{N}{V}}$, this can be written equivalently as

$$e^{\frac{-r_{r,\mathrm{c}}^2}{2\eta^2}} \leq \sqrt{\frac{r_{r,\mathrm{c}}}{2}}\frac{\varepsilon}{\tilde{C}}. \tag{4.5}$$

Our goal is to find the largest value of η such that this condition is still satisfied for given $r_{r,\mathrm{c}}$ and ε because a larger η allows a smaller Fourier space cutoff. Obviously, the left-hand side is in the interval $(0, 1]$, whereas the right-hand side can, in principle, be any non-negative number. If it is greater than one, any positive value of η satisfies the inequality. This means that the Fourier space cutoff can be made arbitrarily small by increasing η accordingly. However, in the derivation of the truncation error estimates in [92], several assumptions are made that will eventually be inaccurate if η is chosen too large. For example, it relies on $\frac{r_{r,\mathrm{c}}}{\sqrt{2}\eta}$ being large and that the sum $\sum_{\|\boldsymbol{k}\|>r_{k,\mathrm{c}}}$ in the Ewald summation can be approximated by an integral. In order to detect when the approximations become inaccurate, we analyzed the following approximation appearing in [92]

$$2\pi\int_{\frac{r_{r,\mathrm{c}}}{\sqrt{2}\eta}}^{\infty}\mathrm{d}t\; t^4 z^2(t) \approx \sqrt{\frac{\sqrt{2}\eta}{r_{r,\mathrm{c}}}}e^{-\frac{r_{r,\mathrm{c}}}{\sqrt{2}\eta}}.$$

Numerically we determined that a ratio of $\frac{r_{r,\mathrm{c}}}{\sqrt{2}\eta} \geq 1.63$ introduces an error of less than $10\,\%$. This bound is then used if eq. (4.5) does not constrain η. In most practical use cases, however, $r_{r,\mathrm{c}}$, ε, $q_{\max}$ and the particle density of the simulation cell lead to a value smaller than one on the right-hand side of eq. (4.5). In this case, we can determine the largest possible value of η by

$$\eta = \frac{r_{r,\mathrm{c}}}{\sqrt{-2\ln\left(\sqrt{\frac{r_{r,\mathrm{c}}}{2}}\frac{\varepsilon}{\tilde{C}}\right)}}.$$

Consequently, starting from eq. (4.1) and imposing a Fourier space truncation error of $\frac{\varepsilon}{\sqrt{2}}$, $r_{k,\mathrm{c}}$ is then determined by

$$\frac{1}{\sqrt{r_{k,\mathrm{c}}}}e^{-\frac{(2\pi r_{k,\mathrm{c}}\eta)^2}{2}} = \frac{\sqrt{\pi}\varepsilon\eta}{\tilde{C}}.$$

Analogously to the discussion of eq. (4.3), one can argue that $r_{k,\mathrm{c}}$ is determined uniquely by this condition. Again, using Newton's method, one can compute its value iteratively by

$$(r_{k,\mathrm{c}})_{n+1} = (r_{k,\mathrm{c}})_n + \frac{2(r_{k,\mathrm{c}})_n}{1+2(2\pi(r_{k,\mathrm{c}})_n\eta)^2}\left(1 - \sqrt{(r_{k,\mathrm{c}})_n}e^{\frac{(2\pi(r_{k,\mathrm{c}})_n\eta)^2}{2}}\frac{\sqrt{\pi}\varepsilon\eta}{\tilde{C}}\right).$$

4.2 Partial Screening of the Electrostatic Potential

Before we can compute the electrostatic energy we need to determine the charges via the charge equilibration method. To do so, we need to construct the matrix $\boldsymbol{A} = \boldsymbol{B} + \mathrm{diag}(\boldsymbol{J})$,

where $\boldsymbol{B}$ is the matrix appearing in the computation of the electrostatic energy $U = \frac{1}{2}\boldsymbol{q}^\intercal \boldsymbol{B}\boldsymbol{q}$ and $\mathrm{diag}(\boldsymbol{J})$ is a diagonal matrix with the hardness values along the diagonal. In the absence of PBCs, eq. (3.4) is used. Otherwise, the matrix $\boldsymbol{B}$ obtained via Ewald summation as given in eq. (3.18) is selected with a minor modification: in agreement with the work of Ko et al. [14] the "tinfoil" boundary condition $\varepsilon \to \infty$ is applied, see section 3.4.3. Thus, the term $\frac{4\pi}{3V}\boldsymbol{r}_i \cdot \boldsymbol{r}_j$ does not appear in the matrix $\boldsymbol{B}$.

The matrix $\boldsymbol{A}$ is then used for the charge equilibration by solving eq. (3.6) for $\boldsymbol{q}$. Afterwards, the electrostatic energy is computed. However, as we have discussed in section 2.4.3, for charges separated by a distance less than the symmetry function cutoff r_c, the interaction is screened by the screening function defined in eq. (2.28). So for periodic systems, U^{elec} from eq. (2.27) is computed by

$$\begin{aligned} U^{\mathrm{elec}} &= \frac{1}{2}\boldsymbol{q}^\intercal \boldsymbol{B}\boldsymbol{q} - \frac{1}{2}\sum_{\boldsymbol{n}}\sum_{i,j}\left[1 - f_{screen}(\|\boldsymbol{r}_{ij} + \boldsymbol{n}\|)\right] u_{ij}(\|\boldsymbol{r}_{ij} + \boldsymbol{n}\|) \\ &= \frac{1}{2}{\sum_{\boldsymbol{n}\in\mathbb{L}}}'\sum_{i,j} q_i q_j \frac{f_{\mathrm{screen}}(\|\boldsymbol{r}_{ij} + \boldsymbol{n}\|)\mathrm{erf}\left(\frac{\|\boldsymbol{r}_{ij}+\boldsymbol{n}\|}{\sqrt{2}\gamma_{ij}}\right) - \mathrm{erf}\left(\frac{\|\boldsymbol{r}_{ij}+\boldsymbol{n}\|}{\sqrt{2}\eta}\right)}{\|\boldsymbol{r}_{ij} + \boldsymbol{n}\|} \\ &\quad + \frac{1}{2\pi V}\sum_{\boldsymbol{k}\neq\boldsymbol{0}}\sum_{i,j} q_i q_j \frac{\exp\left(-2\pi^2\eta^2\|\boldsymbol{k}\|^2\right)}{\|\boldsymbol{k}\|^2}\cos\left(2\pi\boldsymbol{k}\cdot\boldsymbol{r}_{ij}\right) - \sum_{i=1}^{N}\frac{q_i^2}{\sqrt{2\pi}\eta}, \end{aligned}$$

where u_{ij} is the pair potential between Gaussian charges as given in eq. (3.1). Note that all summands with $\|\boldsymbol{r}_{ij} + \boldsymbol{n}\| > r_{\mathrm{c}}$ vanish. For non-periodic systems we obtain a simpler result

$$U^{\mathrm{elec}} = \frac{1}{2}\boldsymbol{q}^\intercal \boldsymbol{B}\boldsymbol{q} - \frac{1}{2}\sum_{i,j}\left[1 - f_{screen}(r_{ij})\right] u_{ij}(r_{ij}) = \frac{1}{2}\sum_{i\neq j} f_{\mathrm{screen}}(r_{ij}) u_{ij}(r_{ij}).$$

4.3 Derivatives

In section 2.4 we discussed what a NNP is. For a given configuration $\boldsymbol{r}^{3N}$ it outputs the potential energy $\tilde{U}(\boldsymbol{\Theta}; \boldsymbol{r}^{3N})$ of the corresponding system. Since this is a $\mathcal{C}^1(\mathbb{R}^{3N})$ function, provided differentiable activation and symmetry functions are used, one can compute the force $\tilde{\boldsymbol{F}}_\alpha = -\boldsymbol{\nabla}_{\boldsymbol{r}_\alpha}\tilde{U}(\boldsymbol{\Theta}; \boldsymbol{r}^{3N})$. In contrast to previous chapters, we will now use the convention that Greek indices refer to atoms in order to distinguish them from Latin letters which refer to vector or matrix components. Thus, the previous expression can be written more explicitly as

$$\tilde{F}_\alpha^i = -\frac{\partial\tilde{U}(\boldsymbol{\Theta}; \boldsymbol{r}^{3N})}{\partial r_\alpha^i}.$$

This relation makes it possible to use a NNP in MD. Furthermore, as mentioned in section 2.4.1 the forces are analytic and are differentiable with respect to the weights and biases θ, so they can also be used for training the NNP.

In the following, we will give a short overview of the derivatives that are needed for force calculations and the training procedure. Derivatives that are needed but had already been implemented in *n2p2* at the start of this project are the following: The derivatives of symmetry functions with respect to atom positions[4], $\frac{\partial G_i}{\partial r_\alpha^j}$, the derivatives of the NN output,

[4] We refer the reader to [6].

in this case the short-range energy U^{short} or the electronegativities χ_i, with respect to the weights and biases $\boldsymbol{\Theta}$[5]. Finally, the derivatives of the NN output with respect to the input vector components, which does not pose any difficulties due to the simple structure of a NN, see eq. (2.1).

Since the quantities for forces, energies, electronegativities and charges always refer to the quantities determined by the NNs in this section, there is no need to distinguish them from reference values by putting a "$\sim$" on top of the symbols in comparison to the notation in chapter 2.

4.3.1 Derivatives for Force Prediction

In this subsection, we are not interested in the derivative with respect to the weights $\boldsymbol{\Theta}$ or the hardness $\boldsymbol{J}$ which is independent of $\boldsymbol{r}^{3N}$, so we will suppress them in the notation and just write $U(\boldsymbol{r}^{3N})$. Using eqs. (2.27) and (2.29) we can write more explicitly

$$F_\alpha^i = -\frac{\mathrm{d}}{\mathrm{d}r_\alpha^i} U^{\text{elec}}\left(\boldsymbol{r}^{3N}, \boldsymbol{q}\left(\boldsymbol{\chi}\left(\boldsymbol{r}^{3N}\right), \boldsymbol{r}^{3N}\right)\right) - \frac{\mathrm{d}}{\mathrm{d}r_\alpha^i} \sum_{\beta=1}^{N} U_\beta^{\text{short}}\left(\boldsymbol{r}^{3N}, q_\beta\left(\boldsymbol{\chi}\left(\boldsymbol{r}^{3N}\right), \boldsymbol{r}^{3N}\right)\right).$$

In order to apply the chain rule correctly one has to be careful with the notation. For a function $f(\boldsymbol{r}^{3N}, g(\boldsymbol{r}^{3N}))$ that is composed with another function g we write the total derivative as[6]

$$\frac{\mathrm{d}}{\mathrm{d}r_\alpha^i} f(\boldsymbol{r}^{3N}, g(\boldsymbol{r}^{3N})) = \lim_{h\to 0} \frac{1}{h} \left| f(\boldsymbol{r}^{3N} + h\boldsymbol{e}_{3\alpha+i}, g(\boldsymbol{r}^{3N} + h\boldsymbol{e}_{3\alpha+i})) - f(\boldsymbol{r}^{3N}, g(\boldsymbol{r}^{3N})) \right|,$$

where $\boldsymbol{e}_j \in \mathbb{R}^{3N}$ is a Cartesian basis vector with entry 1 at position j. In comparison, we interpret the partial derivative of $f(\boldsymbol{r}^{3N}, g(\boldsymbol{r}^{3N}))$ as

$$\frac{\partial}{\partial r_\alpha^i} f(\boldsymbol{r}^{3N}, g(\boldsymbol{r}^{3N})) = \lim_{h\to 0} \frac{1}{h} \left| f(\boldsymbol{r}^{3N} + h\boldsymbol{e}_{3\alpha+i}, g(\boldsymbol{r}^{3N})) - f(\boldsymbol{r}^{3N}, g(\boldsymbol{r}^{3N})) \right|$$

With this, we rewrite the force as

$$F_\alpha^i = -\frac{\partial U^{\text{elec}}}{\partial r_\alpha^i} - \sum_{\beta=1}^{N} \frac{\partial U^{\text{elec}}}{\partial q_\beta} \frac{\mathrm{d}q_\beta}{\mathrm{d}r_\alpha^i} - \sum_{\beta=1}^{N} \left(\sum_{j=1}^{d} \frac{\partial U_\beta^{\text{short}}}{\partial G_\beta^j} \frac{\partial G_\beta^j}{\partial r_\alpha^i} + \frac{\partial U_\beta^{\text{short}}}{\partial q_\beta} \frac{\mathrm{d}q_\beta}{\mathrm{d}r_\alpha^i} \right). \tag{4.6}$$

For the non-periodic case, we obtain the following relation

$$\frac{\partial U^{\text{elec}}}{\partial r_\alpha^i} = \sum_{\delta\neq\alpha} \frac{r_\alpha^i - r_\gamma^i}{r_{\alpha\delta}} \left[f'_{\text{screen}}(r_{\alpha\delta}) u_{\alpha\delta}(r_{\alpha\delta}) + f_{\text{screen}}(r_{\alpha\delta}) u'_{\alpha\delta}(r_{\alpha\delta}) \right],$$

[5] Those can simply be computed via the backpropagation algorithm [32].

[6] Mathematicians sometimes speak about a derivative with respect to a slot and the derivative with respect to a variable. They often use a different notation since the one we are using here is still ambiguous in some cases.

where

$$f'_{\text{screen}}(r) = \begin{cases} 0 & \text{if } r < r_{\text{i,c}} \\ \frac{\pi}{2(r_{\text{c}}-r_{\text{i,c}})} \sin\left(\pi \frac{r-r_{\text{i,c}}}{r_{\text{c}}-r_{\text{i,c}}}\right) & \text{if } r_{\text{i,c}} < r < r_{\text{c}} \\ 0 & \text{else} \end{cases},$$

$$u'_{\alpha\beta}(r) = q_\alpha q_\beta v(r, \gamma_{\alpha\beta}) \quad \text{and}$$

$$v(r,\gamma) = \frac{1}{r}\left(\sqrt{\frac{2}{\pi\gamma^2}} e^{-\frac{r^2}{2\gamma^2}} - \frac{1}{r}\text{erf}\left(\frac{r}{\sqrt{2}\gamma}\right)\right).$$

In the periodic case, we compute

$$\begin{aligned}\frac{\partial U^{\text{elec}}}{\partial r^i_\alpha} = \frac{1}{2}\sum_{\boldsymbol{n}}{}' \sum_{\beta\neq\alpha} \frac{r^i_\alpha - r^i_\beta}{r_{\alpha\beta}} \big[& f'_{\text{screen}}(\|\boldsymbol{r}_{\alpha\beta}+\boldsymbol{n}\|) u_{\alpha\beta}(\|\boldsymbol{r}_{\alpha\beta}+\boldsymbol{n}\|) \\ & + f_{\text{screen}}(\|\boldsymbol{r}_{\alpha\beta}+\boldsymbol{n}\|) u'_{\alpha\beta}(\|\boldsymbol{r}_{\alpha\beta}+\boldsymbol{n}\|) - q_\alpha q_\beta v(\|\boldsymbol{r}_{\alpha\beta}+\boldsymbol{n}\|,\eta)\big] \\ & - \frac{2}{V}\sum_{\boldsymbol{k}\neq\boldsymbol{0}}\sum_{\beta\neq\alpha} q_\alpha q_\beta \frac{\exp\left(-2\pi^2\eta^2\|\boldsymbol{k}\|^2\right)}{\|\boldsymbol{k}\|^2} k^i \sin\left(2\pi\boldsymbol{k}\cdot\boldsymbol{r}_{\alpha\beta}\right).\end{aligned}$$

The term $\frac{\partial U^{\text{elec}}}{\partial q_\alpha}$ is straightforward to compute for both the periodic and non-periodic case. For non-periodic systems one obtains

$$\frac{\partial U^{\text{elec}}}{\partial q_\alpha} = \sum_{\beta\neq\alpha} f_{\text{screen}}(r_{\alpha\beta}) \frac{u_{\alpha\beta}(r_{\alpha\beta})}{q_\alpha},$$

whereas under PBCs the result is

$$\begin{aligned}\frac{\partial U^{\text{elec}}}{\partial q_\alpha} = & \sum_{\boldsymbol{n}\in\mathbb{L}}{}' \sum_\beta q_\beta \frac{f_{\text{screen}}(\|\boldsymbol{r}_{\alpha\beta}+\boldsymbol{n}\|)\text{erf}\left(\frac{\|\boldsymbol{r}_{\alpha\beta}+\boldsymbol{n}\|}{\sqrt{2}\gamma_{\alpha\beta}}\right) - \text{erf}\left(\frac{\|\boldsymbol{r}_{\alpha\beta}+\boldsymbol{n}\|}{\sqrt{2}\eta}\right)}{\|\boldsymbol{r}_{\alpha\beta}+\boldsymbol{n}\|} \\ & + \frac{1}{\pi V}\sum_{\boldsymbol{k}\neq\boldsymbol{0}}\sum_\beta q_\beta \frac{\exp\left(-2\pi^2\eta^2\|\boldsymbol{k}\|^2\right)}{\|\boldsymbol{k}\|^2}\cos\left(2\pi\boldsymbol{k}\cdot\boldsymbol{r}_{\alpha\beta}\right) - \sqrt{\frac{2}{\pi}}\frac{q_\alpha}{\eta}.\end{aligned}$$

Here, the notation $\sum'_{\boldsymbol{n}}\sum_\beta$ means that if $\boldsymbol{n}=\boldsymbol{0}$ then the term $\beta=\alpha$ is not included.

The term

$$\frac{\partial U^{\text{short}}_\beta}{\partial G^j_\beta}\frac{\partial G^j_\beta}{\partial r^i_\alpha}$$

is the product of the derivative of the NN output with respect to its input, i.e. the symmetry function and the derivative of the corresponding symmetry function. Since this was already necessary for the 2G implementation in *n2p2*, we refer the reader to [6]. Similarly, since q_β is also an input of the NN computing U^{short}_β, $\frac{\partial U^{\text{short}}_\beta}{\partial q_\beta}$ is formally a derivative of the NN output with respect to its input as well.

What is left for the force computation is the derivative $\frac{\mathrm{d}q_\beta}{\mathrm{d}r^i_\alpha}$. This is unfortunately costly to compute, since $\boldsymbol{q}$ is determined by solving the linear system of equations in eq. (3.6) and thus also its derivative needs to be computed by solving

$$\bar{\boldsymbol{A}}\frac{\mathrm{d}\bar{\boldsymbol{q}}}{\mathrm{d}r^i_\alpha} = -\frac{\mathrm{d}\bar{\boldsymbol{\chi}}}{\mathrm{d}r^i_\alpha} - \frac{\partial\bar{\boldsymbol{A}}}{\partial r^i_\alpha}\bar{\boldsymbol{q}}. \tag{4.7}$$

The elements of the vector $\frac{\mathrm{d}\bar{\boldsymbol{\chi}}}{\mathrm{d}r_\alpha^i}$ are computed by

$$\sum_j \frac{\partial \chi_\beta}{\partial G_\beta^j} \frac{\partial G_\beta^j}{\partial r_\alpha^i},$$

apart from the last element which is 0. The matrix $\frac{\partial \bar{\boldsymbol{A}}}{\partial r_\alpha^i}$ is sparse [14], up to the last row and column it equals the matrix $\frac{\partial \boldsymbol{B}}{\partial r_\alpha^i}$, which has non-zero entries only along row and column α

$$\left(\frac{\partial \boldsymbol{B}}{\partial r_\alpha^i}\right)_{\beta\delta} = \begin{cases} \frac{r_\alpha^i - r_\delta^i}{r_{\alpha\delta}} v(r_{\alpha\delta}, \gamma_{\alpha\delta}) & \text{if } \alpha = \beta \neq \delta \\ [\delta \leftrightarrow \beta] & \text{if } \beta \neq \delta = \alpha \\ 0 & \text{else} \end{cases}.$$

In the second line on the right-hand side, we have indicated that it is the same expression as in the line above but the index δ is exchanged with β.

