Johannes J. Cartus, BSc

A Neural Network Based Approach to SCF Initial Guesses

MASTER'S THESIS

to achieve the university degree of

Diplom-Ingenieur

Master's degree programme: Technical Physics

submitted to

Graz University of Technology

Supervisor

Ass.Prof. Mag.phil. Dipl.-Ing. Dr.phil. Dr. techn. Andreas W. Hauser

Institute of Experimental Physics

Graz, March 2019

AFFIDAVIT

I declare that I have authored this thesis independently, that I have not used other than the declared sources/resources, and that I have explicitly indicated all material which has been quoted either literally or by content from the sources used. The text document uploaded to TUGRAZonline is identical to the present master‘s thesis.

Date

Signature

Kurzfassung

Algorithmen, die selbstkonsistent arbeiten, wie z.B. Hartree-Fock, ein Standardverfahren in der Elektronenstrukturtheorie, zeigen naturgemäß eine starke Abhängigkeit von der Qualität der Startlösung. Dies umso mehr, je größer und komplexer das untersuchte System ist. Ist die Startlösung jedoch unzureichend, sind nicht nur ein langwieriger Konvergenzprozess, sondern im schlimmsten Fall sogar ein völliger Abbruch der Berechnung zu erwarten. Obwohl in den letzten Jahrzehnten große Fortschritte bei der Verbesserung der Algorithmen selbst gemacht wurden, ist das Generieren einer guten Startlösung bisher mit vergleichsweise wenig Aufmerksamkeit bedacht worden. Viele der gängigen Schemata sind sehr allgemein konstruiert und oft für bestimme Systeme nicht anwendbar.

Ein Beispiel sind Hückel-artige Verfahren, welche sich hervorragend für organische Moleküle eignen, aber für metallische Systeme im Vergleich zu Methoden wie der „superposition of atomic densities“, in der sphärisch gemittelte atomare Dichten zu einer molekularen Startlösung kombiniert werden, deutliche Nachteile zeigen. Hier setzt die vorliegende Arbeit an, welche einen Machine-Learning-Ansatz ins Spiel bringt und untersucht, ob sich brauchbare Startlösungen mit künstlichen neuronalen Netzen generieren lassen. Ausgehend von einer Beziehung zwischen orbitalem Überlapp und konvergierter Elektronendichte, wird der Ansatz in späteren Kapiteln weiter verfeinert.

Abstract

As much fine tuning as the initial guess and the self consistent field (SCF) iteration algorithms of Hartree-Fock implementations have seen in the last decades, in many cases the convergence for large systems is still problematic, and automatic black box-type initial guesses often apply only to a limited selection of scenarios. An inappropriate guess can lead to a very slow and therefore inefficient convergence behavior. Many of the schemes currently in use are designed in a very general way, and may therefore not be applicable to certain systems.

An example are Hückel-based methods, which are very effective for organic molecules, but fail for metallic systems. The latter are preferably treated via the conceptually simple superposition of atomic densities, where the latter are spherically averaged to yield a trial density matrix. This is the starting point for the machine learning ansatz proposed in this thesis. The applicability of a neural network-based approach is investigated with the aim to find fast converging initial guesses for the SCF algorithm. The thesis starts with a very simple first approach, based on relations between atomic orbital overlap and the electron density distribution, which is finally improved towards a more generalized solution in later chapters.

Acknowledgements

I want to express my gratitude to Prof. Wolfgang E. Ernst and the Institute of Experimental Physics for making this thesis possible and giving me the opportunity to work in such a vibrant and stimulating environment. I also want to thank the FWF Austria who funded this thesis through the "Heterogeneous catalysis on metallic nanoparticles" project.

My deepest appreciation goes to Prof. Andreas W. Hauser and to Ralf Mayer, without whom this thesis would have never come into existence. Thank you for supervising, guiding, and both encouraging and restraining me every time I got stuck or wandered off course. I also want to thank the rest of our team, Alex Fuchs, Klemens Schmuck, and Mario Poschner, who supported me with great ideas and insights and made this undertaking an even more joyful experience.

Furthermore, I am very greateful to my family and friends who always supported me unconditionally in all my endavours and to whom I owe everything. In particular, I want to thank Doris Grössl who continued to motivate and push me even though this thesis was probably almost as demanding on her as it was on me.

Contents

1. Introduction

This thesis is conducted to mitigate a problem very common in the machine learning community, not just in quantum chemistry. All models, regardless of their complexity, require fitting data. This data needs to be as representative as possible to yield the best results with the fewest examples. Unfortunately, obtaining data is often difficult in quantum chemistry due to one or both of the following reasons:

- The samples are computationally too expensive to generate. In most cases, this is the main reason why machine learning is needed in the first place.
- Data sampling without introducing a bias is not straight forward and often not possible. Usually the samples correspond to points in very high-dimensional spaces, and this dimensionality makes it computationally impossible to exhaustively sample these spaces e.g. by working along a grid.

In some cases the two problems from above combine. If grid sampling or other deterministic methods are no longer applicable, probabilistic methods (such as Monte Carlo or molecular dynamics[1]) are typically used in order to restrict the sampling to "more relevant" areas of space. This bias, while saving computation time, has of course an impact on the model performance and will cause the model to fail if it is confronted with a sample outside of the sampled regions. However, reducing this bias involves more than just adjusting the sampling algorithm. After a point in the space to sample is selected further calculations (e.g. the application of the SCF-technique in order to obtain properties of interest) are often necessary, and the methods to carry them out might face additional difficulties in certain areas of the space (e.g. far away from equilibrium).

To remedy these difficulties, many ideas have been proposed to make the SCF process more robust and to speed it up. An SCF calculation starts from an initial guess for the electronic density and then improves this guess in an iterative fashion until it either reaches the final density or has to admit defeat. One could compare it to a traveller, trying to get from one city to another as fast as possible. The straight forward way to do this would be to take the shortest possible way and avoid any detours. This corresponds to improvements in the iteration process. A lot of research has been conducted in this area with great results[2]. However, the travel of our hiker would also be a lot shorter if the towns were fairly close in the first place. For the SCF calculation, this would mean a better starting solution. A good initial guess to an SCF calculation definitely reduces the probability of the calculation to fail. To our surprise, not much attention has been paid to this aspect recently. The latest break-through was the very successful superposition of atomic densities approach (Section 2.1.9), published in 2006[2]. In this work, a few

[1] See e.g. [1, p. 22 *et seqq.*] for molecular dynamics and [1, p. 448 *et seqq.*] for Mote Carlo methods.
[2] See e.g. the methods described in Section 2.1.10 or [1, p. 101 *et seqq.*] for more details.

new schemes based on machine learning and artificial neural networks in particular are developed and tested. This way, we hope to leverage a priori available information (in the form of the densities of easily converged samples) together with improved schemes to produce more robust and reliable initial guesses. Since the aim is to reduce computation time, these initial guess schemes need to be as "lightweight" as possible while still providing suitable accuracy. As a first test, we investigate whether the density matrix of a molecule can be guessed using the overlap matrix as input. After that, we attempt to generalize this result. This generalization is with respect to the input as well as the output side of the guess algorithm. So far, to our knowledge, there is no scheme that captures enough information on an atomic environment to estimate non-scalar quantities. Since we need just that, we will discuss ways to create inputs that have this capability. In addition, we will derive a few schemes to generate guesses for the density matrix based on these inputs, and apply the result to a few testing datasets.

Before going into the details a few words shall be said about the actual software that this project is executed with.

- Back End: all of the back end used to set up testing and analysis methods as well as data set generation and processing functionality was done with the Python[3] language. In the course of this, all numerically demanding steps were carried out using SciPy[3] and NumPy[4–6] and all plots were generated with Matplotlib[7], an open-source plotting library for python, most of them using the seaborn front end[4].
- Data Generation: most data generated to be used in this work was sampled using molecular dynamics. The initial geometries were created using Avogadro[8, 9]. The molecular dynamics trajectories were calculated with the QChem[10] computational chemistry package.
- SCF Calculations: the Hartree-Fock calculations to measure the number of iterations required to converge an initial guess were carried out using PySCF[11]. Also, some of the initial guess schemes (H_{core} and SAD, see Section 2.1.9) were used as implemented in the package.
- Neural Networks: parts of this thesis related to neural network approaches employ the library Keras[12] in a Tensorflow[13] based implementation.

[3] `https://www.python.org/`
[4] `https://seaborn.pydata.org/`

2. Fundamentals

2.1. Self Consistent Field Calculations

2.1.1. Introduction

The aim of this thesis is to ultimately make the calculation of molecular properties easier. Therefore we will take a closer look at the problem at hand and introduce a method to solve it. The introduction given here follows to a great extend the book of A. Szabo and N. S. Ostlund[1]. Before going into details, a few words must be spent on units. The problem that is outlined below can be simplified by using atomic units (Tab. 2.1).