Looking at eq. (4.7), it is clear that for a single force component computation one needs to solve a linear system of equations of the size $N+1$. In practical applications we need all components of all N forces, which would require solving $3N$ systems of equations. Fortunately, Ko et al. [14] used a different method for the force computation which reduces the computational effort a lot. The method was already used in a similar application in [95] and is related to the work of Handy and Schaefer [96]. It turns out that the force can be expressed as

$$F_\alpha^i = \frac{\partial U^{\mathrm{elec}}}{\partial r_\alpha^i} + \sum_\beta \sum_j \frac{\partial U_\beta^{\mathrm{short}}}{\partial G_\beta^j} \frac{\partial G_\beta^j}{\partial r_\alpha^i} + \boldsymbol{\mu} \cdot \left(\frac{\partial \bar{\boldsymbol{A}}}{\partial r_\alpha^i} \bar{\boldsymbol{q}} + \frac{\partial \bar{\boldsymbol{\chi}}}{\partial r_\alpha^i} \right), \tag{4.8}$$

where $\boldsymbol{\mu} \in \mathbb{R}^{N+1}$ is a vector of Lagrange multipliers that are determined by solving

$$\left(\bar{\boldsymbol{A}}\boldsymbol{\mu}\right)_i = -\frac{\partial U^{\mathrm{elec}}}{\partial \bar{q}_i} - \sum_\beta \frac{\partial U_\beta^{\mathrm{short}}}{\partial \bar{q}_i} \quad \forall i \in \{1, \ldots, N+1\}.$$

Since $\bar{q}_{N+1} = \lambda$ the right hand side of this equation is 0 for $i = N+1$. Thus eq. (4.8) can be computed by solving only one linear system of equations.

4.3.2 Additional Derivatives for Neural Network Training

During the training of the NNs further derivatives are needed. In particular, during the training of the charge NNs, one needs to compute $\frac{\partial \boldsymbol{q}}{\partial \chi_\alpha}$ and $\frac{\partial \boldsymbol{q}}{\partial J_{[\alpha]}}$. The corresponding relations are obtained straightforwardly from eq. (3.6). Thus, one needs to solve

$$\begin{aligned} \boldsymbol{A} \frac{\partial \boldsymbol{q}}{\partial \chi_\alpha} &= -\boldsymbol{e}_\alpha \\ \left(\boldsymbol{A} \frac{\partial \boldsymbol{q}}{\partial J_{[\alpha]}}\right)_\beta &= -\delta_{[\alpha][\beta]} q_\beta \quad \forall \beta \in \{1, \ldots, N\}, \end{aligned}$$

where $\boldsymbol{e}_\alpha$ is the Cartesian basis vector with entry 1 at index α.

Additionally, the derivatives with respect to the weights and biases $\boldsymbol{\theta}_\chi$ are necessary for training. We have now added the subscript χ to indicate that we only refer to the weights

of the NNs that are used for computing electronegativities. Using the backpropagation algorithm,

$$\frac{\partial \chi_\alpha}{\partial \theta^i_{\chi,[\alpha]}} \quad \text{and} \quad \frac{\partial U^{\text{short}}_\alpha}{\partial \theta^i_{\chi,[\alpha]}}$$

can be computed efficiently. This method had already been implemented in *n2p2*. The charge derivatives with respect to the weights and biases can then be computed by

$$\frac{\mathrm{d}\boldsymbol{q}}{\mathrm{d}\theta^i_{\chi,[\alpha]}} = \sum_\beta \frac{\partial \boldsymbol{q}}{\partial \chi_\beta} \frac{\partial \chi_\beta}{\partial \theta^i_{\chi,[\alpha]}}.$$

For the corresponding derivative of the forces, eq. (4.8) would be a possible starting point to derive an efficient formula. However, this does not work out, since it results in computing $\frac{\mathrm{d}\boldsymbol{\mu}}{\mathrm{d}\theta^i_{U,[\alpha]}}$, which is determined by solving again a system of equation of the size $(N+1)$. For a single update, this would require solving N_θ (see eq. (2.18)) different systems of equations. Furthermore, $\frac{\mathrm{d}\boldsymbol{\mu}}{\mathrm{d}\theta^i_{U,[\alpha]}}$ changes after each update. In comparison, when starting from eq. (4.6) one computes

$$\frac{\mathrm{d}F^i_\alpha}{\mathrm{d}\theta^j_{U,[\gamma]}} = -\sum_{\beta=1}^{N} \left(\sum_{j=1}^{d} \frac{\partial^2 U^{\text{short}}_\beta}{\partial G^j_\beta \partial \theta^j_{U,[\gamma]}} \frac{\partial G^j_\beta}{\partial r^i_\alpha} + \frac{\partial^2 U^{\text{short}}_\beta}{\partial q_\beta \partial \theta^j_{U,[\gamma]}} \frac{\mathrm{d}q_\beta}{\mathrm{d}r^i_\alpha} \right). \tag{4.9}$$

The forces are only used for optimizing the second type of NNs, so we only need the derivative with respect to the weights $\theta^j_{U,[\gamma]}$. It is easy to see that

$$\frac{\partial^2 U^{\text{short}}_\beta}{\partial G^j_\beta \partial \theta^j_{U,[\gamma]}} \quad \text{and} \quad \frac{\partial^2 U^{\text{short}}_\beta}{\partial q_\beta \partial \theta^j_{U,[\gamma]}}$$

are again straightforward to compute and corresponding methods had already existed in *n2p2* at the beginning of this project. Equation (4.9) still requires to solve $\frac{\mathrm{d}\boldsymbol{q}}{\mathrm{d}r^i_\alpha}$. In *n2p2* only the information of a single force component or potential energy is used for each update. Thus, $\frac{\mathrm{d}\boldsymbol{\mu}}{\mathrm{d}\theta^i_{U,[\alpha]}}$ needs to be computed for all $\theta^i_{U,[\alpha]}$ again anyway after each update. This means that eq. (4.9) is unequivocally the better algorithm. Usually one only uses a small fraction of all available forces as discussed in section 2.4.1, so the number of equation systems that need to be solved reduces noticeably.

4.4 Training Procedure

In section 2.4.3 we have already given an overview of the training of 4G NNPs. It is divided into two stages, the charge and the short-range energy training. Since the charges have to be trained first, we call this part sometimes *stage 1* and the latter one *stage 2*. It is common to quantify the length of a particular training in terms of *epochs*. Usually, an epoch consists of as many *training updates* as there are data points in the training data set. A training update is the process of making a single error minimization step by simultaneously changing all weights and parameters. In *n2p2* this definition is slightly modified: it only counts the number of force training points that are actually used and if the multistream EKF is used with n streams, then each update effectively uses n training data points. Consequently, a

more meaningful definition is to define an epoch as the number of selected training data points divided by the number of streams n [16]. For energies and forces, we stick with the implemented method which chooses a single training data point for each update. For the charge update, this turned out unfavorably, so we deviated from this approach as will be explained in the next subsection.

4.4.1 Charge Training

In principle, the training data consists of a vector of charges $\boldsymbol{q}$ for each structure $\boldsymbol{r}^{3N}$, which provides N training data points per structure. Following the notation of section 2.2.2, the error vector could be defined as the scalar $\xi = q_\alpha - \tilde{q}_\alpha$. Assuming that the EKF is a reasonable approximation, it would then minimize $\mathbb{E}[(q_\alpha - \tilde{q}_\alpha)^2]$ in the following update with respect to the weights, biases and hardness of all elements. Although it is not part of the NN but only appears in the charge equilibration step, the optimization of the charge error with respect to this quantity works analogously to the weights and biases $\boldsymbol{\Theta}$. Minimizing $\mathbb{E}[(q_\alpha - \tilde{q}_\alpha)^2]$ would mean the NN is trained in a way that optimizes individual charge prediction. Although this is favorable, the computational costs are comparatively large. One update step requires solving the equation system in eq. (3.6). Despite needing only a single charge of the structure, we obtain all charges by solving the equation. After updating $\boldsymbol{\Theta}$ and $\boldsymbol{J}$, eq. (3.6) changes and has to be solved again.

This led to the decision of changing the definition of the error to

$$\xi = \sqrt{\sum_{\alpha=1}^{N} \left(q_\alpha - \tilde{q}_\alpha(\boldsymbol{\Theta}, \boldsymbol{J})\right)^2}. \tag{4.10}$$

This is certainly a major modification since previously ξ was assumed to be at least approximately Gaussian distributed with zero mean. However, the new definition only allows for non-negative values and is thus far from being symmetric around zero. The only justification that we can give at the moment are numerical results exhibiting a satisfying decrease of the empirical error

$$\frac{1}{N^{\mathrm{st}}} \sum_{i=1}^{N^{\mathrm{st}}} \frac{1}{N_i} \sum_{\alpha}^{N_i} (q_{\alpha,i} - \tilde{q}_{\alpha,i})^2, \tag{4.11}$$

where N^{st} is the number of structures that are contained in the training data set, N_i is the number of atoms in structure i and $q_{\alpha,i}, \tilde{q}_{\alpha,i}$ are the charges of atom α in structure i. During training, we need to ensure that the $J_{[\alpha]}$ are non-negative, otherwise the charge equilibration may not be well-defined, see section 3.2. Analogously as done in [14], we express $J_{[\alpha]} = j_{[\alpha]}^2$ and optimize j during training. The question arises how $\boldsymbol{H}$ is defined when using the error in eq. (4.10). It is clear that $\boldsymbol{H}$ needs to contain the derivatives with respect to $\boldsymbol{\Theta}$ and $\boldsymbol{j}$. For notational convenience, we combine them to a single vector

$$\boldsymbol{\Lambda} = (\boldsymbol{\theta}_1^\intercal, \ldots, \boldsymbol{\theta}_{N_{\mathrm{el}}}^\intercal, \boldsymbol{j}^\intercal)^\intercal.$$

From the definitions for ξ and $\boldsymbol{H}$ in the EKF one obtains the relation

$$\boldsymbol{H}_i = -\left.\frac{\partial \xi(\boldsymbol{\Lambda})}{\partial \Lambda_i}\right|_{\boldsymbol{\Lambda}=\hat{\boldsymbol{\Lambda}}} = -\frac{1}{\xi(\boldsymbol{\Lambda})} \sum_\beta [q_\beta - \tilde{q}_\beta(\boldsymbol{\Lambda})] \left.\frac{\partial \tilde{q}_\beta(\boldsymbol{\Lambda})}{\partial \Lambda_i}\right|_{\boldsymbol{\Lambda}=\tilde{\boldsymbol{\Lambda}}},$$

where $\hat{\mathbf{\Lambda}}$ is the best current guess of $\mathbf{\Lambda}$. The derivatives on the right-hand side have all been discussed in section 4.3.2.

In summary, a single charge update takes the training data of all charges belonging to a single structure, which still only needs one charge equilibration and thus improves the computational performance drastically. It was also taken care of the available selection modes for the training data in the sense that they are also applicable for the charge training. Those include *random*, *sorted* and *threshold* selection. Random selection draws a random training data point that has not been used yet in the current epoch. Sorted selection means that the training data is sorted at the beginning of each epoch such that data points for which the prediction error is high are used first for updates. When using threshold selection, only data points that cause prediction errors higher than a specified threshold are considered. Among these the order is random.

4.4.2 Energy and Force Training

Energy and force training had already been part of *n2p2* before the beginning of this project. Only a single energy value or force component is selected for each training update. The error vectors are

$$\xi = U - \tilde{U} \quad \text{and} \quad \xi = \beta(F_\alpha^i - \tilde{F}_\alpha^i).$$

The corresponding matrices $\boldsymbol{H}$ are straightforward to compute by using eq. (4.9) and the backpropagation algorithm. The parameter β has been discussed in section 2.4.1. Previously, forces have been selected independently in the sense that they did not belong to the same structure in general. From eq. (4.8) it is clear that this is an expensive decision in 4G, since for each force belonging to a different structure, the charges have to be computed again. This is why we have chosen to make small batches of forces belonging to the same structure. The size of those batches is roughly the number of forces available in the structure, i.e. $3N$, times the percentage of forces that are used. However, we still allow energy updates to interrupt the consecutive updates of forces of the same batch. A typical updating schedule can thus look like this

$$F \to F \to U \to F \to F \to U \overset{\text{new batch}}{\to} F \to F \ldots.$$

The only disadvantage of this decision is that forces of the same batch may be correlated more than the ones from different structures and make the selection not completely random. Because of the recency effect of the EKF this could lead to an overemphasis in the updates on the current structure. Nevertheless, forces from the following structures should compensate for that. Similar to the charge training it was taken care of supporting all selection modes. Random selection of forces, however, means now that the structures are selected randomly and afterwards the forces in the small batches are randomized.

4.4.3 Memory and Parallelization Considerations

One may argue that during training the expensive computation of quantities like $\boldsymbol{A}$, $\frac{\partial \bar{\boldsymbol{A}}}{\partial r_\alpha^i}$ and $\frac{\mathrm{d}\boldsymbol{q}}{\mathrm{d}r_\alpha^i}$ has to be done only once for each structure in the training data set and is therefore negligible when training over hundreds of epochs. Although this is a justified objection, we point out that, depending on the size of the training data set, storing all these values may become a memory issue. The largest array is $\frac{\partial \bar{\boldsymbol{A}}}{\partial r_\alpha^i}$, if stored for each pair (α, i). This can be improved by observing that we only ever need the product $\frac{\partial \bar{\boldsymbol{A}}}{\partial r_\alpha^i}\boldsymbol{q}$ [14]. This reduces the

memory footprint by a factor of N. A simple estimate of the memory needed for storing all of these quantities is

$$8\left[N^2 + 2(3N_sN^2)\right] \text{ bytes.}$$

Here we have assumed that entries are saved as 64-bit floats. A realistic training data set may easily contain 5000 structures with 500 atoms each. This would lead to a memory footprint of around 60 GB. Since there are also other large arrays to store, this definitely rules out most personal computers and may even become a problem for decent scientific computers. This is why we are only storing $\boldsymbol{A}$ at the moment but free the memory for the other quantities after each change of structure.

Another object that has a large memory footprint in *n2p2* are neighbor lists. To avoid redundancy we opted for one neighborlist per atom which contains neighbors up to the cutoff distance $\max\{r_{r,\mathrm{c}}, r_\mathrm{c}\}$, where $r_{r,\mathrm{c}}$ and r_c are the real-space cutoffs of the symmetry functions and the Ewald summation, respectively. For performance reasons, these lists are sorted by distance. Note that for neighbors that are beyond r_c but closer than $r_{r,\mathrm{c}}$, large arrays like $\frac{\partial \boldsymbol{G}_{[\alpha]}}{\partial r^i_\beta}$ are not needed.

Parallelization is an integral part of *n2p2*. Especially the training routine can be run with multiple tasks either on distributed- or shared-memory processors by following the MPI standard. As already explained in section 2.2.2, the multistream EKF can be parallelized in most parts. Each update is synchronized, but the individual computations for different training data points happen independently in different MPI tasks. For stage 1 no special treatment was needed for compatibility with this parallelization method. Stage 2 of the training is straightforward in most parts too. The only exception is that one should make sure that the small batches of force updates discussed in section 4.4.2 are aligned. This means that for each task batches are of the same size as in other tasks and new batches start at the same update. Otherwise computational time would be lost while other tasks wait for those tasks which have to compute large arrays (as the one discussed in this subsection) for new structures.

When deploying the NNP in the prediction mode, parallelization is also available. However, this part of *n2p2* can only be run on multiple threads on shared-memory processors. This is implemented with the *OpenMP* API. In this work, we followed this approach and further parallelized the sorting of all neighborlists, the computation of the matrix $\boldsymbol{A}$ needed for charge equilibration and the corrections due to screening short-ranged electrostatics. Additionally, computing $\frac{\partial \boldsymbol{A}}{\partial r^i_\alpha}\boldsymbol{q}$ can also be multithreaded. Force computation had already been parallelized, but since this procedure was extended noticeably during this project, modifications had to be made to continue the support of multithreading.

The new implementations in *n2p2* have been analyzed in serial execution with the programming tools provided by *valgrind*. On Intel CPUs parallel efficiency of the OpenMP multithreading was investigated with the *Intel VTune* profiler. Both tools enabled considerable improvement of the code's efficiency.

4.5 Prototyping Interface to *LAMMPS*

Altough 4G NNPs can be used on its own for some studies [14], the use cases are limited. This changes when they are used as force fields for MD. As we have already mentioned, NNPs provide the advantage that they feature an accuracy comparable with ab initio methods but only for a fraction of the computational costs. This makes them appealing for MD because

they enable the simulation of longer time intervals. The previous version of *n2p2* made it possible to conduct MD simulations with a 2G NNP by providing an interface to the popular MD simulation package *LAMMPS* [18].

LAMMPS is a complex program with over a million lines of code. Its core methods are optimized thoroughly. In order to implement an interface between *n2p2* and *LAMMPS*, one has to understand how the computation of forces and time integration is parallelized in *LAMMPS*. The simulation cell is partitioned into smaller cells, where each cell is assigned to an individual MPI task [17, 18]. In general, those cells can be triclinic and of different sizes to allow for load-balancing. Each task only stores atoms belonging to the corresponding cell (*owned atoms*) as well as so-called *ghost atoms*. Those are atoms that are part of neighboring cells but have distances smaller than the cutoff of the force field with respect to the shared boundary. Consequently, each task has its own neighbor lists containing only owned and ghost atoms. Reassigning atoms to different tasks when they have crossed boundaries and rebuilding the neighbor lists happens after a certain amount of time steps. In between those expensive synchronization steps, communication is minimized by only exchanging information of ghost atoms. This approach to parallelization has been combined seaminglessly with 2G NNPs in the sense that each task calls the corresponding methods of *n2p2* independently [6]. For a correct force computation, *n2p2* only needs the information of the atom itself and its neighbors inside the cutoff sphere.