Table 2.1.: Atomic units and their values in SI units [14, p. 42].

Physical Quantity	**Denotation**	**Value in SI**
Length	a_0	$5.2918 \times 10^{-11}\,\mathrm{m}$
Mass	m_e	$9.1095 \times 10^{-31}\,\mathrm{kg}$
Charge	e	$1.6022 \times 10^{-19}\,\mathrm{C}$
Energy	E_h	$4.3598 \times 10^{-18}\,\mathrm{J}$
Angular Momentum	$\hbar$	$1.0546 \times 10^{-34}\,\mathrm{J\,s}$
Electric Dipole Moment	$e\,{a_0}$	$8.4784 \times 10^{-30}\,\mathrm{C\,m}$
Electric Polarizability	$e^2\,{a_0}^2\,{E_\mathrm{h}}^{-1}$	$1.6488 \times 10^{-41}\,\mathrm{C^2\,m^2\,J^{-1}}$
Electric Field	$E_\mathrm{h}\,e^{-1}\,{a_0}^{-1}$	$5.1423 \times 10^{11}\,\mathrm{V\,m^{-1}}$
Wave Function	${a_0}^{-3/2}$	$2.5978 \times 10^{15}\,\mathrm{m^{-3/2}}$

By expressing physical quantities in atomic units most constants are set to 1, allowing us to omit them and thus saving us time and energy. If not explicitly stated otherwise, these units are used throughout the whole thesis.

In order to calculate basic molecular properties, the non-relativistic time-independent Schrödinger equation

$$\boxed{H\,|\Phi\rangle = E\,|\Phi\rangle} \tag{2.1}$$

has to be solved, with the many-body Hamilton operator (in atomic units)

$$H = -\frac{1}{2}\sum_{i=1}^{N}\nabla_i^2 - \frac{1}{2}\sum_{A=1}^{M}\frac{1}{M_A}\nabla_A^2 - \sum_{i=1}^{N}\sum_{A=1}^{M}\frac{Z_A}{r_{i,A}} + \sum_{i=1}^{N}\sum_{j>i}^{N}\frac{1}{r_{ij}} + \sum_{A=1}^{N}\sum_{B>A}^{N}\frac{Z_A Z_B}{R_{AB}}. \tag{2.2}$$

[1] Most of Chapters 2 and 3[14, p. 39-152].

describing a molecule with N electrons and M nuclei. This poses a very complicated problem that can only be solved by making a few approximations.

2.1.2. Born-Oppenheimer Approximation

The first approximation is a very powerful one. With the mass of the electrons being much smaller[2] than the mass of any of the nuclei, we can treat the electronic motion as decoupled from the nuclear motion. In practice this means that equation (2.1) is solved for fixed nuclear positions, which yields the Hamiltonian

$$H = -\frac{1}{2}\sum_{i=1}^{N}\nabla_i^2 - \sum_{i=1}^{N}\sum_{A=1}^{M}\frac{Z_A}{r_{i,A}} + \sum_{i=1}^{N}\sum_{j>i}^{N}\frac{1}{r_{ij}}. \tag{2.3}$$

Details of how this is achieved can be found in Appendix A.1.

2.1.3. Electronic Wave Functions

The problem above is still very complicated. We will try to simplify it even further by expressing our solution, the many-electron wave function $|\Phi\rangle$ as a combination of one-electron wave functions $|\phi\rangle$, which we will call orbitals. The many-electron wave function must fullfil the Pauli exclusion principle and thus be antisymmetric in the electronic coordinates. A way to realize this is to use Slater-determinants, where all combinations of electrons and orbitals are arranged in a matrix of which the determinant is taken to obtain an antisymmetrized N-electron wave function.

$$\Psi(\boldsymbol{r}_1, \boldsymbol{r}_2, \ldots) = \frac{1}{\sqrt{N!}}\begin{vmatrix} \psi_1(\boldsymbol{r}_1) & \psi_1(\boldsymbol{r}_2) & \ldots & \psi_1(\boldsymbol{r}_N) \\ \psi_2(\boldsymbol{r}_1) & \psi_2(\boldsymbol{r}_2) & \ldots & \psi_2(\boldsymbol{r}_N) \\ \vdots & & \ddots & \\ \psi_N(\boldsymbol{r}_1) & \psi_N(\boldsymbol{r}_2) & \ldots & \psi_N(\boldsymbol{r}_N) \end{vmatrix} \tag{2.4}$$

Throughout this thesis, a short notation for kets corresponding to determinants such as the one above will be used, writing it as $|\psi_1 \ldots \psi_N\rangle$ or even shorter as $|1 \ldots N\rangle$.

2.1.4. Variational Method and Hartree-Fock Equations

With this many-electron wave function we can now begin our search for the ground state energy

$$E_0 = \langle\Psi_0|H|\Psi_0\rangle\,. \tag{2.5}$$

We apply the variational method to find the orbitals that yield the slater determinant with the lowest energy.

$$E_0 = \min_{\{\psi_i\}}\langle\Psi|H|\Psi\rangle \tag{2.6}$$

[2] The ratio of the mass of a single proton to that of an electron is approx. $m_\mathrm{p}/m_\mathrm{e} \approx 1838$.

This is discussed in detail in Appendix A.3. As is demonstrated, the orbitals that yield the lowest energy E_0 are also the eigenvalues of the Fock operator, an effective one-electron operator occurring in the Hartree-Fock equations

$$\boxed{F\left|\psi_i\right\rangle = \varepsilon_i \left|\psi_i\right\rangle} \tag{2.7}$$

or in more detail

$$\left[h(1) + \sum_{j\neq i} J_j(1) - \sum_{j\neq i} K_j(1)\right]\psi_i(1) = \varepsilon_i\psi_i(1), \tag{2.8}$$

with J denoting the Coulomb operator,

$$J_b(1)\psi_a(1) := \left[\int \psi_b^*(2)\frac{1}{r_{12}}\psi_b(2)\mathrm{d}\boldsymbol{x}_2\right]\psi_a(1), \tag{2.9}$$

K as the Exchange operator,

$$K_b(1)\psi_a(1) := \left[\int \psi_b^*(2)\frac{1}{r_{12}}\psi_a(2)\mathrm{d}\boldsymbol{x}_2\right]\psi_b(1), \tag{2.10}$$

and h as the one-electron Hamiltonian (also called core Hamiltonian)

$$h(i) = -\frac{1}{2}\nabla_i^2 - \sum_A^M \frac{Z_A}{r_{iA}} \tag{2.11}$$

Their sum (the term in brackets in (2.8)) yields the Fock operator

$$F(1) = h(1) + \sum_{j\neq i} J_j(1) - \sum_{j\neq i} K_j(1) \tag{2.12}$$

The numbers in the parentheses express on which electron in the system they act or belong to (e.g. $\psi(1) = \psi(\boldsymbol{x}_1)$ has the first electron's coordinates $\boldsymbol{x}_1$ as argument).

This is a remarkable result. The sums in (2.8) can be seen as averaging the influence of all electrons in the system on the i-th electron[3]. This can be interpreted as an electron moving in the mean-field produced by all other electrons. This allows us to work with one-electron operators while still maintaining electron-electron interaction.

The eigenvalues of the Fock operator can be interpreted as orbital energies. This is e.g. used to derive Koopmans' theorem[15]. The total energy can be calculated directly from the orbitals via

$$E_0 = \sum_a \langle a|h|a\rangle + \frac{1}{2}\sum_{ab} \langle ab||ab\rangle\,, \tag{2.13}$$

as is shown in Appendix A.2. The energy in this expression is of course not exact and will lie above the actual ground state energy as it was derived using a variational method. We will refer to it as Hartree-Fock energy (HF energy).

[3] $i \neq j$ avoids self-interaction. This could be skipped here, since $J_j(1)\psi_i = K_j(1)\psi_i$ for $i = j$, causing the problematic terms to cancel each other.