Although 4G with its global charge equilibration step offers interesting new capabilities, it also introduces an inevitable disadvantage regarding parallelization. The forces, which are needed in each time step, depend on the charges and thus a synchronization of all atoms in the simulation cell is always necessary. Apart from this downside, an interface making use of 4G NNPs is nevertheless possible. Under these circumstances, the most efficient implementation would involve two communication steps per time step via the interface. The first one exchanges atom positions for electronegativities. Those can then be used to conduct the global charge equilibration inside *LAMMPS*. Afterwards, the charges are communicated to *n2p2* again in order to receive the resulting forces. Since this implementation is quite involved owing to the high complexity of *LAMMPS*, this approach was decided to be outside the scope of this work[7]. As a consequence, we settled with a simpler but also less efficient solution. In this project, we limited *LAMMPS* to a single task and let it exchange all atoms in the simulation cell with a single instance of *n2p2*. Charge equilibration as well as force calculation is then done consecutively inside *n2p2* and afterwards, the forces are handed back to *LAMMPS*. In order to still make use of multiple processors to some extent, OpenMP parallelization inside *n2p2* is utilized to a great extent as explained in section 4.4.3. Also, the communication step between the two programs involving the transfer of atom positions and forces can be multithreaded with OpenMP. For a system consisting of 384 atoms, this approach yielded better performance than initially anticipated. An advantage of this interface design is the minimal coupling between both programs. This allows quick prototyping of any modifications to the NNP and is therefore interesting during development.

[7]However, there is ongoing work on this solution in cooperation with the Behler group.

Chapter 5

Test Case Liquid Water

In order to test the implementation of a 4G NNP in *n2p2*, we decided to test it by simulating liquid water by means of MD. This decision was motivated by the fact that in the Dellago group intensive investigations about this system had been conducted before. Among other tools this included the 2G NNP implementation *n2p2* [6]. On top of that, liquid water is in general an interesting benchmarking system because of its anomalies which are non-trivial to simulate. Additionally, there is a tremendous amount of experimental data, which can be used for comparison.

5.1 Neural Network Potential Setup

One of the most important decisions when constructing a Behler-Parrinello NNP is the choice of symmetry functions. Since there are two different kinds of NNs in the 4G approach, one could in principle use different sets of symmetry functions for the charge and the energy prediction, respectively. However, there is little motivation in doing so since the requirements are the same: they should provide detailed information about the local neighborhood. Since liquid water has already been successfully analyzed with the 2G version of *n2p2* [6] and *RuNNer* [5], we decided to use the same set of symmetry functions as were used in those works. Remember that symmetry functions of atoms of the same element only differ by their chosen reference system, which is typically chosen to be the atom's position as we have explained in section 2.4.2. Consequently, we only need to specify the symmetry functions of different elements. In this work, we have used 16 radial symmetry functions of the type given in eq. (2.22) for both hydrogen and oxygen, as well as 11 angular symmetry functions (see eq. (2.23)) for hydrogen and 14 for oxygen. Those were combined with the cutoff function from eq. (2.25) with a cutoff of $r_\mathrm{c} = 6.35\,\text{Å}$. The parameter values can be found in table D.1. In general, if a suitable set of symmetry functions has to be constructed from scratch, [97] provides a systematic approach.

The architecture of both the charge and the energy NNs consisted of two hidden layers with $N_1 = N_2 = 25$ nodes per layer. The activation functions for both hidden layers were $\sigma = \tanh(x)$ whereas no activation function was applied on the output node[1].

[1] In the definition of a NN in eq. (2.1) an activation function at the output node does not even appear.

5.2 Training Details

For both training stages, we rescaled the symmetry functions. That means that each symmetry function G_α^i was rescaled according to

$$G_\alpha'^i = \frac{G_\alpha^i - \bar{G}_{[\alpha]}^i}{G_{[\alpha],\max}^i - G_{[\alpha],\min}^i},$$

where the average $\bar{G}_{[\alpha]}^i$, maxima $G_{[\alpha],\max}^i$ and minima $G_{[\alpha],\min}^i$ of G_α^i are taken over all symmetry function values in the complete data set corresponding to the same element $[\alpha]$ and index i.

In *n2p2* it is also possible to normalize energies and forces, which means that internally custom units are used such that the energy has zero mean and both energy and force standard deviations are one [16]. However, in this work this feature was not used and a corresponding normalization of the charges has not been implemented yet[2].

The weights for all NN have been initialized by drawing them from a uniform distribution as discussed in section 2.2.2.

5.2.1 Data Set

Regarding the training data that is necessary for the training, we have used the one that was initially prepared in [5]. The data set itself is provided at [98]. Actually, it is part of four distinct data sets, where we have used the one that was computed with DFT using the RPBE functional [99] with D3 [100] corrections. RPBE was chosen over BLYP because latter shows reduced water dynamics [5]. The D3 corrections account for the London dispersion interactions and are essential for reproducing the anomalies of water [5]. The data set consists of 7241 different structures with varying numbers of atoms, ranging from 16 to 128 water molecules. Seven different ice phases as well as liquid water are represented in the data set. Although not documented, the authors have confirmed that the charges are computed by the Hirshfeld method, see section 3.1.1. The charges are given in multiples of the elementary charge e, whereas the energies are provided in Hartree (E_h) and distances in Bohr (a_0). Consequently, forces are given in Hartree / Bohr.

5.2.2 Charge Training

In 4G the charges are modeled to be Gaussian distributed. This means widths $\sigma_i = \sigma_{[i]}$ have to be assigned, see eq. (2.30). In agreement with [14] we have used the covalent radii of the respective elements. Using the data provided in [101] we chose

$$\sigma_\mathrm{H} = 0.31\,\text{Å} \quad \text{and} \quad \sigma_\mathrm{O} = 0.66\,\text{Å}$$

for hydrogen and oxygen, respectively. The hardness was initialized to $J_\mathrm{H} = J_\mathrm{O} = 18.9/\text{Å}$, while the threshold for the truncation of the Ewald summation was set to $0.51\,\text{meV}/\text{Å}$ with a maximum charge of $q_\mathrm{max} = 0.36\,e$. The cutoffs were determined after optimizing the Ewald parameter η.

[2] A good normalization scheme of the charges in the sense that it improves the fitting quality of the NN is non-trivial to construct.

For the EKF parameters, we mostly followed the suggestion in [16] with one important difference. We started with a learning rate[3] of $\eta = 0.005/e^2$ and increased up to a maximum of $\eta_{\max} = 0.1/e^2$ during training. With the usually recommended setting of $\eta_{\max} = 1/e^2$ we encountered convergence issues in the training.

The training was conducted across 24 MPI tasks for 40 epochs. The selection mode of training data points was random and 10 % of the data set have not been used for training but for the test set.

5.2.3 Energy and Force Training

The inner and outer cutoff radius of the screening function were set to $r_{\mathrm{i,c}} = 2.54\,\text{Å}$ and $r_{\mathrm{c}} = 6.35\,\text{Å}$, respectively, where the latter one should match the symmetry function cutoff. Parameters concerning the Ewald summation were left unchanged. The force error weight was configured to $\beta = 5.29\,\text{Å}$ and only around 2.5 % of the available force components have been considered for the EKF updates. In contrast to the charge training the recommended parameter settings for the EKF in [16] produced satisfying results, so we adhered to $\eta_{\max} = 1.0/{E_{\mathrm{h}}}^2 = 1.35 \times 10^{-3}/\mathrm{eV}^2$. For the energy and force training, we utilized 16 MPI tasks over a period of again 40 epochs. For each epoch, only training data points that produced an error of at least 80 % of the root-mean-square error (RMSE) of the last epoch were considered[4]. Again, 10 % of the data set have been reserved for the test set. For comparison we also trained a 2G NNP with the same setup as the 4G one, apart from the 4G-specific parameters.

5.3 Molecular Dynamics

We constructed a supercell of 128 water molecules with an orthogonal box with lengths

$$a_1 = 17.58\,\text{Å}, \quad a_2 = 15.22\,\text{Å}, \quad a_3 = 14.35\,\text{Å},$$

leading to a density of $\rho_{\mathrm{H_2O}} = 997.047\,\mathrm{kg/m^3}$, which corresponds to Standard Mean Ocean Water (SMOW) at a temperature of $T = 298.15\,\mathrm{K}$ and a pressure of $p = 101\,325\,\mathrm{Pa}$ [102]. This cell was generated by starting from the cell in [103], which is provided by the *Materials Project* [104], and was afterwards replicated and rescaled accordingly.

The MD was conducted with *LAMMPS* under PBCs. Because the force error in the short-range NN overshadowed the truncation error of the Ewald summation, we relaxed the accuracy parameter ε to $2.6\,\mathrm{meV/Å}$, which increased the performance slightly while still keeping the truncation error well below the error that is induced by the NN. Again, the cutoffs were determined after optimizing η. Using this setup, the initial supercell was equilibrated in the NVT ensemble at a temperature of $T = 298.15\,\mathrm{K}$ for a period of 0.5 ns with a time step of 1 fs. The Nosé-Hoover thermostat was used with a dampening value of $\tau_{\mathrm{damp}} = 100\,\mathrm{fs}$.

Afterwards, the equilibrated structure was used in the NVE ensemble with the same time step as before but over a period of 5 ns.

[3] Note that since we have not normalized the training data, entries of matrices appearing in the EKF are in general not dimensionless. And since we combine weights (dimensionless) with hardness (dimension $\frac{1}{\text{length}}$) updates, different entries of the same matrix can have different dimensions. This also means that the suggested values for the parameters do not apply here in general.

[4] Actually, if four consecutively random selected data points are below this threshold, then the one with the biggest error of those is selected to avoid noticeable performance impacts.

5.4 Analysis

5.4.1 Electrostatic Properties

For the analysis, we discarded the trajectories from the equilibration phase and afterwards only used every 100^{th} time step (=100 fs). We computed the histograms of the charges of hydrogen and oxygen as well as the total charge of water molecules. In order to partition all atoms into water molecules, we started with a random oxygen atom and assigned it the two closest hydrogen atoms while respecting PBCs. Those atoms were taken from the set and the scheme was repeated until all molecules had been defined. This method is comparatively efficient but may fail if effects like ionizations are present. In order to support this method, we made a visual inspection of this partition at a random time step and checked regularly if the partition ever changed during the simulation, which did not occur.

Additionally, we were interested in the dipole moment of the water molecules and its distribution. As it turned out, however, the total charge of the water molecules fluctuated non-negligibly around the uncharged state as we will see later in section 6.2. A non-vanishing net charge makes the computation of the dipole moment dependent on the chosen reference point. This is why we have chosen to compute the dipole moment of each molecule with respect to its center of mass (COM) to make it comparable.

We also tested the method for estimating the truncation error of the Ewald summation demonstrated in section 4.1. For this, we computed the forces of 10 structures with a total number of 11520 force components for a variety of different precision parameters ε in conjunction with the optimal value of η. Comparing the forces to the reference values is of little interest in this case because the overall precision is dominated at some point by the error of the short-range force error, when the Ewald precision is increased. So for each ε we computed the differences between all forces and their corresponding values that are obtained with the smallest used value of ε_{min}. From those differences we then computed the standard deviation and the maximum in magnitude.

5.4.2 Performance

Although the interface to *LAMMPS* that we have discussed in section 4.5 is just a prototype, we still tested its (OpenMP) parallel performance. This was done by computing the MD simulation in the NVT ensemble with the same initial configuration as before for 30 time steps with varying numbers of OpenMP threads. As a measure of performance, we used the average number of time steps per second. For several numbers of threads, the same computation was conducted five times and the average was taken over those results. From the time steps per second we computed the speedup

$$\frac{n_{\text{d}t}(N)}{n_{\text{d}t}(1)},$$

where $n_{\text{d}t}(N)$ is the average number of time steps per second when utilizing N OpenMP threads. The test was conducted on two different hardware architectures at the Vienna Scientific Cluster (VSC), one being a node at the VSC-4 with two Intel Xeon Platinum 8174 Processors and the other one being a node at the VSC-5 with two AMD Epyc CPUs (Milan architecture). For the Intel node we compiled *n2p2* with the Intel C++ compiler and the Intel Math Kernel Library whereas the version for the AMD node was compiled with

the GNU compiler and linked to the AMD-optimized BLIS, LAPACK and ScaLAPACK libraries.

Additionally, we also tested the scaling of the computation time in *LAMMPS* when using the interface to *n2p2* by running MD simulations for different numbers of atoms N_i for ten time steps and comparing the average time $\tau = n_{\mathrm{dt}}^{-1}$ that is needed for one time step. For this, we started with a unit cell of two water molecules and replicated it along all three dimensions several times. The overall scaling of the interface with the system size becomes visible for very large systems but for smaller systems algorithms of lower scaling order may still dominate so we checked the computation time in a range of six to 6000 atoms. An interesting quantity to look at is the following

$$k(N_i) = \frac{\log \frac{\tau(N_i)}{\tau(N_{i-1})}}{\log \frac{N_i}{N_{i-1}}}. \tag{5.1}$$

Visually speaking it corresponds to the slope in a double logarithmic plot between consecutive data points. Assuming that the computational effort of the interface can be described as a polynomial, k converges to the degree of this polynomial when $N \to \infty$. The interface consists of many different methods with different scaling behavior. The worst-scaling part is the charge equilibration, which scales with $\mathcal{O}(N^3)$, since it requires a linear system of equations to be solved. However, for small system sizes other computations may dominate the overall computation time. By measuring k over a range of system sizes N we can identify when the cubic scaling becomes noticeable. The scaling test was only conducted on the VSC-4 but for both the 2G and 4G HDNNP in serial as well as parallelized mode, using 16 OpenMP threads. While the accuracy parameter ε of the Ewald truncation method remained the same as in the MD simulation, the real-space cutoff was now fixed to the symmetry function cutoff, so $r_{r,\mathrm{c}} = 6.35\,\text{Å}$. This was done because the optimal η is dependent on the volume of the simulation cell and thus, the resulting real-space cutoff changes considerably during the scaling test. This has noticeable performance impact on *LAMMPS*' routines and would distort the scaling results of *n2p2*'s force calculation.

Chapter 6

Results and Discussion

6.1 Training

In section 4.4.1 we explained that we modified the error vector ξ of the charge training in a way that improves computational performance but violates the statistical assumptions made in the EKF noticeably. Nevertheless, we continued with this approach but recorded the RMSE defined in eq. (4.11) over the training and the test data set. A plot of the RMSE against the epochs of the training is sometimes called *learning curve*. For the charge training, it can be found in fig. 6.1. As we can see, the RMSE of the charges decreases despite

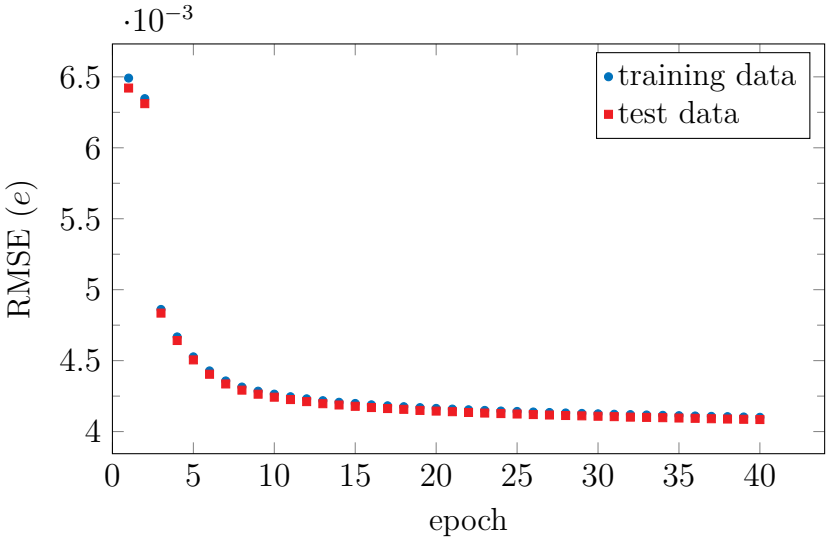


Figure 6.1: Learning curve of the charge prediction during stage 1 training. The RMSE of the NN after each epoch is drawn for both the training and the test data set. The RMSE of epoch 0 has been clipped.

the redefinition of ξ. Furthermore, the learning curve falls off monotonically for both the training and the test data set. Consequently, we have used the weights and biases as well as the hardness from epoch 40. The corresponding RMSEs are $4.10 \times 10^{-3}\, e$ and $4.08 \times 10^{-3}\, e$ for the training and the test data set, respectively. To put this into context, these RMSEs are more than three times smaller than the standard deviation of the hydrogen charges encountered during an MD simulation. This will be discussed in more detail in section 6.2.

The fact that the RMSE for the training and the test data set agree closely indicates that overfitting of the training data does not occur.

Additionally, we visualized the agreement of the individually predicted charges after epoch 40 with the corresponding reference values. Since the complete data set consists of over 5×10^5 charge reference values, we restricted the graph to a subset of around 1 % randomly selected pairs $(q, \tilde{q})$. It can be seen in fig. 6.2. The RMSEs of the individual

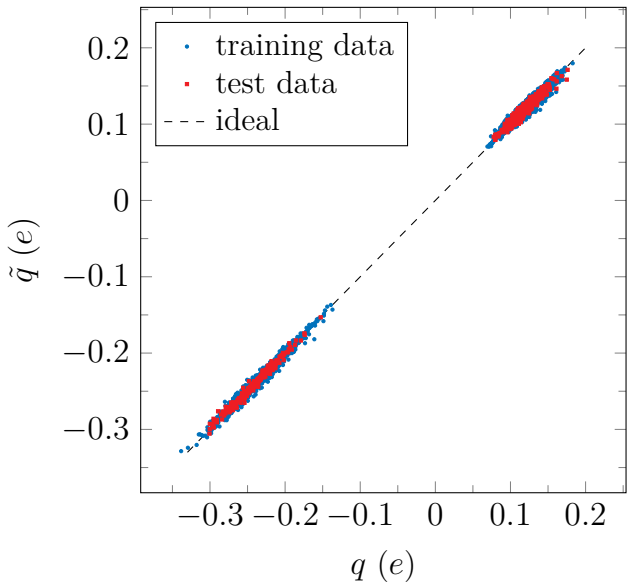


Figure 6.2: Charge agreement between reference charges q and predicted charges $\tilde{q}$ at epoch 40. Note that only randomly selected 1 % of all available pairs $(q, \tilde{q})$ are drawn.

hydrogen or oxygen charge prediction computed over the test data set are $4.15 \times 10^{-3}\, e$ and $3.96 \times 10^{-3}\, e$, respectively. From this, we conclude that the charges of oxygen (negatively charged) and hydrogen (positively charged) are predicted with comparable accuracy. We also do not notice any accuracy problems with charge values that appear only rarely in the data set. Again, no signs of overfitting are detected.