2.1.5. Spin and Closed-Shell Restricted Hartree-Fock

To be able to execute concrete computations we must find a way to represent our orbitals. However, before we do this we must briefly talk about spin. A suitable description electrons naturally requires a spin *and* a spatial component in the orbitals. This means our orbitals from Section 2.1.3 are actually product states of a space $\left|\tilde{\psi}\right\rangle$ and a spin state $|\chi\rangle$: $|\psi_i\rangle = \left|\tilde{\psi}_i\right\rangle \cdot |\chi_i\rangle$. For the sake of simplicity, we will restrict ourselves in this derivation to "restricted" Hartree-Fock, which means that for the rest of this introduction we assume that all spins in our system are paired (closed-shell) and our orbitals are of the form

$$|\psi_i\rangle = \begin{cases} \left|\tilde{\psi}_j\right\rangle \cdot |\chi_i\rangle \Big|_{\chi_i=\uparrow} & \text{if } i \text{ is odd} \\ \left|\tilde{\psi}_j\right\rangle \cdot |\chi_i\rangle \Big|_{\chi_i=\downarrow} & \text{if } i \text{ is even} \end{cases}, \tag{2.14}$$

where j is related to i via $j = \text{floor}(\frac{i+1}{2}))$[4]. This means that a pair of spin functions $\{\uparrow, \downarrow\}$ shares a common spatial component. The fact that all spins are paired forces us to stick to systems with an even number of electrons[5], but it also allows us to integrate the spin part out of the Hartree-Fock equations. The result is

$$\left[h(1) + 2\sum_{j=1, j\neq i}^{\frac{N}{2}} J_j(1) - \sum_{j=1, j\neq i}^{\frac{N}{2}} K_j(1)\right] \tilde{\psi}_i(1) = \varepsilon_i \tilde{\psi}_i(1), \qquad i \in \left\{1, \ldots, \frac{N}{2}\right\} \tag{2.15}$$

which is very similar to (2.8). Since the spatial part of two orbitals at a time is identical, the sums have reduced to twice the sum over the available spatial orbitals. This corresponds to cases of combinations of parallel and opposite spin. The latter vanishes for the exchange term, reflecting that exchange interaction only takes place between electrons of equal spin. This can also bee seen when investigating the HF energy, which becomes

$$E_0 = 2\sum_i h_{ii} + \sum_{ij} (2J_{ij} - K_{ij}) \tag{2.16}$$

with the Coulomb integral J_{ij}, the Exchange integral K_{ij} and h_{ij} as defined in Appendix A.2.

We will continue with our closed shell approximation for the rest of this chapter and shall thus drop the tilde from our spatial orbitals $\tilde{\psi} \to \psi$, and denote them as we used to denote the product states before. Thus, ψ shall always denote the spatial component if not states otherwise.

2.1.6. The Roothaan-Equations

Above the necessity to find a (computerreadable) representation of our now merely spatial orbitals was mentioned. We will do this by introducing a set basis of functions and

[4] floor maps a real number to the largest integer that is lower than itself, i.e., $\text{floor}(x) := \max\{k \in \mathbb{Z} : k \leq x\}$ for $x \in \mathbb{R}$.

[5] For other cases see e.g. Szabo and Ostlund[14, p. 205 *et seqq.*], where they are discussed in detail.

expanding our orbitals in this basis. As computers are finite instruments, we will have to make due with a finite basis of K elements:

$$\psi_i \approx \sum_\nu C_{\nu i}\phi_\nu, \quad \nu \in \{1,\dots,K\} \tag{2.17}$$

The fact that our basis is finite and thus not complete introduces an error which is discussed in Section 2.2.4. Nevertheless, by introducing a basis, (2.7) will simplify ultimately to a set of algebraic equations, which is much easier to solve numerically. First, we will insert (2.17) into (2.15), multiply by ϕ_μ^* and integrate to obtain:

$$\sum_\nu C_{\nu_i} \int \phi_\mu^*(1)F(1)\phi_\nu(1)\,\mathrm{d}\boldsymbol{r}_1 = \varepsilon_i \sum_\nu C_{\nu i} \int \phi_\mu^*(1)\phi_\nu(1)\,\mathrm{d}\boldsymbol{r}_1 \tag{2.18}$$

By defining

$$F_{\mu\nu} := \int \phi_\mu^*(1)F(1)\phi_\nu(1)\,\mathrm{d}\boldsymbol{r}_1 \qquad \text{(Fock matrix)} \tag{2.19}$$

$$S_{\mu\nu} := \int \phi_\mu^*(1)\phi_\nu(1)\,\mathrm{d}\boldsymbol{r}_1 \qquad \text{(Overlap matrix)} \tag{2.20}$$

we can rewrite the equations above to receive

$$\sum_\nu F_{\mu\nu}C_{\nu i} = \varepsilon_i \sum_\nu S_{\mu\nu}C_{\nu i}, \quad i \in \{1,\dots,K\} \tag{2.21}$$

or, written as matrix equation,

$$\boldsymbol{FC} = \boldsymbol{SC\varepsilon}. \tag{2.22}$$

Here $\boldsymbol{\varepsilon}$ denotes a matrix with $\varepsilon_i,\ i \in \{1,\dots,K\}$ in the diagonal elements and the other elements being zero. This is an incredible step forward. Not only does this matrix form allow us to store representations of integro-differential operators but matrix equations can also be efficiently processed on the computer.