The energy and force training can be discussed analogously. The learning curves of both quantities are shown in fig. 6.3. In comparison to the charge training the decrease in both the energy and force prediction error is not monotonous but fluctuates. To understand why this is not particularly surprising, let us first recall that the EKF was set up in a way that minimizes the combined error

$$\mathbb{E}\left[\frac{1}{m_U}\sum_{i=1}^{m_U}\left(U(\boldsymbol{r}_i^{3N}) - \tilde{U}(\boldsymbol{r}_i^{3N})\right)^2 + \frac{\beta^2}{m_F}\sum_{\alpha\in I}\left(\left[\boldsymbol{F}_{\alpha_\mathrm{a}}(r_{\alpha_\mathrm{s}}^{3N}) - \tilde{\boldsymbol{F}}_{\alpha_\mathrm{a}}(r_{\alpha_\mathrm{s}}^{3N})\right]^{\alpha_\mathrm{c}}\right)^2\right]$$

with respect to the weights and biases. Here we have used the same symbols as in eq. (2.21). For each energy update, the covariance matrix $\boldsymbol{P}$ of the EKF does not only take previous energy errors into account but also incorporates information from force errors. Analogous holds for the force updates. Thus, each energy or force update may not minimize the pure force or energy error, respectively, but the combined error of those quantities.

The learning curves of a 2G NNP with the same setup apart from the electrostatic part is also shown. It converges to a lower RMSE after the same number of epochs regarding both the forces and energies. Although it was shown in [14] that a 4G NNP can in principle

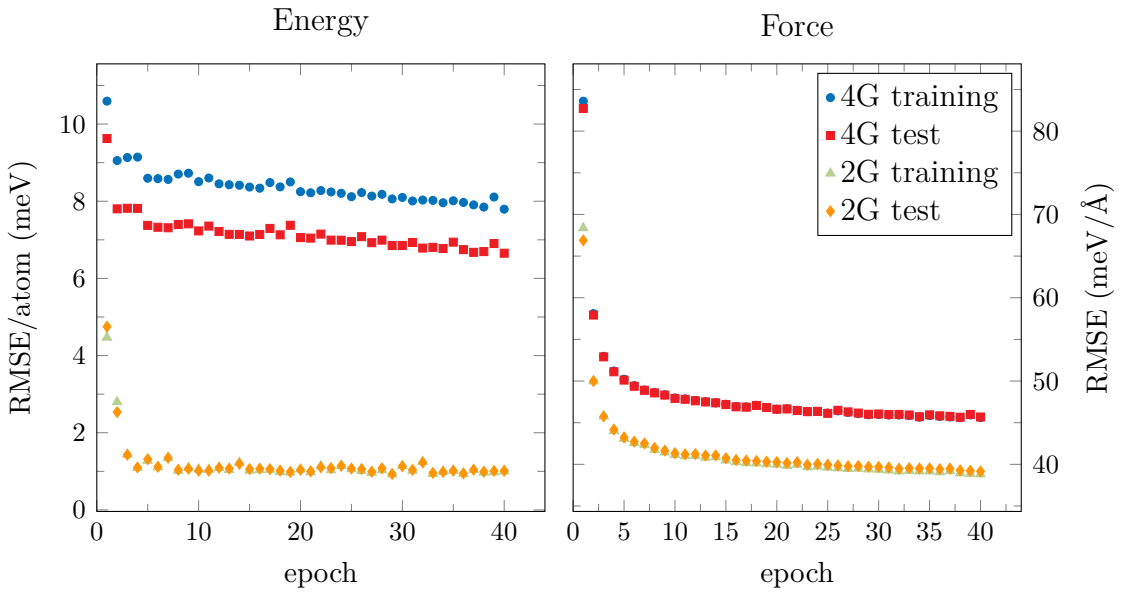


Figure 6.3: Learning curves of the energy and force prediction during stage 2 training for the 4G NNP. For comparison also the learning curve of a corresponding 2G NNP is shown. Regarding the energy, the RMSE of the NN per atom is plotted, whereas for the forces only the RMSE is drawn. The values of each epoch are shown for both the training and the test data set, apart from epoch 0, which has been clipped.

outperform the predecessor in systems that feature long-ranged charge transfer, it can also degrade the overall predictive performance. We did, however, not test if this qualitative difference can be removed by changing the set of symmetry functions, NN architecture or the charge partition method. The latter may have the biggest impact, since as we will discuss below, the Hirshfeld charges could not describe the molecular dipole moment of liquid water accurately. But dipole-dipole interactions are long-ranged and since the water molecules can be considered approximately neutrally charged, they dipole-dipole interaction is the largest contribution to the electrostatic interaction.

Interestingly, the RMSE of the energy prediction of the 4G NNP computed for the test data set is about 14 % smaller than the corresponding RMSE for the training data set. Although we do not know why this is the case, it is not unusual and can also be observed in other works, e.g. [16]. One could try to train another NNP on the same data set but with a different test and training data set splitting. Nevertheless, since the error over the test set is smaller, overfitting is not indicated.

The NNP was trained in order to conduct an MD simulation afterwards. Consequently, the accuracy of the force prediction is prioritized. This led to the choice of using weights and biases from epoch 38, where the RMSE of the forces was the lowest when computed over the test data set. To be precise, the energy RMSE per atom at this epoch was 6.70 meV while the RMSE of the forces was 45.7 meV/Å. Both values refer to the test data set. The 2G NNP achieved its best performance at epoch 40 where the energy RMSE per atom was 1.01 meV and the force RMSE equaled 39.2 meV/Å. While the energy error is considerably smaller compared to the 4G NNP, the force error is of similar magnitude.

Again we compared the predicted and reference values of the data set for forces and energies, which can be seen in fig. 6.4. Regarding the forces, only about 0.3 % randomly selected pairs $(F^i, \tilde{F}^i)$ are shown. Both plots indicate a close agreement, even for rarely

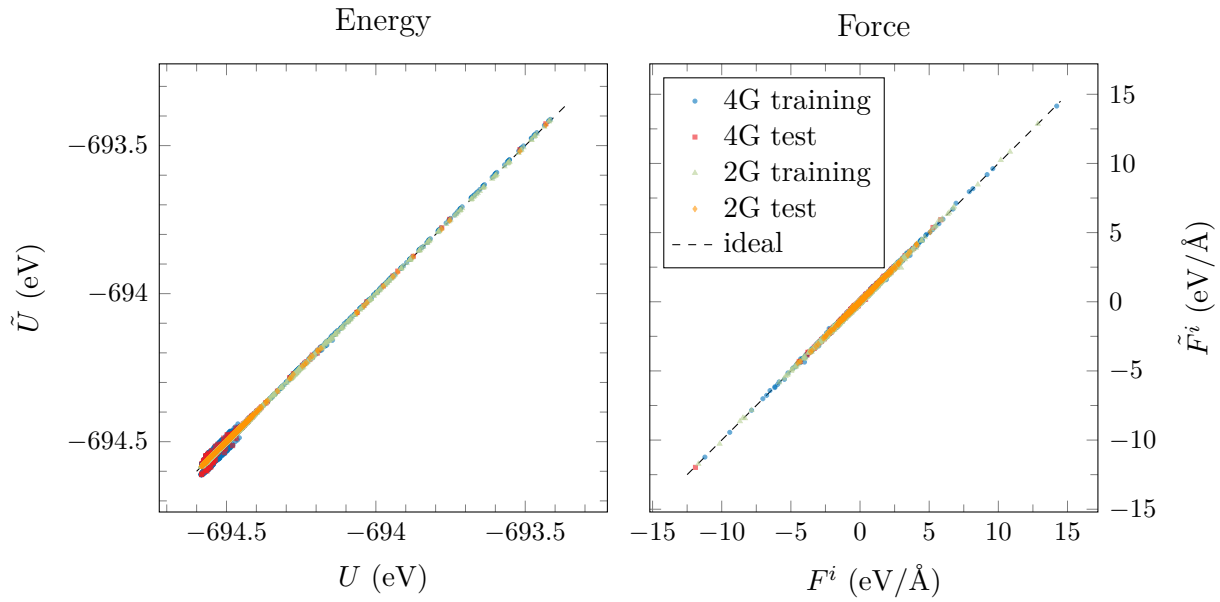


Figure 6.4: Force and energy agreement between reference values U, F^i and predicted ones $\tilde{U}$, $\tilde{F}^i$ at for the 4G NNP at epoch 38 and the 2G NNP at epoch 40 for both the training and the test data set. Only randomly selected 0.3 % of all force component pairs $(F^i, \tilde{F}^i)$ are drawn.

occurring values of energies and forces. Outliers are not visible in the data. While the 4G and 2G seem to match closely when predicting forces or high energies, their difference seems to occur in the prediction of structures with comparably low potential energy, though this regime makes up most of the training data set.

6.2 Electrostatic Properties

The histograms of the hydrogen and oxygen charges as well as water molecule net charges during an MD simulation are shown in fig. 6.5. The bin width was set to $1 \times 10^{-3}\,e$. The mean and standard deviation $(\bar{q}, \sigma_q)$ of this distributions are $(-0.239\,e, 0.022\,e)$ for oxygen, $(0.119\,e, 0.014\,e)$ for hydrogen and $(0.000\,e, 0.038\,e)$ for water molecules. The corresponding normal distributions are also drawn in fig. 6.5, which agree well with the histograms of oxygen and the water molecules but deviate more noticeably from the histogram of hydrogen. As mentioned before, the RMSE of the charge prediction is distinctly smaller than the fluctuations of the charge distributions. More precisely, the fluctuations are more than three times larger than the RMSE of the hydrogen charge prediction and more than five times larger than the RMSEs of the charges for oxygen and water molecules[1]. This makes the charge prediction of the NN suitable for meaningful charge analysis at only a fraction of the cost of other methods that first need to compute the electron density before they can assign atomic charges, see section 3.1.

In [68] the distribution of oxygen charges in liquid water has been analyzed using three different charge partition schemes. The averages of all three methods (CHelpG [105], Iterative Hirshfeld, and natural population analysis (NPA) [106]) lie in the range $-1.05\,e$ to

[1]For the RMSE of the water molecule net charge we have assumed that it can be computed approximately as the sum of three independently normal distributed errors of the corresponding atoms.

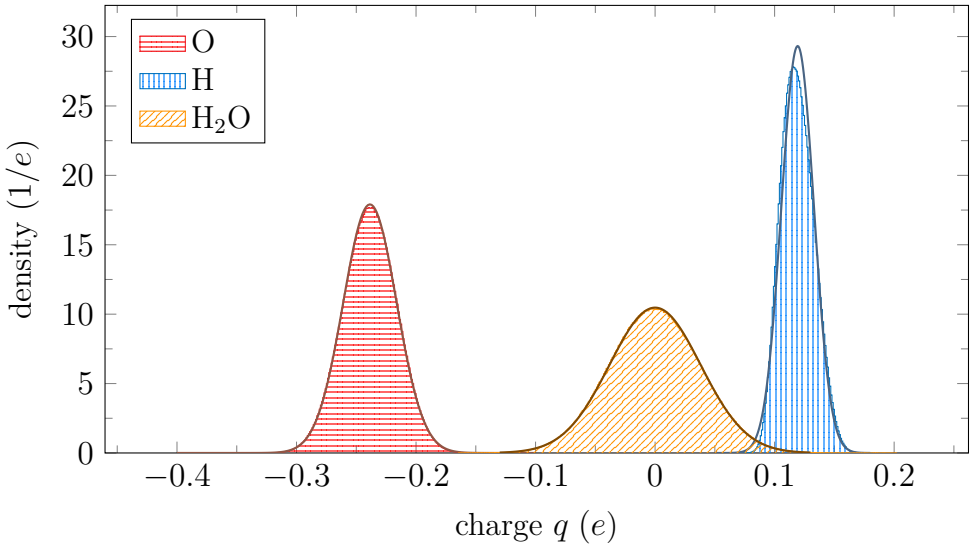


Figure 6.5: Normalized histograms of the charges of hydrogen, oxygen and water obtained during an MD simulation. The bin width is $1 \times 10^{-3}\,e$. Additionally, normal distributions with the corresponding mean and standard deviation (see section 6.2) are drawn.

$-0.75\,e$. This discrepancy in our results is, however, a result of the Hirshfeld method that was used for the training of the NN. Han et al. [68] discuss that the Hirshfeld method fails to describe molecular dipoles accurately, which we can verify in this work as well. In fig. 6.6 the distribution of the magnitude of the dipole moments of the water molecules is drawn. It has a mean of $\|\boldsymbol{\mu}_{H_2O}\| = 0.693\,\mathrm{D}$ with a standard deviation of $0.058\,\mathrm{D}$. Comparing this

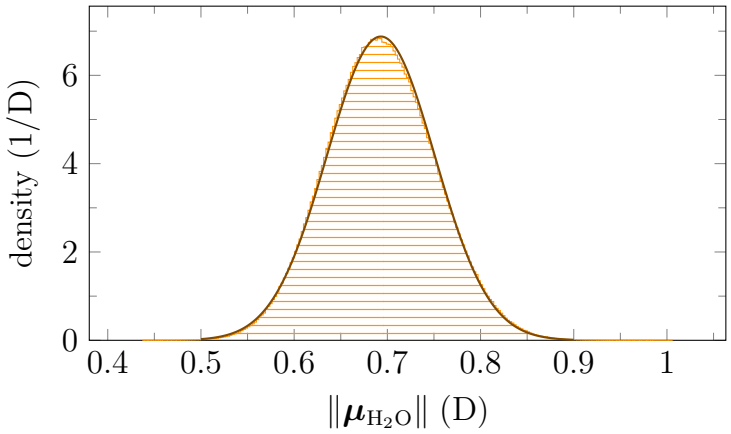


Figure 6.6: Normalized histogram of the magnitude of all water dipole moments $\boldsymbol{\mu}_{H_2O}$. The bin width is 0.5 mD. A normal distribution with the corresponding mean and standard deviation (see section 6.2) is also drawn.

with the experimental results of 2.9 D at 297.6 K [107] and 2.95 D at 300 K [108] shows that the Hirshfeld scheme underestimates the dipole moment of individual water molecules in liquid water by a factor of four. In [68] the Hirshfeld method is not used for the dipole moment calculation in liquid water, but they demonstrated that this method underestimates the dipole moment in various different water clusters of varying size. The computed average dipole moments are between 0.7 D to 0.8 D, resembling our results in liquid water closely. As

discussed before, the inaccurate description of the water molecule dipole may be the reason for the degraded energy prediction when compared to the 2G NNP.

The truncation method for the Ewald summation was tested for multiple values of the accuracy parameter ε in the range of $5.1 \times 10^{-1}\,\mathrm{eV/Å}$ to $5.1 \times 10^{-11}\,\mathrm{eV/Å}$. The comparison with the standard deviation $\sigma(F^i - F^i_{\varepsilon_{\min}})$ of the error and the maximum error is listed in table 6.1. As explained in section 5.4.1 the errors are computed with respect to the forces

Table 6.1: Empirical standard deviation of the differences $F^i - F^i_{\varepsilon_{\min}}$ for different values of the accuracy parameter ε used in the truncation scheme of the Ewald summation. $F^i_{\varepsilon_{\min}}$ are the forces with respect to the smallest used value of ε, which corresponds to the last row. All quantities with the dimension of a force are given in eV/Å.

ε	$\sigma(F^i - F^i_{\varepsilon_{\min}})$	$\max_i\{F^i - F^i_{\varepsilon_{\min}}\}$	$\sigma(F^i - F^i_{\varepsilon_{\min}})/\varepsilon$	$\max_i\{F^i - F^i_{\varepsilon_{\min}}\}/\varepsilon$
5.1	1.0×10^{-1}	6.0×10^{-1}	0.020	0.116
5.1×10^{-1}	2.2×10^{-2}	1.2×10^{-1}	0.042	0.232
5.1×10^{-2}	4.3×10^{-3}	2.6×10^{-2}	0.084	0.505
5.1×10^{-3}	4.2×10^{-4}	2.1×10^{-3}	0.081	0.406
5.1×10^{-4}	3.9×10^{-5}	2.3×10^{-4}	0.076	0.446
5.1×10^{-5}	4.1×10^{-6}	2.2×10^{-5}	0.080	0.423
5.1×10^{-6}	3.9×10^{-7}	2.0×10^{-6}	0.076	0.387
5.1×10^{-7}	3.8×10^{-8}	1.8×10^{-7}	0.074	0.347
5.1×10^{-8}	4.1×10^{-9}	2.5×10^{-8}	0.080	0.485
5.1×10^{-9}	4.6×10^{-10}	2.1×10^{-9}	0.090	0.410
5.1×10^{-10}	5.3×10^{-11}	2.8×10^{-10}	0.103	0.540
5.1×10^{-11}	-	-	-	-

at accuracy $\varepsilon_{\min} = 5.1 \times 10^{-11}\,\mathrm{eV/Å}$ For easy comparison, the ratios of those quantities with respect to ε are also given. Clearly, the implemented method makes a pessimistic estimate about the truncation error, since the actual standard deviation is always smaller by a factor of at least ten for the liquid water system we have used. This is not particularly surprising because in the derivation we adapted the original method of [92] by introducing upper bounds of the truncation error such that we arrive at a formula resembling the original approach. Even the maximum error for this particular data set is always smaller than the specified parameter ε. From this test we conclude that the chosen accuracy parameter of $\varepsilon = 2.6\,\mathrm{meV/Å}$ was quite high in combination with a NNP yielding an overall RMSE of $45.7\,\mathrm{meV/Å}$ in the force prediction. We remark that the results in table 6.1 do not directly transfer to other materials with different charge distributions because of the assumption that were made in section 4.1. In the original work of Kolafa and Perram [92] and in the work of [93] the empirical error agreed very well with the parameter ε, but both tested this particular method only for molten NaCl[2], so their results may not directly transfer to liquid water.

6.3 Performance

The speedup of the prototyping interface is shown in fig. 6.7. Ideal speedup would mean

[2]Kolafa and Perram [92] also investigated truncation errors of diethyl ether, but a different error estimate was used for this study.

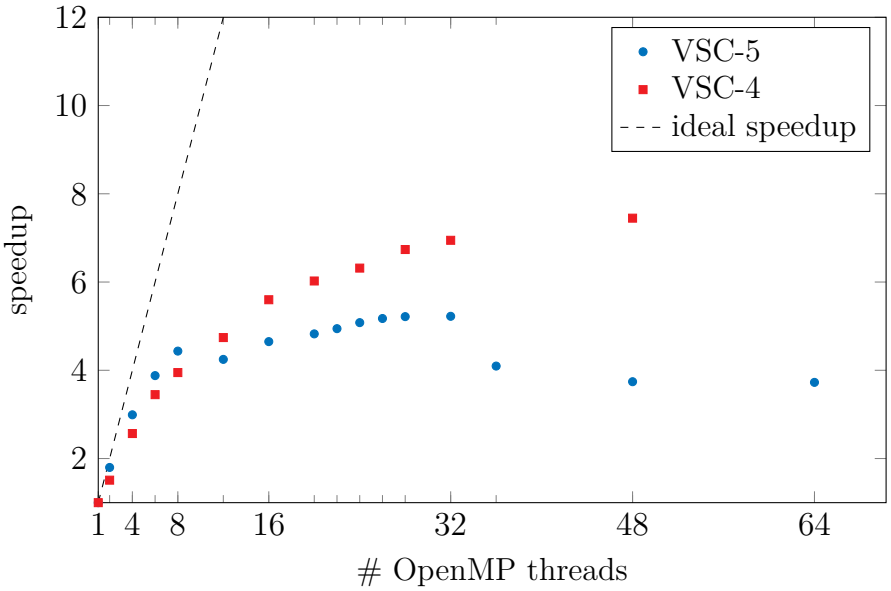


Figure 6.7: Speedup of the *LAMMPS* interface simulating 384 atoms as a function of the number of OpenMP threads. The speedup was tested on the VSC-4 and the VSC-5. The dashed line indicates linear speedup which corresponds to perfectly sharing the workload on all threads.

that if we utilize N OpenMP threads, then $n_{\mathrm{dt}}(N) = N n_{\mathrm{dt}}(1)$. In practice, this is rarely achieved because of additional overhead resulting from forking and joining threads and serial methods that are part of the code and cannot be parallelized[3]. We observe that for both CPU architectures, a reasonable speedup is achieved up to around eight threads. Increasing the number of threads even further is of little use and may eventually even decrease the performance because the parallelization overhead becomes more expensive than the gain in reduced execution time in the parallelized routines. For this interface, the bottleneck for parallelization is the charge equilibration step. While costly routines such as the symmetry function evaluation and the force computation are parallelized, the equation system in the charge equilibration is still running in serial. The number of atoms in the structure dictate the overall effort that is needed for this routine. In fig. 6.7 one can also see an artifact caused by the AMD Epyc Milan architecture of the VSC-5. Namely, the drop in the speedup when going from eight to twelve threads. This is most likely caused by the fact that this CPU is grouped into eight core complexes, which are connected by AMD's Infinity Fabric and each core complex consists of eight cores. When using more than eight threads, the parallelization overhead is affected by the slower communication speed between the core complexes. The speedup does not tell anything about the absolute performance of the interface, however. The maximum number of time steps were achieved on an AMD CPU with 32 cores, which completed 14.12 time steps per second compared to a serial performance of 2.70 time steps per second.

The computational time that is needed per time step is plotted in fig. 6.8 (left) for different numbers of atoms. The results are shown for both serial and parallel runs with 2G and 4G HDNNPs. Since we are interested in the point where the cubic scaling of the charge

[3]Actually, superlinear scaling can sometimes be observed but this is highly dependent on the hardware architecture and is often related to improved cache utilization.

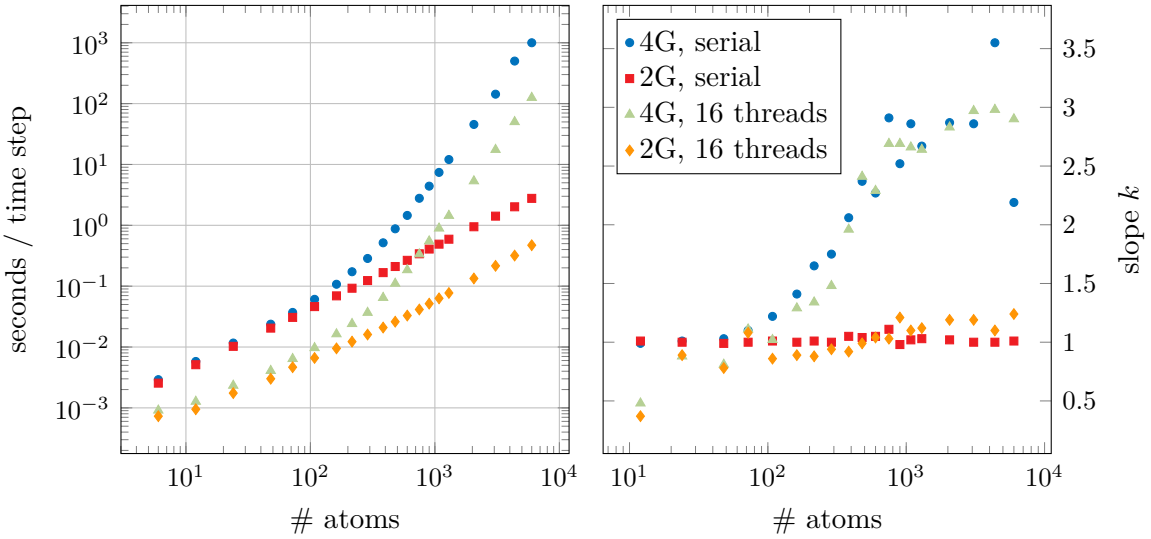


Figure 6.8: Scaling of the simulation time in *LAMMPS* using the interface to *n2p2* for different numbers of atoms in the system. The results are shown for 2G and 4G HDNNPs in combination with both serial and multithreaded execution (using OpenMP parallelization). On the left-hand side, the average simulation time per time step is plotted, whereas on the right hand side, the corresponding slopes between consecutive points in the double-log plot are drawn, which should asymptotically converge to the scaling order k (see eq. (5.1)) of the simulation with respect to the number of atoms.

equilibration becomes visible, we plotted the slopes k of the left diagram on the right hand side. For up to around 100 atoms, all setups seem to scale linearly with the system size and the performance of the 2G and 4G NNPs match closely, which indicates that the charge equilibration is negligible in this regime. This is also in agreement with the small difference between the parallelized runs since the equation system in the charge equilibration is solved in serial. For those runs we even notice sublinear scaling up to around 50 atoms when using 16 threads, from which we conclude that computations that are independent of the system size make up a comparatively large fraction. The serial 4G run is transitioning from linear to cubic scaling in the range of hundred to thousand atoms. This is a sign that the charge equilibration starts to make up a non-vanishing fraction of the overall computational effort. For the parallelized run, this transition happens even slightly faster because many costly routines are parallelized but, as mentioned before, not the process of solving the equation system. This is why OpenMP parallelization becomes less efficient for larger systems. The last two points of the slopes k for the 4G serial run fluctuate noticeably, which may be due to hardware-specific reasons. In comparison, both 2G runs show that they scale linearly with the system size everywhere in the tested regime. For 2G NNPs, the most expensive part to compute are usually the symmetry functions. This calculation scales with the number of neighbors and since neighbor lists are constructed by *LAMMPS* via Verlet lists, this part behaves linearly.

We conclude that the prototyping interface to *LAMMPS* is usable for system sizes of up to a few hundred atoms and can make use of around as many cores as are typically found on decent personal computer hardware. Large-scale systems like scientific clusters cannot be taken advantage of at the moment because of the limited speedup with an increasing number of threads and the limitation of shared memory parallelization, which restricts the program

to be parallelized only on CPUs that are connected to the same memory.

Chapter 7

Conclusion and Outlook

In this thesis, we started from the recently published work of Ko et al. [14], who introduced the next generation (4G) of the Behler-Parrinello HDNNPs. The goal was to implement this approach into *n2p2*, an efficient, object-oriented C++ program that was initially developed for 2G HDNNPs to provide parallelized multistream EKF training of the NNs and a direct interface to the popular MD simulation package *LAMMPS*.

First, we gave a concise introduction to the statistical learning problem and the Kalman filter in chapter 2, including its variants. We discussed how the (multistream) EKF can be applied to the training of NNPs and why it is the preferred optimization algorithm for NNPs, although it is rarely found in recent deep learning projects due to its unfavorable scaling with the size of the NNs. We further elaborated on the theory behind Behler-Parrinello HDNNPs and formalized the mathematical requirements of the symmetry functions, which lie at the core of the HDNNPs.

Afterwards, the improvements introduced by 4G HDNNPs were explained. Since those are obtained by explicitly modeling atomic charges through the help of a charge equilibration step, we discussed the details of this procedure and its nuances. In particular, we emphasized the ambiguity of the electrostatic energy that arises when PBCs are applied and how this is handled in the Ewald summation. In this context we made clear that although a unit cell under PBCs has the topology of a flat torus, this point of view is unsuitable to relate the ambiguity of the electrostatic energy to different boundary conditions of large but finite systems.

When implementing the Ewald summation, the necessity for a sophisticated truncation scheme arose. Starting from one of the existing methods, we modified it to fit into our application, in which we are using a priori unknown Gaussian distributed charges instead of specified point charges. When testing this method later with liquid water, the error in the electrostatic forces stayed well below the threshold and thus provides a reliable way of choosing the necessary cutoffs in the Ewald summation. However, the actually measured error was typically one order of magnitude smaller, which indicates that the scheme may be too pessimistic, whereby computational effort is higher than necessary. For a profound conclusion, the method still needs to be tested for systems with different charge distributions.

Computational details like memory concerns, parallelization and overall performance have been discussed in detail for both the training and the prediction mode. While the interface to *LAMMPS* was constructed with prototyping concerns in mind, it still seems to yield reasonable performance and parallel speedup for hardware that is usually found on personal computers. We showed that although the cubic scaling caused by solving an

equation system is unfavorable, the overall impact of this behavior only becomes noticeable when going beyond a few hundred atoms.

Finally, we gave a preview of how the predicted atomic charges can be used in MD. We computed the charge distributions of atoms and molecules and also the dipole moment distribution of water. Although the dipole moment did not agree with the experimental values, we could relate this problem to the Hirshfeld charge partition scheme. Since the NNP learns the charges during the training, there is in principle no problem in changing the charge partition method. Thus we expect much better results if the same test will be conducted e.g. with the iterative Hirshfeld method. Correspondingly, it should also improve the energy and force prediction of the NNP. Furthermore, it was verified that the charge prediction error is distinctly smaller than the overall charge fluctuations in liquid water during an MD simulation.

In conclusion, the 4G HDNNP method was successfully implemented into *n2p2* and provides new capabilities like long-range charge transfer, which the previously implemented 2G version could, by construction, not capture. However, this was not tested in this work since Ko et al. [14] have demonstrated this already. For systems where this effect does not occur, 4G HDNNPs may cause unnecessary computational overhead and one should resort to 2G or 3G HDNNPs. If the atomic charge model is inappropriate for modeling the electrostatic interactions of the material, the energy and force prediction error may even increase compared to the 2G HDNNP. However, for a sophisticated conclusion, further studies of the influence of the charge partition method are necessary. Regarding the computational performance aspects, there is now ongoing work together with the Behler group about the development of a production-ready 4G interface between *n2p2* and *LAMMPS*. This new interface will handle the charge equilibration inside *LAMMPS*, which will enable the use of *LAMMPS*' inbuilt MPI parallelization. Furthermore, a gradient descent approach will be implemented for the charge equilibration step, which is expected to improve the computational performance for larger systems. We expect that the 4G HDNNPs implementation into *n2p2* will be a valuable tool in future studies of various materials.

Appendix A

Interaction of Gaussian Charges

A.1 Potential Energy of Two Gaussian Charges

Let ρ_1 and ρ_2 be two Gaussian charge distributions of the form

$$\rho_i(\boldsymbol{x}) = \frac{q_i}{(2\pi\sigma_i^2)^{\frac{3}{2}}} e^{-\frac{\|\boldsymbol{x}-\boldsymbol{r}_i\|^2}{2\sigma_i^2}},$$

with charges q_1, q_2 and widths σ_1, σ_2 centered at $\boldsymbol{r}_1$, $\boldsymbol{r}_2$, respectively. We are interested in calculating the integral

$$U = \int_{\mathbb{R}^3} \mathrm{d}^3\boldsymbol{x} \int_{\mathbb{R}^3} \mathrm{d}^3\boldsymbol{y}\, \frac{\rho_1(\boldsymbol{x})\rho_2(\boldsymbol{y})}{\|\boldsymbol{x}-\boldsymbol{y}\|}, \tag{A.1}$$

which represents the electrostatic energy associated with ρ_1 in the field generated by ρ_2 or vice versa. This integral is invariant under a simultaneous translation or rotation of $\boldsymbol{x}$, $\boldsymbol{y}$ and $\boldsymbol{r}_i$. Thus the result can only depend on the distance $r_{ij} = \|\boldsymbol{r}_i - \boldsymbol{r}_j\|$ and we can assume without loss of generality $\boldsymbol{r}_1 = \boldsymbol{0}$. For notational convenience, we will abbreviate the normalizing constants of ρ_1 and ρ_2 with

$$N_i = \frac{q_i}{(2\pi\sigma_i^2)^{\frac{3}{2}}}.$$

We can rewrite eq. (A.1) as

$$N_1 N_2 \int_{\mathbb{R}^3} \mathrm{d}^3\boldsymbol{y}\, e^{-\frac{\|\boldsymbol{y}-\boldsymbol{r}_2\|^2}{2\sigma_2^2}} \underbrace{\int_{\mathbb{R}^3} \mathrm{d}^3\boldsymbol{x}\, \frac{1}{\|\boldsymbol{x}-\boldsymbol{y}\|} e^{-\frac{\|\boldsymbol{x}\|^2}{2\sigma_1^2}}}_{=:A}.$$

Changing to spherical coordinates (r_x, θ_x, ϕ_x),(r_y, θ_y, ϕ_y) for $\boldsymbol{x}$ and $\boldsymbol{y}$, respectively, and choosing the z-axis of $\boldsymbol{x}$ along $\boldsymbol{y}$ the denominator in A can be written as

$$\|\boldsymbol{x}-\boldsymbol{y}\| = \sqrt{r_x^2 - 2r_x r_y \cos\theta_x + r_y^2}. \tag{A.2}$$

Further substituting $u \coloneqq \cos\theta_x$ leads to

$$A = 2\pi \int_0^\infty \mathrm{d}r_x\, r_x^2 e^{-\frac{r_x^2}{2\sigma_1^2}} \underbrace{\int_{-1}^1 \mathrm{d}u\, \frac{1}{\sqrt{r_x - 2r_x r_y u + r_y}}}_{=:B}.$$

It follows that

$$B = \frac{1}{r_y r_x}\left(|r_y + r_x| - |r_y - r_x|\right) = \begin{cases} \frac{2}{r_y} & \text{if } r_x \leq r_y \\ \frac{2}{r_x} & \text{if } r_x > r_y. \end{cases}$$

Hence,

$$A = 2\pi\left(\int_0^{r_y} \mathrm{d}r_x\ \frac{2}{r_y} r_x^2 e^{-\frac{r_x^2}{2\sigma_1^2}} + \int_{r_y}^{\infty} \mathrm{d}r_x\ 2r_x e^{-\frac{r_x^2}{2\sigma_1^2}}\right).$$

Computing the integrals one arrives at

$$A = (2\pi\sigma_1)^{\frac{3}{2}} \frac{\operatorname{erf}\left(\frac{r_y}{\sqrt{2}\sigma_1}\right)}{r_y} \tag{A.3}$$

Using this we can express eq. (A.1) as

$$U = (2\pi\sigma_1^2)^{\frac{3}{2}} N_1 N_2 \underbrace{\int_{\mathbb{R}^3} \mathrm{d}^3\boldsymbol{y}\ \frac{1}{\|\boldsymbol{y}\|} \operatorname{erf}\left(\frac{\|\boldsymbol{y}\|}{\sqrt{2}\sigma_1}\right) e^{-\frac{\|\boldsymbol{y}-\boldsymbol{r}_2\|^2}{2\sigma_2^2}}}_{=:C} \tag{A.4}$$

For the integration over $\boldsymbol{y}$ one can choose the z-axis along $\boldsymbol{r}_2$ and rewrite $\|\boldsymbol{y} - \boldsymbol{r}_2\|$ analogously to eq. (A.2). By making use of spherical coordinates once again and using the substitutions $w \coloneqq \cos\theta_y$ and $r' = \frac{r_y}{\sqrt{2}\sigma_1}$ one can write the integral as

$$C = 4\pi\sigma_1^2 e^{-\frac{r_2^2}{2\sigma_2^2}} \int_0^{\infty} \mathrm{d}r'\ r' e^{-\left(\frac{\sigma_1}{\sigma_2} r'\right)^2} \operatorname{erf}(r') \int_{-1}^{1} \mathrm{d}w\ e^{\sqrt{2}\frac{\sigma_1}{\sigma_2} r' r_2 w}.$$

Out of convenience we introduce $\alpha \coloneqq \sqrt{2}\frac{\sigma_1}{\sigma_2^2} r_2$ and arrive at

$$\begin{aligned} C =& \frac{4\pi\sigma_1^2}{\alpha} e^{-\frac{r_2^2}{2\sigma_2^2}} \int_0^{\infty} \mathrm{d}r'\ e^{-\left(\frac{\sigma_1}{\sigma_2} r'\right)^2} \operatorname{erf}(r') \left(e^{\alpha r'} - e^{-\alpha r'}\right) \\ =& -\frac{4\pi\sigma_1^2}{\alpha} e^{-\frac{r_2^2}{2\sigma_2^2}} \int_{-\infty}^{\infty} \mathrm{d}r'\ e^{-\left(\frac{\sigma_1}{\sigma_2} r'\right)^2 - \alpha r'} \operatorname{erf}(r') \\ =& -\frac{4\pi\sigma_1^2}{\alpha} e^{-\frac{r_2^2}{2\sigma_2^2} + \left(2\frac{\sigma_2}{\sigma_1}\alpha\right)^2} \int_{-\infty}^{\infty} \mathrm{d}r'\ e^{-\left(\frac{\sigma_1}{\sigma_2} r' + \frac{r_2}{\sqrt{2}\sigma_2}\right)^2} \operatorname{erf}(r') \end{aligned}$$

The argument of the exponential outside the integral vanishes when plugging α back in and the integral can be evaluated by using [109]

$$\int_{-\infty}^{\infty} \mathrm{d}x\ \operatorname{erf}(x)\, e^{-(ax+b)^2} = -\frac{\sqrt{\pi}}{a} \operatorname{erf}\left(\frac{b}{\sqrt{a^2+1}}\right).$$

This leads to

$$C = (2\pi)^{\frac{3}{2}} \frac{\sigma_2^3}{r_2} \operatorname{erf}\left(\frac{r_2}{\sqrt{2(\sigma_1^2 + \sigma_2^2)}}\right).$$

Plugging the result back into eq. (A.4) we finally arrive at

$$U = \frac{q_1 q_2}{r_{12}} \operatorname{erf}\left(\frac{r_{12}}{\sqrt{2(\sigma_1^2 + \sigma_2^2)}}\right),$$

where we have replaced r_2 with r_{12} to take the general case without the particular choice $\boldsymbol{r}_1 = \boldsymbol{0}$ into account.