In (2.22) we found a generalized eigenvalue problem. We will transform it into a simple eigenvalue problem by constructing a matrix $\boldsymbol{X}$, such that

$$\boldsymbol{X}^\dagger \boldsymbol{S} \boldsymbol{X} \stackrel{!}{=} \boldsymbol{1} \tag{2.23}$$

and use orbitals

$$\phi_i' = \sum_j X_{ji}\phi_j, \quad \forall i \in \{1,\dots,K\}. \tag{2.24}$$

While there are many ways to construct such a matrix we will use

$$\boldsymbol{X} = \boldsymbol{U}\boldsymbol{s}, \tag{2.25}$$

where $\boldsymbol{s}$ is the diagonalized version of the Overlap matrix $\boldsymbol{S}$ and $\boldsymbol{U}$ is the diagonalizing matrix, i.e. $\boldsymbol{S} = \boldsymbol{U}\boldsymbol{s}\boldsymbol{U}^\dagger$. This is called canonical orthogonalization. It is preferred over symmetric orthogonalization (using $\boldsymbol{S}^{1/2}$ directly as $\boldsymbol{X}$) because it allows a reduction of dimension if the basis set shows linear dependence[6].

[6] More about this can be found in the book of Szabo and Ostlund [14, p. 142 *et seqq.*].

2.1.7. The (Charge) Density Matrix

We will now investigate if we can find a simpler way to calculate $\boldsymbol{F}$. We define the charge-density matrix

$$P_{\mu\nu} = 2\sum_{i}^{\frac{N}{2}} C_{\mu i} C_{\nu i}^{*} \tag{2.26}$$

which allows us to express the probability to find an electron at the position $\boldsymbol{r}$ in our system in the following way:

$$\mathrm{P}(\boldsymbol{r}) = \sum_{\mu\nu} P_{\mu\nu}\phi_{\mu}(\boldsymbol{r})\phi_{\nu}^{*}(\boldsymbol{r}) \tag{2.27}$$

as is shown in Appendix A.4. We can use the charge-density matrix to rewrite the Fock operator,

$$F_{\mu\nu} = H_{\mu\nu}^{\mathrm{core}} + G_{\mu\nu} \tag{2.28}$$

where $H_{\mu\nu}^{\mathrm{core}}$ denotes the elements of the one-electron hamiltonian h as defined in (2.11),

$$H_{\mu\nu}^{\mathrm{core}} = \int \phi_{\mu}^{*}(1) h(1) \phi_{\nu}(1)\, \mathrm{d}\boldsymbol{r}_1 \tag{2.29}$$

and $G_{\mu\nu}$ contains all two-electron parts:

$$G_{\mu\nu} = P_{\mu\nu}\left[(\mu\nu|\sigma\lambda) - \frac{1}{2}(\mu\lambda|\sigma\nu)\right]. \tag{2.30}$$

We used the notation defined in Appendix A.2. The details of the last step are shown in Appendix A.5. What was gained in this step? The advantages are that the core Hamiltonian and the charge-density appear explicitly, allowing us to calculate them separately. More importantly, the core Hamiltonian and two-center integrals of the basis functions $(\mu\nu|\sigma\lambda)$ may be calculated at the beginning of the computation, as they remain the same throughout the SCF algorithm. This can save a lot of unnecessary work.

Before moving on, we will point out a few properties of the density matrix which can be used later to validate our results.

- Symmetry: as the density matrix is a matrix representation of a hermitian operator, the matrix is hermitian too.
- Idempotence: the density matrix can easily be shown to be idempotent with respect to the overlap matrix:

$$\boldsymbol{PSP} = \boldsymbol{C}\underbrace{\boldsymbol{C}^{\dagger}\boldsymbol{S}\boldsymbol{C}}_{(\langle\psi_i|\psi_j\rangle)=\mathbf{1}}\boldsymbol{C}^{\dagger} = \boldsymbol{C}\boldsymbol{C}^{\dagger} = \boldsymbol{P}. \tag{2.31}$$