A.2 Positivity of the Potential Energy

We will show the positivity of the electrostatic potential between Gaussian distributed charges. First, we start with the following expression of the electrostatic energy

$$U^{\text{elec}} = \frac{1}{2}\int_{\mathbb{R}^3} \mathrm{d}^3\boldsymbol{x} \int_{\mathbb{R}^3} \mathrm{d}^3\boldsymbol{y}\, \frac{\rho(\boldsymbol{x})\rho(\boldsymbol{y})}{\|\boldsymbol{x}-\boldsymbol{y}\|} = \frac{1}{2}\int_{\mathbb{R}^3} \mathrm{d}^3\boldsymbol{x}\rho(\boldsymbol{x})\phi(\boldsymbol{x}) \,,$$

provided that the integrals do not diverge for ρ, which is the case for Gaussian distributed charges. The electric potential ϕ is defined as

$$\phi(\boldsymbol{x}) = \int_{\mathbb{R}^3} \mathrm{d}^3\boldsymbol{y}\, \frac{\rho(\boldsymbol{y})}{\|\boldsymbol{x}-\boldsymbol{y}\|}$$

and satisfies $\Delta\phi = -4\pi\rho$. We can rewrite U^{elec} as

$$\begin{aligned} U^{\text{elec}} =& -\frac{1}{8\pi}\int_{\mathbb{R}^3} \mathrm{d}^3\boldsymbol{x}\, \phi(\boldsymbol{x})\Delta\phi(\boldsymbol{x}) \\ =& -\frac{1}{8\pi}\left[\int_{\mathbb{R}^3} \mathrm{d}^3\boldsymbol{x}\, \operatorname{div}(\phi\Delta\phi) - \int_{\mathbb{R}^3} \mathrm{d}^3\boldsymbol{x}\, \boldsymbol{\nabla}\phi\cdot\boldsymbol{\nabla}\phi\right] \\ =& \frac{1}{8\pi}\lim_{R\to\infty}\left[\int_{\partial B_R} (\mathrm{d}\boldsymbol{S}\cdot\boldsymbol{E})\phi + \int_{B_R} \mathrm{d}^3\boldsymbol{x}\, \|\boldsymbol{E}\|^2\right]. \end{aligned}$$

In the last step, we replaced the integral $\int_{\mathbb{R}^3}$ with $\lim_{R\to\infty}\int_{B_R}$, where B_R denotes the ball centered at $\mathbf{0}$ with radius R. Additionally, we made use of the divergence theorem and used $\boldsymbol{E} = -\boldsymbol{\nabla}\phi$. The second integral is clearly non-negative and is even positive for any non-vanishing $\boldsymbol{E}$.

If $\phi\boldsymbol{E}$ decays fast enough, the first integral vanishes when taking the limit. For Gaussian distributed charges this is easy to show. Because the integral is over the surface of a ball with radius R we can simplify $\mathrm{d}\boldsymbol{S}\cdot\boldsymbol{E} = E^r R^2\,\mathrm{d}\varphi\mathrm{d}\theta$. E^r is the radial component of $\boldsymbol{E}$ and φ and θ denote the azimuthal and the polar angle, respectively. Since we are considering a sum of Gaussian distributed charges, we decompose $\phi\boldsymbol{E} = \sum_{i,j}\phi_i\boldsymbol{E}_j$, where ϕ_i and $\boldsymbol{E}_i$ are the potential and the field generated by charge i. Thus, for finitely many Gaussian charges it is enough to show that the surface integral over $\phi_i E_j^r$ tends to zero as R goes to infinity. The potential ϕ_i of a charge centered at the origin has already appeared in eq. (A.3) up to a normalizing constant. If the charge is centered at $\boldsymbol{r}_i$ we obtain

$$\phi_i(\boldsymbol{x}) = q_i\frac{\operatorname{erf}\left(\frac{\|\boldsymbol{x}-\boldsymbol{r}_i\|}{\sqrt{2}\sigma_i}\right)}{\|\boldsymbol{x}-\boldsymbol{r}_i\|}.$$

By using the spherical coordinates $\boldsymbol{x} = (r,\theta,\phi)$ with the z-axis along $\boldsymbol{r}_i$, we can write E_i^r as

$$E_i^r = -\frac{\partial}{\partial r}\phi_i = -q_i\left[\sqrt{\frac{2}{\pi\sigma_i^2}}e^{-\frac{\|\boldsymbol{x}-\boldsymbol{r}_i\|^2}{2\sigma_i^2}}\frac{r-r_i\cos\theta}{\|\boldsymbol{x}-\boldsymbol{r}_i\|^2} - \operatorname{erf}\left(\frac{\|\boldsymbol{x}-\boldsymbol{r}_i\|}{\sqrt{2}\sigma_i}\right)\frac{r-r_i\cos\theta}{\|\boldsymbol{x}-\boldsymbol{r}_i\|^3}\right].$$

Under the condition $r \geq r_i, r_j$ we can find the upper bound

$$|\phi_i E_j^r|(\boldsymbol{x}) \overset{\text{if } r\geq r_i, r_j}{\leq} \frac{|q_i q_j|}{r-r_i}\left[\sqrt{\frac{2}{\pi\sigma_j^2}}e^{-\frac{(r-r_j)^2}{2\sigma_j^2}}\frac{r+r_j}{(r-r_j)^2} + \frac{r+r_j}{(r-r_j)^3}\right].$$

This upper bound is independent of φ and θ which makes the integration straightforward. We find

$$\begin{aligned}
\lim_{R\to\infty}\left|\int_{\partial B_R}(\mathrm{d}\boldsymbol{S}\cdot\boldsymbol{E})\phi\right| &\leq \sum_{i,j}\lim_{R\to\infty}\int_{\partial B_R}\|\mathrm{d}\boldsymbol{S}\|\;|\phi_i E_j^r| \\
&\leq \sum_{i,j}\lim_{R\to\infty} 4\pi R^2\frac{|q_iq_j|}{R-r_i}\left[\sqrt{\frac{2}{\pi\sigma_j^2}}e^{-\frac{(R-r_j)^2}{2\sigma_j^2}}\frac{R+r_j}{(R-r_j)^2}+\frac{R+r_j}{(R-r_j)^3}\right] \\
&= \sum_{i,j}\lim_{R\to\infty} 4\pi|q_iq_j|\left[\sqrt{\frac{2}{\pi\sigma_j^2}}e^{-\frac{R^2}{2\sigma_j^2}}+\frac{1}{R}\right] \\
&= 0.
\end{aligned}$$

We conclude that $U^{\mathrm{elec}}\geq 0$ and if $\boldsymbol{E}\neq\boldsymbol{0}$ then $U^{\mathrm{elec}}>0$.

Appendix B

Properties of the Flat Torus $\mathbb{T}^3$

B.1 Atlas of a Flat Torus

For simplicity, we will only show how to construct an atlas for $\mathbb{T}^2 = {}^{\mathbb{R}^2}\!/_{\mathbb{Z}^2}$. The concept can be generalized to $\mathbb{T}^3$ straightforwardly. A common way to draw the two-dimensional flat torus can be seen in fig. B.1.

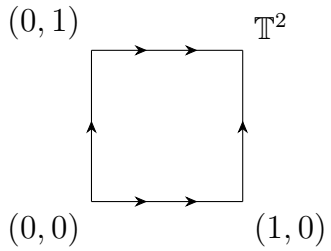


Figure B.1: A two-dimensional flat torus. Lines with the same number of arrows are identified in such a way that the orientation is respected according to the arrows.

One chart that is almost trivial to construct by defining its inverse is

$$\begin{aligned} \psi_1^{-1} : V_1 \coloneqq (0,1)^2 &\to \mathbb{T}^2 \\ \boldsymbol{x} &\mapsto [\boldsymbol{x}]. \end{aligned}$$

That this is a homeomorphism between $(0,1)^2$ and $\psi_1^{-1}((0,1)^2)$ is easy to see. Bijectivity is obvious and continuity follows from the fact that the quotient topology is precisely the topology on $\mathbb{T}^2$ that makes the canonical projection p (see eq. (3.7)) continuous. Note that ψ_1^{-1} is identical to p on $(0,1)^2$ and that this argument can also be applied for ψ_1.

This chart already covers most of $\mathbb{T}^2$. The points $[(0,y)] = [(1,y)]$ and $[(x,0)] = [(x,1)]$ are however not included. An illustration of the chart can be seen in fig. B.2.

Another chart that covers most of the points on $\mathbb{T}^2$ that ψ_1 did not cover is again defined by its inverse

$$\begin{aligned} \psi_2^{-1} : V_2 \coloneqq \left(\frac{1}{2}, \frac{3}{2}\right)^2 &\to \mathbb{T}^2 \\ \boldsymbol{x} &\mapsto [\boldsymbol{x}]. \end{aligned}$$

A sketch of this map can be seen in fig. B.3.

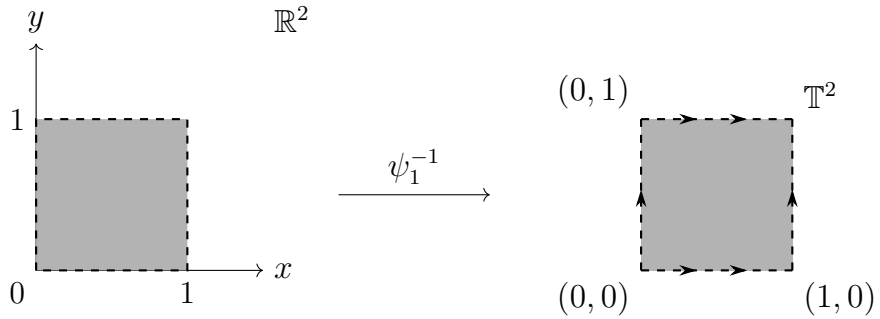


Figure B.2: Definition of the chart ψ_1 defined by its inverse $\psi_1^{-1} : (0,1)^2 \to \mathbb{T}^2$. The dashed lines bounding the open domains which are colored in gray indicate that this boundary is not part of the set anymore.

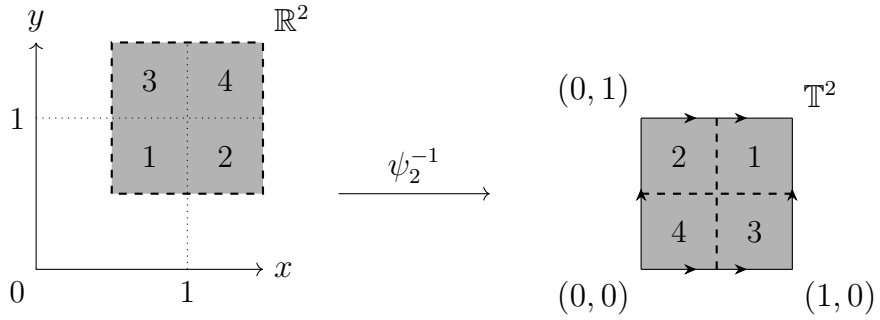


Figure B.3: Definition of the chart ψ_2 defined by its inverse $\psi_2^{-1} : (\frac{1}{2}, \frac{3}{2})^2 \to \mathbb{T}^2$. The numbers in the four quarters indicate where these regions are mapped onto the torus

Combining the charts ψ_1 and ψ_2, the only points that are still not covered on $\mathbb{T}^2$ are $[(0, \frac{1}{2})] = [(1, \frac{1}{2})]$ and $[(\frac{1}{2}, 0)] = [(\frac{1}{2}, 1)]$. Those can be covered by the charts defined via

$$\begin{aligned} \psi_3^{-1} : V_3 \coloneqq \left(\frac{1}{2}, \frac{3}{2}\right) \times (0,1) &\to \mathbb{T}^2 \\ \boldsymbol{x} &\mapsto [\boldsymbol{x}], \\ \psi_4^{-1} : V_4 \coloneqq (0,1) \times \left(\frac{1}{2}, \frac{3}{2}\right) &\to \mathbb{T}^2 \\ \boldsymbol{x} &\mapsto [\boldsymbol{x}]. \end{aligned}$$

The charts are $\mathcal{C}^\infty$-compatible on their respective common domains $U_{ij} = \psi_i^{-1}(V_i) \cap \psi_j^{-1}(V_j)$. It can be shown that this set is not connected for $i \neq j$ but has finitely many connected components. For $(i,j) = (1,2)$ the connected components and its images under the charts are drawn in fig. B.4. The connected components of the images are of the form $(a,b)\times(c,d) \subset \mathbb{R}^2,\ a,b,c,d \in \mathbb{R}$.

Let U_{ij}^k be one of the finitely many connected components of U_{ij}, then we can write more precisely, for $\boldsymbol{x} \in \psi_j(U_{ij}^k) = (a,b)\times(c,d)$, $\psi_i \circ \psi_j^{-1}(\boldsymbol{x}) = \boldsymbol{x} + \boldsymbol{n}_k \Rightarrow \psi_i \circ \psi_j^{-1} \in \mathcal{C}^\infty(\psi_j(U_{ij}^k))$. Here $\boldsymbol{n}_k \in \mathbb{Z}^2$ is in general different for every connected component U_{ij}^k. Since for each k the map $\psi_i \circ \psi_j^{-1}$ is $\mathcal{C}^\infty$ on $\psi_j(U_{ij}^k)$[1], it is $\mathcal{C}^\infty$ on $\psi_j(U_{ij})$. The argument works for all pairs

[1]Limiting procedures like checking continuity or taking the derivative can be conducted at every point $\boldsymbol{x} \in (a,b)\times(c,d)$ since we can always find an open ball $B_\varepsilon(\boldsymbol{x})$ s.t. the function $\boldsymbol{x} + \boldsymbol{n}_k$ is defined, continuous and differentiable on this ball.

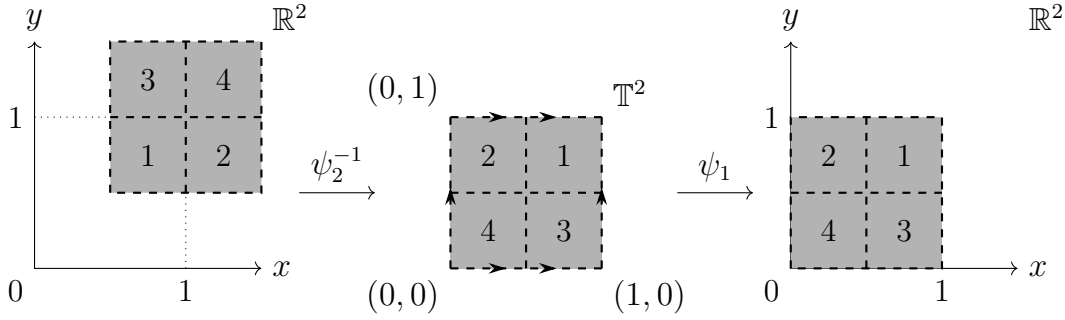


Figure B.4: Transition map $\psi_1 \circ \psi_2^{-1}$. In the middle the common domain $\psi_1^{-1}(V_1) \cap \psi_2^{-2}(V_2)$ of the charts ψ_1, ψ_2 is drawn. The domain consists of four connected components numbered from 1 to 4. They are pairwise disjoint. Left and right are the associated images of the charts on this domain.

(i, j).

Thus, we have constructed an atlas $\{(\psi_i^{-1}(V_i), \psi_i)\}$ of $\mathbb{T}^2$. Since the ψ_i are part of an atlas they are also diffeomorphisms. The analogous procedure for $\mathbb{T}^3$ leads to seven charts.

B.2 Geometry

B.2.1 Metric Tensor

The metric of $\mathbb{T}^3 = {}^{\mathbb{R}^3}\!/_{\mathbb{Z}^3}$ that is naturally induced from the canonical quotient map introduced in eq. (3.7)

$$\begin{aligned} p : \mathbb{R}^3 &\to \mathbb{T}^3 \\ \boldsymbol{x} &\mapsto [\boldsymbol{x}], \end{aligned}$$

is a *flat* metric. This means that $\mathbb{T}^3$ is locally isometric to $\mathbb{R}^3$ equipped with the Euclidean metric. We are going to show this by arguing that p is a Riemannian covering map, i.e. a smooth covering map that is also a local isometry. For this we use theorems which are mostly found in [74, 110].

Clearly, $\mathbb{R}^3$ and $\mathbb{T}^3$ are both path-connected. Furthermore, for any $[\boldsymbol{x}] \in \mathbb{T}^3$ let $U_\varepsilon = \{[\boldsymbol{y}] \in \mathbb{T}^3 |\ \boldsymbol{y} \in \mathbb{R}^3, \|\boldsymbol{y} - \boldsymbol{x}\| < \epsilon\}$ with $\varepsilon < \frac{1}{4}$ be an open connected neighborhood. Its preimage $p^{-1}(U_\varepsilon)$ clearly consists of connected components $\tilde{U}_\varepsilon^i$ which are the open balls $B_\varepsilon(\boldsymbol{x}+\boldsymbol{n}_i)$, $\boldsymbol{n}_i \in \mathbb{Z}^3$. For any $\boldsymbol{y} \in \tilde{U}_\varepsilon^i$, $\boldsymbol{y} + \boldsymbol{n} \in \tilde{U}_\varepsilon^j$ with $i \neq j$, $\boldsymbol{n} \in \mathbb{Z}^3$. Each of those connected components is mapped homeomorphically[2] onto U by p. This shows that p is a covering map.

Further, notice that $p : \mathbb{R}^3 \to \mathbb{T}^3$ is also a diffeomorphism on each $\tilde{U}_\varepsilon^i$. On $\tilde{U}_\varepsilon^i \subset \mathbb{R}^3$ take the chart $\mathrm{id}_{\tilde{U}_\varepsilon^i}$ and choose an open neighborhood $U \subset p(\tilde{U}_\varepsilon^i) \subset \mathbb{T}^3$ of $[\boldsymbol{x}]$ that is also contained in the chart ψ_i which is constructed as demonstrated in appendix B.1. Then the map $\psi_i \circ p \circ \mathrm{id}_{\tilde{U}_\varepsilon^i}^{-1} = \mathrm{id}$ on $\tilde{U}_\varepsilon^{-1} \cap p^{-1}(U)$ which is $\mathcal{C}^\infty$. This can be done for any point $[\boldsymbol{x}] \in p(\tilde{U}_\varepsilon^i)$ and so p is $\mathcal{C}^\infty$. Similarly one argues for p^{-1} which shows that p is a diffeomorphism on $\tilde{U}_\varepsilon^i$. Thus, p is a smooth covering map.

[2]Continuity is clear because the quotient topology makes p continuous. Bijectivity is clear since no two points in $\tilde{U}_\varepsilon^i$ are separated by $\boldsymbol{n}$

Next, since $p : \mathbb{R}^3 \to \mathbb{T}^3$ and $\mathbb{R}^3$ is a connected smooth manifold, p is a universal covering map which also makes it a normal smooth covering.

Let $\phi : \mathbb{R}^3 \to \mathbb{R}^3$ be a covering automorphism of p which means $p \circ \boldsymbol{\phi} = p$. Those are exactly the diffeomorphisms for which $[\phi(\boldsymbol{x})] = [\boldsymbol{x}]$. So $\phi(\boldsymbol{x}) = \boldsymbol{x} + \boldsymbol{n}$ for any $\boldsymbol{n} \in \mathbb{Z}^3$. Since the standard Euclidean metric on $\mathbb{R}^3$ is invariant under translations we can conclude that there is a unique metric on $\mathbb{T}^3$ s.t. p is a Riemannian covering. Therefore $\mathbb{T}^3$ inherits a flat metric $\tilde{g}$ that makes p a Riemannian covering[3].

Since we conduct any calculations in local coordinates we need to find the coordinate representation of $\tilde{g}$ in one of the charts ψ_i presented in appendix B.1. Since p is a local isometry it holds that the pullback of $\tilde{g}$ by p is the Euclidean metric

$$p^*\tilde{g} = \delta.$$

For two vectors $u, v \in T_q\mathbb{R}^3$ this means

$$(p^*\tilde{g})_q(u, v) = \tilde{g}_{p(q)}(\mathrm{d}p(u), \mathrm{d}p(v)) = \delta_q(u, v),$$

where the differential $\mathrm{d}p : T_q\mathbb{R}^3 \to T_{p(q)}\mathbb{T}^3$ is defined pointwise for any $f \in \mathcal{C}^\infty(\mathbb{R}^3)$ by

$$(\mathrm{d}p(u))\,(f) = u(f \circ p).$$

Let $\psi = (x^1, x^2, x^3)$ be the local coordinates defined by a particular chart ψ acting on $\psi^{-1}(V) \subset \mathbb{T}^3$. The standard coordinates on $\mathbb{R}^3$ are denoted by r^1, r^2, r^3 and it holds [110]

$$x^i = r^i \circ \psi.$$

The partial derivatives around $\tilde{q} \in \mathbb{T}^3$, $\frac{\partial}{\partial x^i}\big|_{\tilde{q}} =: \partial_i|_{\tilde{q}}$ are defined as

$$\left.\frac{\partial}{\partial x^i}\right|_{\tilde{q}} f \coloneqq \left.\frac{\partial}{\partial r^i}\right|_{\psi(\tilde{q})} (f \circ \psi^{-1}).$$

In local coordinates, the metric tensor at point $\tilde{q}$ takes the form

$$\tilde{g}_{ij}(\tilde{q}) = \tilde{g}_{\tilde{q}}(\,\partial_i|_{\tilde{q}}\,,\,\partial_i|_{\tilde{q}})$$

Let $\tilde{u}, \tilde{v} \in T_{\tilde{q}}\mathbb{T}^3$. We can "pushforward" them by ψ and write

$$\tilde{g}_{p(\psi(\tilde{q}))}(\mathrm{d}p(\mathrm{d}\psi(\tilde{u})), \mathrm{d}p(\mathrm{d}\psi(\tilde{v}))) = \delta_{\psi(\tilde{q})}(\mathrm{d}\psi(\tilde{u}), \mathrm{d}\psi(\tilde{v})).$$

The chart ψ in appendix B.1 was constructed s.t. $p|_V = \psi^{-1}$. Using this we see that for any $f \in \mathcal{C}^\infty(\mathbb{T}^3)$

$$\mathrm{d}p(\mathrm{d}\psi(\tilde{u}))(f) = \mathrm{d}\psi(\tilde{u})(f \circ p) = \tilde{u}(f \circ p \circ \psi) = \tilde{u}(f).$$

So we obtain

$$\tilde{g}_{\tilde{q}}(\tilde{u}, \tilde{v}) = \delta_{\psi(\tilde{q})}(\mathrm{d}\psi(\tilde{u}), \mathrm{d}\psi(\tilde{v})).$$

Choosing $\tilde{u} = \partial_i|_{\tilde{q}}$, $\tilde{v} = \partial_j|_{\tilde{q}}$ and computing

$$(\mathrm{d}\psi(\,\partial_i|_{\tilde{q}}))(f) = \partial_i|_{\tilde{q}}\,(f \circ \psi) = \left.\frac{\partial}{\partial r^i}\right|_{\psi(\tilde{q})} (f \circ \psi \circ \psi^{-1}) = \left.\frac{\partial}{\partial r^i}\right|_{\psi(\tilde{q})} (f),$$

one arrives at

$$\tilde{g}_{ij}(\tilde{q}) = \delta_{\psi(\tilde{q})} \left(\left.\frac{\partial}{\partial r^i}\right|_{\psi(\tilde{q})}, \left.\frac{\partial}{\partial r^j}\right|_{\psi(\tilde{q})} \right) = \delta_{ij}.$$

Thus, for local coordinates in any chart ψ that satisfies $p|_V = \psi^{-1}$ the Euclidean metric needs to be deployed.

[3] Actually, there is a freedom of choosing a metric on $\mathbb{T}^3$ but by requiring that the quotient map p is a local isometry the corresponding metric on $\mathbb{T}^3$ becomes unique which is a natural choice

B.2.2 Laplace-Beltrami Operator in Local Coordinates

On a Riemannian manifold M with metric g and local coordinates $(x^1, \dots, x^n)$ the Laplace-Beltrami operator can be written as [74]

$$\Delta f = \frac{1}{\sqrt{\det g}} \frac{\partial}{\partial x^i} \left(g^{ij} \sqrt{\det g} \, \frac{\partial f}{\partial x^j} \right).$$

Here, $f \in \mathcal{C}^\infty(M)$, $\det g = \det(g_{ij})$ and $(g^{ij}) = (g_{ij})^{-1}$ and the Einstein summation convention is used. Since the metric tensor field $\tilde{g}_{ij}$ on $\mathbb{T}^3$ in local coordinates defined by a chart constructed like in appendix B.1 is the Euclidean metric tensor field, the Laplace-Beltrami operator looks exactly like in $\mathbb{R}^3$

$$\Delta f = \sum_{i=1}^{3} \frac{\partial^2 f}{(\partial x^i)^2}.$$

Appendix C

Convergence of Electrostatics on the Torus

In section 3.4.4 we encountered the term

$$\int_{\mathbb{R}^3} \mathrm{d}^3\boldsymbol{x}\; e^{-\frac{\|\boldsymbol{x}-\boldsymbol{r}_i\|^2}{2\sigma_j^2}} \sum_{\boldsymbol{k}\neq\boldsymbol{0}} \frac{1}{\pi\|\boldsymbol{k}\|^2} e^{-2\pi^2\sigma_j^2\|\boldsymbol{k}\|^2 + i2\pi\boldsymbol{k}\cdot(\boldsymbol{x}-\boldsymbol{r}_j)},$$

where we would like to change the order of integration and summation. By simplifying the notation, the question is if one can interchange the summation with the integration of an expression of the form

$$\int_{\mathbb{R}^3} \mathrm{d}^3\boldsymbol{x} \sum_{\boldsymbol{k}\neq\boldsymbol{0}} f_{\boldsymbol{k}}(\boldsymbol{x}).$$

Using a combination of the monotone and the dominated convergence theorem for Lebesgue integrals this is a valid operation if one finds a sequence $\{g_{\boldsymbol{k}}\}_{\boldsymbol{k}\neq\boldsymbol{0}}$ for which $|f_{\boldsymbol{k}}(\boldsymbol{x})| \leq g_{\boldsymbol{k}}(\boldsymbol{x})$ almost everywhere and both

$$\sum_{\boldsymbol{k}\neq\boldsymbol{0}} g_{\boldsymbol{k}}(\boldsymbol{x}) \quad \text{and} \quad \sum_{\boldsymbol{k}\neq\boldsymbol{0}} \int_{\mathbb{R}^3} \mathrm{d}^3\boldsymbol{x}\; g_{\boldsymbol{k}}(\boldsymbol{x})$$

are finite. It can easily be checked that a suitable sequence for that is

$$g_{\boldsymbol{k}}(\boldsymbol{x}) = e^{-\frac{\|\boldsymbol{x}-\boldsymbol{r}_i\|}{2\sigma_i^2} - 2\pi^2\sigma_j^2\|\boldsymbol{k}\|^2},$$

provided the lattice sum $\sum_{k\in\mathbb{L}} g_{\boldsymbol{k}}(\boldsymbol{x})$ converges for all $\boldsymbol{x} \in \mathbb{R}^3$.

We will show that in general a sum of the form

$$\sum_{\boldsymbol{n}\in\mathbb{L}} e^{-a'\|\boldsymbol{n}+\boldsymbol{x}\|^2}$$

is convergent for any $\boldsymbol{x} \in \mathbb{R}^3$ and $a \in \mathbb{R}_{>0}$. First, note that it is enough to show that $\sum_{\boldsymbol{n}\in\mathbb{L}} e^{-a\|\boldsymbol{n}\|^2}$ is convergent since

$$\sum_{\boldsymbol{n}\in\mathbb{L}} e^{-a'\|\boldsymbol{n}+\boldsymbol{x}\|^2} < C + \sum_{\substack{\boldsymbol{n}\in\mathbb{L} \\ \|\boldsymbol{n}\|>2\|\boldsymbol{x}\|}} e^{-\frac{a'}{4}\|\boldsymbol{n}\|^2},$$

where $a' \in \mathbb{R}_{>0}$ and $C \in R_{\geq 0}$. Next, let us introduce a lemma.

Lemma C.1. *Let* $\boldsymbol{A} \in \mathbb{R}^{d\times d}$. *If* $\det\boldsymbol{A} \neq 0$, *then*

$$\mathrm{Inf}(\boldsymbol{A}) \coloneqq \inf_{\boldsymbol{x}\in S^{d-1}\subset\mathbb{R}^d} \|\boldsymbol{A}\boldsymbol{x}\| > 0,$$

where S^{d-1} *indicates the* $d-1$*-dimensional unit sphere.*

Proof. Since S^{d-1} is compact and the map

$$\begin{aligned} f\colon\ S^{d-1} \subset \mathbb{R}^d &\to \mathbb{R} \\ \boldsymbol{x} &\mapsto \|\boldsymbol{A}\boldsymbol{x}\| \end{aligned}$$

is continuous, it follows from the generalized extreme value theorem that the infimum is attained on S^{d-1}. More precisely, there exists a $\boldsymbol{q} \in S^{d-1}$ such that $\mathrm{Inf}(\boldsymbol{A}) = \|\boldsymbol{A}\boldsymbol{q}\|$. Since $\det\boldsymbol{A} \neq 0$, $\ker\boldsymbol{A} = \{\boldsymbol{0}\}$ and thus, $\|\boldsymbol{A}\boldsymbol{q}\| > 0$. □

Now use the fact that one can write the lattice vectors as $\boldsymbol{n} = \boldsymbol{A}\boldsymbol{m}$, where $\boldsymbol{m} \in \mathbb{Z}^3$ and $\boldsymbol{A} \in \mathbb{R}^{3\times3}$ is a matrix with the lattice vectors $\boldsymbol{a}_i, \boldsymbol{a}_2$ and $\boldsymbol{a}_3$ as its columns. Since those vectors are linearly independent, the determinant is non-zero. By using the previous lemma,

$$\|\boldsymbol{n}\|^2 = \|\boldsymbol{A}\boldsymbol{m}\|^2 = \left\|\boldsymbol{A}\frac{\boldsymbol{m}}{\|\boldsymbol{m}\|}\right\|^2 \|\boldsymbol{m}\|^2 \geq \mathrm{Inf}(\boldsymbol{A})^2\|\boldsymbol{m}\|^2$$

for any $\boldsymbol{m} \neq \boldsymbol{0}$. By defining $b = a\mathrm{Inf}(\boldsymbol{A})^2$ we conclude

$$\sum_{\boldsymbol{n}\in\mathbb{L}} e^{-a\|\boldsymbol{n}\|^2} \leq \sum_{\boldsymbol{n}\in\mathbb{L}} e^{-b\|\boldsymbol{m}\|^2} = \left(\sum_{n=-\infty}^{\infty} e^{-bn^2}\right)^3 = \left(-1 + 2\sum_{n=0}^{\infty} e^{-bn^2}\right)^3 < \infty.$$

The last step can easily be shown by the integral convergence test.

Appendix D

Neural Network Potential Parameters

D.1 Symmetry Functions

Table D.1: Parameters of the radial symmetry functions G^{rad} (left) and the narrow angular symmetry functions $G^{\text{ang.n.}}$ (right). In agreement with the units of the used data set, η is given in $1/{a_0}^2$, while R_s and r_c are given in a_0.

	G^{rad}					$G^{\text{ang.n.}}$						
#	T_{center}	T	η	R_s	r_c	T_{center}	T	T'	η	λ	ζ	r_c
1	H	H	0.001	0.0	12.0	H	O	H	0.010	+1	4.0	12.0
2	H	H	0.010	0.0	12.0	H	O	H	0.010	−1	4.0	12.0
3	H	H	0.030	0.0	12.0	H	O	H	0.030	+1	1.0	12.0
4	H	H	0.060	0.0	12.0	H	O	H	0.030	−1	1.0	12.0
5	H	H	0.150	1.9	12.0	H	O	H	0.070	+1	1.0	12.0
6	H	H	0.300	1.9	12.0	H	O	H	0.070	−1	1.0	12.0
7	H	H	0.600	1.9	12.0	H	O	H	0.200	+1	1.0	12.0
8	H	H	1.500	1.9	12.0	H	O	O	0.001	+1	4.0	12.0
9	H	O	0.001	0.0	12.0	H	O	O	0.001	−1	4.0	12.0
10	H	O	0.010	0.0	12.0	H	O	O	0.030	+1	1.0	12.0
11	H	O	0.030	0.0	12.0	H	O	O	0.030	−1	1.0	12.0
12	H	O	0.060	0.0	12.0	O	H	H	0.010	+1	4.0	12.0
13	H	O	0.150	0.9	12.0	O	H	H	0.010	−1	4.0	12.0
14	H	O	0.300	0.9	12.0	O	H	H	0.030	+1	1.0	12.0
15	H	O	0.600	0.9	12.0	O	H	H	0.030	−1	1.0	12.0
16	H	O	1.500	0.9	12.0	O	H	H	0.070	+1	1.0	12.0
17	O	H	0.001	0.0	12.0	O	H	H	0.070	−1	1.0	12.0
18	O	H	0.010	0.0	12.0	O	O	H	0.001	+1	4.0	12.0
19	O	H	0.030	0.0	12.0	O	O	H	0.001	−1	4.0	12.0
20	O	H	0.060	0.0	12.0	O	O	H	0.030	+1	1.0	12.0
21	O	H	0.150	0.9	12.0	O	O	H	0.030	−1	1.0	12.0
22	O	H	0.300	0.9	12.0	O	O	O	0.001	+1	4.0	12.0
23	O	H	0.600	0.9	12.0	O	O	O	0.001	−1	4.0	12.0
24	O	H	1.500	0.9	12.0	O	O	O	0.030	+1	1.0	12.0
25	O	O	0.001	0.0	12.0	O	O	O	0.030	−1	1.0	12.0

(Continuation) Parameters of the radial symmetry functions G^{rad} (left) and the narrow angular symmetry functions $G^{\text{ang.n.}}$ (right). In agreement with the units of the used data set, η is given in $1/{a_0}^2$, while R_s and r_c are given in a_0.

	G^{rad}					$G^{\text{ang.n.}}$						
#	T_{center}	T	η	R_s	r_c	T_{center}	T	T'	η	λ	ζ	r_c
26	O	O	0.010	0.0	12.0							
27	O	O	0.030	0.0	12.0							
28	O	O	0.060	0.0	12.0							
29	O	O	0.150	4.0	12.0							
30	O	O	0.300	4.0	12.0							
31	O	O	0.600	4.0	12.0							
32	O	O	1.500	4.0	12.0							

Glossary

2G Second generation of the Behler-Parrinello HDNNP vi, 1, 2, 14, 15, 18–21, 51, 57, 59, 61, 63, 66–68, 70–72, 75, 76
3G Third generation of the Behler-Parrinello HDNNP 1, 2, 19, 76
4G Fourth generation of the Behler-Parrinello HDNNP ii, vi, 2, 19, 20, 53, 55–57, 59–61, 63, 66–68, 71, 72, 75, 76

ACSF atom-centered symmetry function 16–19

BLYP Becke-Lee-Yang-Parr functional used in DFT 60

CENT charge equilibration neural network technique 19
COM center of mass 62

DFT density functional theory 1, 11, 23, 60, 91

EKF extended Kalman filter 5, 7–10, 53–56, 61, 65, 66, 75

HDNNP high-dimensional neural network potential ii, vi, 1, 2, 13, 15, 18–20, 63, 71, 72, 75, 76, 91

MD molecular dynamics ii, 1, 2, 11, 28, 29, 46, 49, 56, 57, 59, 61–63, 65, 67–69, 75, 76
MIC minimum image convention 46, 47
ML machine learning 1, 14, 16
MPI The Message Passing Interface standard used for parallel computing. It does not require shared memory. 56, 57, 75, 76

NN neural network 1–5, 7–14, 17–21, 44, 49–54, 59–61, 65, 67–69, 75, 91
NNP neural network potential 1, 2, 11–15, 18–21, 49, 53, 56, 57, 59, 61, 66–68, 70, 72, 75, 76
NPA natural population analysis 68

PBC periodic boundary condition 2, 29–31, 34, 36–39, 43, 49, 51, 61, 62, 75
PES potential energy surface 11

Qeq Charge (q) equilibration 19

ReLU rectified linear unit, a popular activation function used in NNs 14
RMSE root-mean-square error 61, 65–68, 70
RPBE revised Perdew-Burke-Enzerhof functional used in DFT 60

SF symmetry function 16, 17
SGD stochastic gradient descent 5, 9, 10

SMOW Standard Mean Ocean Water 61

VSC Vienna Scientific Cluster 62, 63, 71

wACSF weighted atom-centered symmetry function 17, 18

Bibliography

[1]W. L. Jorgensen, J. Chandrasekhar, J. D. Madura, R. W. Impey, and M. L. Klein, "Comparison of simple potential functions for simulating liquid water", The Journal of Chemical Physics **79**, 926–935 (1983).

[2]H. J. C. Berendsen, J. R. Grigera, and T. P. Straatsma, "The missing term in effective pair potentials", The Journal of Physical Chemistry **91**, 6269–6271 (1987).

[3]A. P. Bartók, M. C. Payne, R. Kondor, and G. Csányi, "Gaussian Approximation Potentials: The Accuracy of Quantum Mechanics, without the Electrons", Physical Review Letters **104**, 136403 (2010).

[4]A. P. Bartók, R. Kondor, and G. Csányi, "On representing chemical environments", Physical Review B **87**, 184115 (2013).

[5]T. Morawietz, A. Singraber, C. Dellago, and J. Behler, "How van der Waals interactions determine the unique properties of water", Proceedings of the National Academy of Sciences **113**, 8368–8373 (2016).

[6]A. Singraber, "Library-Based LAMMPS Implementation of High-Dimensional Neural Network Potentials", J. Chem. Theory Comput., 14 (2019).

[7]R. Jinnouchi, F. Karsai, and G. Kresse, "On-the-fly machine learning force field generation: Application to melting points", Physical Review B **100**, 014105 (2019).

[8]J. Behler and M. Parrinello, "Generalized Neural-Network Representation of High-Dimensional Potential-Energy Surfaces", Physical Review Letters **98**, 146401 (2007).

[9]B. G. Sumpter and D. W. Noid, "Potential energy surfaces for macromolecules. A neural network technique", Chemical Physics Letters **192**, 455–462 (1992).

[10]T. B. Blank, S. D. Brown, A. W. Calhoun, and D. J. Doren, "Neural network models of potential energy surfaces", The Journal of Chemical Physics **103**, 4129–4137 (1995).

[11]S. Lorenz, A. Groß, and M. Scheffler, "Representing high-dimensional potential-energy surfaces for reactions at surfaces by neural networks", Chemical Physics Letters **395**, 210–215 (2004).

[12]J. Behler, "Four Generations of High-Dimensional Neural Network Potentials", Chemical Reviews **121**, 10037–10072 (2021).

[13]N. Artrith, T. Morawietz, and J. Behler, "High-dimensional neural-network potentials for multicomponent systems: Applications to zinc oxide", Physical Review B **83**, 153101 (2011).