- Its trace: Mullikan Population analysis[16] is a very practical tool to get a rough understanding of the charge distribution in the system. The idea is that the number of electrons at atom A can be calculated by

$$N_A = \sum_{i\in I}(\boldsymbol{PS})_{ii}, \tag{2.32}$$

where I denotes the subset of indices that label basis functions associated with atom A (more on that in Section 2.2). The charge at this atom is now the difference of the atomic number and the number of electrons. Furthermore, we denote that for the total number of electrons in the system we can write:

$$N = \mathrm{tr}\,(\boldsymbol{PS}). \tag{2.33}$$

2.1.8. The Hartree-Fock SCF Procedure

Now we have all components together to automatically calculate our approximation to the ground state energy for an arbitrary molecule. The procedure is summarized below in Algorithm 1.

Algorithm 1: The Hartree-Fock procedure to calculate an approximation of the ground state energy of a given molecule in a given basis set.

Data: The molecule (i.e. the nuclear coordinates R_a and the basis set.)
begin
  Calculate $S_{\mu\nu}$, $H_{\mu\nu}^{\mathrm{core}}$ and $(\mu\nu|\sigma\lambda)$;
  Diagonalize $\boldsymbol{S}$ to obtain $\boldsymbol{U}$ and $\boldsymbol{s}$ to calculate $\boldsymbol{X}$;
  Create an initial guess for $\boldsymbol{P}$;
  while *energy not converged* **do**
    Calculate $\boldsymbol{G}$ from $\boldsymbol{P}$ and $(\mu\nu|\sigma\lambda)$;
    Calculate $\boldsymbol{F}$ from $\boldsymbol{H}^{\mathrm{core}}$ and $\boldsymbol{G}$;
    Transform the generalized eigenvalue problem to an ordinary eigenvalue problem by transforming $\boldsymbol{F} \rightarrow \boldsymbol{X}^{\dagger}\boldsymbol{F}\boldsymbol{X} := \boldsymbol{F'}$;
    Diagonalize $\boldsymbol{F'}$, i.e. find $\boldsymbol{C'}$, so that $\boldsymbol{F'} = \boldsymbol{C'}\boldsymbol{\varepsilon}\boldsymbol{C'}^{\dagger}$;
    Transform back: calculate $\boldsymbol{C} = \boldsymbol{X}\boldsymbol{C'}$;
    Recalculate $\boldsymbol{P}$ with the new $\boldsymbol{C}$;
    Calculate the energy from $\boldsymbol{P}$;
  end
  Result: The orbital occupations in form of $\boldsymbol{C}$ and perhaps the converged energy.
end

2.1.9. Initial Guesses

In Algorithm 1 an initial guess for the charge density matrix was mentioned. Though seeming to be just a small detail, it can have a major impact on convergence speed. The quality of the initial guess can also make the difference between convergence and divergence. With "initial guesses" being the main topic of this thesis, we will take a closer look at some of the schemes currently in use.

One Electron Hamiltonian

The probably simplest and cheapest initial guess is to simply diagonalize the core hamiltonian $\boldsymbol{H}^{\text{core}}$ to obtain a density that completely neglects electron interaction. However, it already incorporates some information about the system (e.g. charge and position of the nuclei) and is very easily implemented. Furthermore, it does not require any additional computation as $\boldsymbol{H}^{\text{core}}$ is required for the SCF algorithm anyway. However, as we will see later, the fact that electron-electron interaction is neglected entirely means that there is still a long way to go to obtain meaningful results from the SCF calculation, requiring many iterations to converge, and often showing "oscillating" behavior around the final energy. This guess scheme shall be denoted as "H_{core}" throughout this thesis.

Generalized Wolfsberg-Helmholtz Ansatz

The generalized Wolfsberg-Helmholtz guess (GWH)[17], sometimes also referred to as extended Hückel guess after the work of R. Hoffman[18], is a semi-empirical method to first set up a trial Fock matrix, which is then diagonalized to obtain a guess for the density of the molecule. The scheme also starts with the core Hamiltonian, but only uses its diagonal elements. All off-diagonal elements are interpolated by combining the corresponding diagonal elements and weighting the result with the overlap matrix $\boldsymbol{S}$,

$$F_{\mu\nu} = KS_{\mu\nu}\frac{H^{\text{core}}_{\mu\mu} + H^{\text{core}}_{\nu\nu}}{2}, \tag{2.34}$$

where K is a small dimensionless constant very often chosen to be 1.75. This scheme is a drastic improvement to simply using the one-electron Hamiltonian as electron-electron interaction is not neglected completely anymore. The weighting with the overlap matrix is argued in the following way: if two orbitals have large overlap (which implies that the corresponding element in the overlap matrix will be large) there will be more interaction, therefore the corresponding element in the Fock matrix must be larger too. In the simplest approximation this relationship may be assumed as a linear proportionality - and that is exactly what is implemented in the GWH scheme.