[14]T. W. Ko, J. A. Finkler, S. Goedecker, and J. Behler, "A fourth-generation high-dimensional neural network potential with accurate electrostatics including non-local charge transfer", Nature Communications **12**, 398 (2021).

[15]S. A. Ghasemi, A. Hofstetter, S. Saha, and S. Goedecker, "Interatomic potentials for ionic systems with density functional accuracy based on charge densities obtained by a neural network", Physical Review B **92**, 045131 (2015).

[16]A. Singraber and T. Morawietz, "Parallel Multistream Training of High-Dimensional Neural Network Potentials", J. Chem. Theory Comput., 18 (2019).

[17]S. Plimpton, "Fast Parallel Algorithms for Short-Range Molecular Dynamics", Journal of Computational Physics **117**, 1–19 (1995).

[18]A. P. Thompson, H. M. Aktulga, R. Berger, D. S. Bolintineanu, W. M. Brown, P. S. Crozier, P. J. in 't Veld, A. Kohlmeyer, S. G. Moore, T. D. Nguyen, R. Shan, M. J. Stevens, J. Tranchida, C. Trott, and S. J. Plimpton, "LAMMPS - a flexible simulation tool for particle-based materials modeling at the atomic, meso, and continuum scales", Computer Physics Communications **271**, 108171 (2022).

[19]M. Egmont-Petersen, D. de Ridder, and H. Handels, "Image processing with neural networks—a review", Pattern Recognition **35**, 2279–2301 (2002).

[20]G. E. Dahl, D. Yu, L. Deng, and A. Acero, "Context-Dependent Pre-Trained Deep Neural Networks for Large-Vocabulary Speech Recognition", IEEE Transactions on Audio, Speech, and Language Processing **20**, 30–42 (2012).

[21]R. Yamashita, M. Nishio, R. K. G. Do, and K. Togashi, "Convolutional neural networks: an overview and application in radiology", Insights into Imaging **9**, 611–629 (2018).

[22]J. Duarte, S. Han, P. Harris, S. Jindariani, E. Kreinar, B. Kreis, J. Ngadiuba, M. Pierini, R. Rivera, N. Tran, and Z. Wu, "Fast inference of deep neural networks in FPGAs for particle physics", Journal of Instrumentation **13**, P07027–P07027 (2018).

[23]X. Jin, S. Cai, H. Li, and G. E. Karniadakis, "NSFnets (Navier-Stokes flow nets): Physics-informed neural networks for the incompressible Navier-Stokes equations", Journal of Computational Physics **426**, 109951 (2021).

[24]G. Cybenko, "Approximation by superpositions of a sigmoidal function", Mathematics of Control, Signals and Systems **2**, 303–314 (1989).

[25]K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators", Neural Networks **2**, 359–366 (1989).

[26]M. Leshno, V. Y. Lin, A. Pinkus, and S. Schocken, "Multilayer feedforward networks with a nonpolynomial activation function can approximate any function", Neural Networks **6**, 861–867 (1993).

[27]T. M. Mitchell, *Machine Learning*, McGraw-Hill Series in Computer Science (McGraw-Hill, New York, 1997), 414 pp.

[28]I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*, Adaptive Computation and Machine Learning (The MIT Press, Cambridge, Massachusetts, 2016), 775 pp.

[29]F. Cucker and S. Smale, "On the mathematical foundations of learning", Bulletin of the American Mathematical Society **39**, 1–49 (2002).

[30]P. Petersen, M. Raslan, and F. Voigtlaender, "Topological Properties of the Set of Functions Generated by Neural Networks of Fixed Size", Foundations of Computational Mathematics **21**, 375–444 (2021).

[31]D. P. Kingma and J. Ba, *Adam: A Method for Stochastic Optimization*, (Jan. 29, 2017) preprint.

[32]D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors", Nature **323**, 533–536 (1986).

[33]R. E. Kalman, "A New Approach to Linear Filtering and Prediction Problems", Journal of Basic Engineering **82**, 35–45 (1960).

[34]G. L. Smith, S. F. Schmidt, and L. A. McGee, *Application of statistical filter theory to the optimal estimation of position and velocity on board a circumlunar vehicle*, Technical Report R-135 (National Aeronautics and Space Administration, Washington, 1962).

[35]S. J. Julier and J. K. Uhlmann, "New extension of the Kalman filter to nonlinear systems", in Signal Processing, Sensor Fusion, and Target Recognition VI, Vol. 3068 (July 28, 1997), pp. 182–193.

[36]R. J. Meinhold and N. D. Singpurwalla, "Understanding the Kalman Filter", The American Statistician **37**, 123–127 (1983).

[37]H. W. Sorenson, "Least-squares estimation: from Gauss to Kalman", IEEE Spectrum **7**, 63–68 (1970).

[38]A. Gelb, J. F. J. Kasper, R. A. N. Nash Jr., C. F. Price, and A. A. Sutherland Jr., eds., *Applied optimal estimation* (M.I.T. Press, Cambridge, Mass, 1974), 374 pp.

[39]S. S. Haykin, *Kalman filtering and neural networks* (Wiley, New York, 2001).

[40]S. Singhal and L. Wu, "Training feed-forward networks with the extended Kalman algorithm", in International Conference on Acoustics, Speech, and Signal Processing, (May 1989), 1187–1190 vol.2.

[41]Y. Iiguni, H. Sakai, and H. Tokumaru, "A real-time learning algorithm for a multilayered neural network based on the extended Kalman filter", IEEE Transactions on Signal Processing **40**, 959–966 (1992).

[42]J. Sum, C.-S. Leung, G. Young, and W.-K. Kan, "On the Kalman filtering method in neural network training and pruning", IEEE Transactions on Neural Networks **10**, 161–166 (1999).

[43]G. Puskorius and L. Feldkamp, "Multi-stream extended Kalman filter training for static and dynamic neural networks", in Computational Cybernetics and Simulation 1997 IEEE International Conference on Systems, Man, and Cybernetics, Vol. 3 (Oct. 1997), 2006–2011 vol.3.

[44]L. Feldkamp and G. Puskorius, "Training controllers for robustness: multi-stream DEKF", in Proceedings of 1994 IEEE International Conference on Neural Networks (ICNN'94), Vol. 4 (June 1994), 2377–2382 vol.4.

[45]F. Heimes, "Extended Kalman filter neural network training: experimental results and algorithm improvements", in SMC'98 Conference Proceedings. 1998 IEEE International Conference on Systems, Man, and Cybernetics (Cat. No.98CH36218), Vol. 2 (Oct. 1998), 1639–1644 vol.2.

[46]G. Puskorius and L. Feldkamp, "Decoupled extended Kalman filter training of feedforward layered networks", in IJCNN-91-Seattle International Joint Conference on Neural Networks, Vol. i (July 1991), 771–777 vol.1.

[47]V. I. Arnold, *Mathematical Methods of Classical Mechanics*, Vol. 60, Graduate Texts in Mathematics (Springer New York, New York, NY, 1989).

[48]K. Hornik, M. Stinchcombe, and H. White, "Universal approximation of an unknown mapping and its derivatives using multilayer feedforward networks", Neural Networks **3**, 551–560 (1990).

[49]E. Prodan and W. Kohn, "Nearsightedness of electronic matter", Proceedings of the National Academy of Sciences **102**, 11635–11638 (2005).

[50]W. Kohn, "Density Functional and Density Matrix Method Scaling Linearly with the Number of Atoms", Physical Review Letters **76**, 3168–3171 (1996).

[51]W. Yang, "Direct calculation of electron density in density-functional theory", Physical Review Letters **66**, 1438–1441 (1991).

[52]W. Kohn and L. J. Sham, "Self-Consistent Equations Including Exchange and Correlation Effects", Physical Review **140**, A1133–A1138 (1965).

[53]A. P. Thompson, L. P. Swiler, C. R. Trott, S. M. Foiles, and G. J. Tucker, "Spectral neighbor analysis method for automated generation of quantum-accurate interatomic potentials", Journal of Computational Physics **285**, 316–330 (2015).

[54]J. Behler, "Atom-centered symmetry functions for constructing high-dimensional neural network potentials", The Journal of Chemical Physics **134**, 074106 (2011).

[55]T. Morawietz, A. Singraber, C. Dellago, and J. Behler, "How van der Waals interactions determine the unique properties of water", Proceedings of the National Academy of Sciences **113**, 8368–8373 (2016).

[56]M. P. Bircher, A. Singraber, and C. Dellago, "Improved description of atomic environments using low-cost polynomial functions with compact support", Machine Learning: Science and Technology **2**, 035026 (2021).

[57]M. Gastegger, L. Schwiedrzik, M. Bittermann, F. Berzsenyi, and P. Marquetand, "wACSF - Weighted atom-centered symmetry functions as descriptors in machine learning potentials", The Journal of Chemical Physics **148**, 241709 (2018).

[58]N. Artrith and J. Behler, "High-dimensional neural network potentials for metal surfaces: A prototype study for copper", Physical Review B **85**, 045439 (2012).

[59]T. Morawietz, V. Sharma, and J. Behler, "A neural network potential-energy surface for the water dimer based on environment-dependent atomic energies and charges", The Journal of Chemical Physics **136**, 064103 (2012).

[60]R. S. Mulliken, "Electronic Population Analysis on LCAO–MO Molecular Wave Functions. II. Overlap Populations, Bond Orders, and Covalent Bond Energies", The Journal of Chemical Physics **23**, 1841–1846 (1955).

[61]F. L. Hirshfeld, "Bonded-atom fragments for describing molecular charge densities", Theoretica chimica acta **44**, 129–138 (1977).

[62]P. Bultinck, C. Van Alsenoy, P. W. Ayers, and R. Carbó-Dorca, "Critical analysis and extension of the Hirshfeld atoms in molecules", The Journal of Chemical Physics **126**, 144111 (2007).

[63]R. F. W. Bader, "Atoms in molecules", Accounts of Chemical Research **18**, 9–15 (1985).

[64]R. F. W. Bader, "Principle of stationary action and the definition of a proper open system", Physical Review B **49**, 13348–13356 (1994).

[65]P. Politzer and R. S. Mulliken, "Comparison of Two Atomic Charge Definitions, as Applied to the Hydrogen Fluoride Molecule", The Journal of Chemical Physics **55**, 5135–5136 (1971).

[66]P. W. Ayers, "Atoms in molecules, an axiomatic approach. I. Maximum transferability", The Journal of Chemical Physics **113**, 10886–10898 (2000).

[67]E. R. Davidson and S. Chakravorty, "A test of the Hirshfeld definition of atomic charges and moments", Theoretica chimica acta **83**, 319–330 (1992).

[68]B. Han, C. M. Isborn, and L. Shi, "Determining Partial Atomic Charges for Liquid Water: Assessing Electronic Structure and Charge Models", Journal of Chemical Theory and Computation **17**, 889–901 (2021).

[69]D. E. P. Vanpoucke, P. Bultinck, and I. Van Driessche, "Extending Hirshfeld-I to bulk and periodic materials", Journal of Computational Chemistry **34**, 405–417 (2013).

[70]K. Finzel, Á. Martín Pendás, and E. Francisco, "Efficient algorithms for Hirshfeld-I charges", The Journal of Chemical Physics **143**, 084115 (2015).

[71]A. K. Rappe and W. A. Goddard, "Charge equilibration for molecular dynamics simulations", The Journal of Physical Chemistry **95**, 3358–3363 (1991).

[72]P. T. Kiss, M. Sega, and A. Baranyai, "Efficient Handling of Gaussian Charge Distributions: An Application to Polarizable Molecular Models", Journal of Chemical Theory and Computation **10**, 5513–5519 (2014).

[73]D. Frenkel and B. Smit, *Understanding molecular simulation: from algorithms to applications*, 2. ed, Computational Science Series 1 (Academic Press, San Diego, 2002), 638 pp.

[74]J. M. Lee, *Introduction to Riemannian Manifolds*, Vol. 176, Graduate Texts in Mathematics (Springer International Publishing, Cham, 2018).

[75]M. Ruzhansky and V. Turunen, *Pseudo-differential operators and symmetries: background analysis and advanced topics*, 1st ed, Pseudo-Differential Operators. Theory and Applications vol. 2 (Birkhaeuser, Basel ; Boston, 2010), 709 pp.

[76]M. A. Pinsky, *Introduction to Fourier analysis and wavelets*, Graduate Studies in Mathematics v. 102 (American Mathematical Society, Providence, R.I, 2008), 376 pp.

[77]D. Borwein, J. M. Borwein, and K. F. Taylor, "Convergence of lattice sums and Madelung's constant", Journal of Mathematical Physics **26**, 2999–3009 (1985).

[78]A. Redlack and J. Grindlay, "Coulombic potential lattice sums", Journal of Physics and Chemistry of Solids **36**, 73–82 (1975).

[79]S. W. de Leeuw, J. W. Perram, E. R. Smith, and J. S. Rowlinson, "Simulation of electrostatic systems in periodic boundary conditions. I. Lattice sums and dielectric constants", Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences **373**, 27–56 (1980).

[80]G. Makov and M. C. Payne, "Periodic boundary conditions in ab initio calculations", Physical Review B **51**, 4014–4022 (1995).

[81]J. D. Jackson, *Classical electrodynamics*, 3rd ed (Wiley, New York, 1999), 808 pp.

[82]G. B. Arfken, H.-J. Weber, and F. E. Harris, *Mathematical methods for physicists: a comprehensive guide*, 7th ed (Elsevier, Amsterdam ; Boston, 2013), 1205 pp.

[83]R. E. Crandall and J. P. Buhler, "Elementary function expansions for Madelung constants", Journal of Physics A: Mathematical and General **20**, 5497 (1987).

[84]B. Stamm, L. Lagardère, É. Polack, Y. Maday, and J.-P. Piquemal, "A coherent derivation of the Ewald summation for arbitrary orders of multipoles: The self-terms", The Journal of Chemical Physics **149**, 124103 (2018).

[85]P. P. Ewald, "Die Berechnung optischer und elektrostatischer Gitterpotentiale", Annalen der Physik **369**, 253–287 (1921).

[86]E. R. Smith and J. S. Rowlinson, "Electrostatic energy in ionic crystals", Proceedings of the Royal Society of London. A. Mathematical and Physical Sciences **375**, 475–505 (1981).

[87]T. A. Darden, "Extensions of the Ewald method for Coulomb interactions in crystals", in *International Tables for Crystallography* (John Wiley & Sons, Ltd, Jan. 1, 2010), pp. 458–481.

[88]E. R. Smith, "Calculating the pressure in simulations using periodic boundary conditions", Journal of Statistical Physics **77**, 449–472 (1994).

[89]T. R. Gingrich and M. Wilson, "On the Ewald summation of Gaussian charges for the simulation of metallic surfaces", Chemical Physics Letters **500**, 178–183 (2010).

[90]M. Mamode, "Fundamental solution of the Laplacian on flat tori and boundary value problems for the planar Poisson equation in rectangles", Boundary Value Problems **2014**, 221 (2014).

[91]R. A. Jackson and C. R. A. Catlow, "Computer Simulation Studies of Zeolite Structure", Molecular Simulation **1**, 207–224 (1988).

[92]J. Kolafa and J. W. Perram, "Cutoff Errors in the Ewald Summation Formulae for Point Charge Systems", Molecular Simulation **9**, 351–368 (1992).

[93]H. G. Petersen, "Accuracy and efficiency of the particle mesh Ewald method", The Journal of Chemical Physics **103**, 3668–3679 (1995).

[94]B. A. Wells and A. L. Chaffee, "Ewald Summation for Molecular Simulations", Journal of Chemical Theory and Computation **11**, 3684–3695 (2015).

[95]P. P. Poier, L. Lagardère, J.-P. Piquemal, and F. Jensen, "Molecular Dynamics Using Nonvariational Polarizable Force Fields: Theory, Periodic Boundary Conditions Implementation, and Application to the Bond Capacity Model", Journal of Chemical Theory and Computation **15**, 6213–6224 (2019).

[96]N. C. Handy and H. F. Schaefer, "On the evaluation of analytic energy derivatives for correlated wave functions", The Journal of Chemical Physics **81**, 5031–5033 (1984).

[97]G. Imbalzano, A. Anelli, D. Giofré, S. Klees, J. Behler, and M. Ceriotti, "Automatic selection of atomic fingerprints and reference configurations for machine-learning potentials", The Journal of Chemical Physics **148**, 241730 (2018).

[98]T. Morawietz and J. Behler, *HDNNP training data set for H2O* (Zenodo, Apr. 9, 2019).

[99]B. Hammer, L. B. Hansen, and J. K. Nørskov, "Improved adsorption energetics within density-functional theory using revised Perdew-Burke-Ernzerhof functionals", Physical Review B **59**, 7413–7421 (1999).

[100]S. Grimme, J. Antony, S. Ehrlich, and H. Krieg, "A consistent and accurate ab initio parametrization of density functional dispersion correction (DFT-D) for the 94 elements H-Pu", The Journal of Chemical Physics **132**, 154104 (2010).

[101]B. Cordero, V. Gómez, A. E. Platero-Prats, M. Revés, J. Echeverría, E. Cremades, F. Barragán, and S. Alvarez, "Covalent radii revisited", Dalton Transactions, 2832–2838 (2008).

[102]M. Tanaka, G. Girard, R. Davis, A. Peuto, and N. Bignell, "Recommended table for the density of water between 0 C and 40 C based on recent experimental reports", Metrologia **38**, 301–309 (2001).

[103]*Materials Data on H2O by Materials Project*, in collab. with T. M. Project (LBNL Materials Project; Lawrence Berkeley National Laboratory (LBNL), Berkeley, CA (United States), 2020).

[104]A. Jain, S. P. Ong, G. Hautier, W. Chen, W. D. Richards, S. Dacek, S. Cholia, D. Gunter, D. Skinner, G. Ceder, and K. A. Persson, "Commentary: The Materials Project: A materials genome approach to accelerating materials innovation", APL Materials **1**, 011002 (2013).

[105]C. M. Breneman and K. B. Wiberg, "Determining atom-centered monopoles from molecular electrostatic potentials. The need for high sampling density in formamide conformational analysis", Journal of Computational Chemistry **11**, 361–373 (1990).

[106]A. E. Reed, R. B. Weinstock, and F. Weinhold, "Natural population analysis", The Journal of Chemical Physics **83**, 735–746 (1985).

[107]Y. S. Badyal, M.-L. Saboungi, D. L. Price, S. D. Shastri, D. R. Haeffner, and A. K. Soper, "Electron distribution in water", The Journal of Chemical Physics **112**, 9206–9208 (2000).

[108]A. V. Gubskaya and P. G. Kusalik, "The total molecular dipole moment for liquid water", The Journal of Chemical Physics **117**, 5290–5302 (2002).

[109]E. W. Ng and M. Geller, "A table of integrals of the Error functions", Journal of Research of the National Bureau of Standards, Section B: Mathematical Sciences **73B**, 1 (1969).

[110]L. W. Tu, *An Introduction to Manifolds*, Universitext (Springer New York, New York, NY, 2011).