Superposition of Atomic Densities

Superposition of atomic densities (SAD)[2] is a scheme to construct the initial density by creating a separate, spherically averaged density for every atom in the molecule and adding them together to yield a raw molecular density. This raw density is then used to construct a raw Fock matrix, which is diagonalized to give an orbital structure to be used as initial guess. While this may seem rather simple, it is actually a very powerful scheme, as the density only needs to be calculated once per atom species. Therefore, even for very large molecules, a guess can be obtained with very little effort if many of the atoms featured in the molecule are of the same element (as is e.g. the case in metallic clusters). However, SAD can be laborious to implement as a separate implementation is necessary for every desired basis set. Also, it is entirely independent of the geometry of the molecule. For two different isomers of a molecule SAD will thus return the same guess

in the sense that the density matrices corresponding to the initial guesses will be identical (if the atoms are stored in the same order for both isomers).

Minimal Basis with projection

Very often it is necessary to use a larger basis set (see Section 2.2) to obtain results in an acceptable quality. However, in some cases it might be practical to do a first calculation in a much smaller basis set to obtain an initial guess. While still lacking the accuracy, calculations in a smaller basis provide significant speedups. Very often, the smallest basis available, usually called minimal basis (cf. Section 2.2.3), is used.

2.1.10. Convergence Enhancements

As mentioned above, some molecules tend to show oscillating convergence behavior (some only for certain initial guesses, some more often). SCF Calculations have seen great improvements regarding this and other issues due to the development of convergence enhancement methods. A few of them are briefly discussed below, following the explanations given by F. Jensen in [1, p. 101 *et seqq.*].

Damping

The first thing that comes to mind when dealing with oscillation is to introduce some kind of damping, i.e.

$$\boldsymbol{P}'_{n+1} = \omega \boldsymbol{P}_n + (1-\omega)\boldsymbol{P}_{n+1} \tag{2.35}$$

where the density matrix of the current iteration is coupled the one from the previous iteration with a damping parameter ω.

Direct Inversion of Iterative Subspace

Direct Inversion of Iterative Subspace (DIIS)[19, 20] is a method using data from previous iterations to extrapolate the Fock matrix. First, an error measure (i.e. how far the current solution is away from convergence) for every iteration step n,

$$\boldsymbol{E}_n = \quad \boldsymbol{F}_n\boldsymbol{P}_n\boldsymbol{S} - \boldsymbol{S}\boldsymbol{P}_n\boldsymbol{F}_n \tag{2.36}$$

$$\text{Error} = \quad \operatorname{tr} \boldsymbol{E}_n \cdot \boldsymbol{E}_n, \tag{2.37}$$

is defined. In every iteration step the density matrices $\{\boldsymbol{P}_1, \boldsymbol{P}_2, \ldots\}$ and Fock matrices $\{\boldsymbol{F}_1, \boldsymbol{F}_2, \ldots\}$ are logged and the error matrices $\{\boldsymbol{E}_1, \boldsymbol{E}_2, \ldots\}$ are calculated. Finally, we set for the linear combination of error matrices

$$\boldsymbol{E}_{n+1} = \sum_{i=0}^{n} c_i \boldsymbol{E}_i, \tag{2.38}$$

under the condition $\sum_{i=0}^{n} c_i = 1$. The c_i that minimize this expression are then used to construct an extrapolated Fock matrix,

$$\boldsymbol{F}'_n = \sum_{i=0}^{n} c_i \boldsymbol{F}_i, \tag{2.39}$$

which is then used to calculate the density for the next iteration step.

In Fig. 2.1 the techniques mentioned above are compared for the SCF calculation of an ethyne molecule. As expected, both damping and DIIS lead to a significant speedup, but it also clearly shows the downside of damping. There is always a risk of loosing speed in case of a good initial guess. In the first few steps of the iteration Pure is actually ahead of Damped, but the latter is held back by its nature, which becomes an advantage again a few iterations later, when Pure probably overshoots and thus produces a larger error.

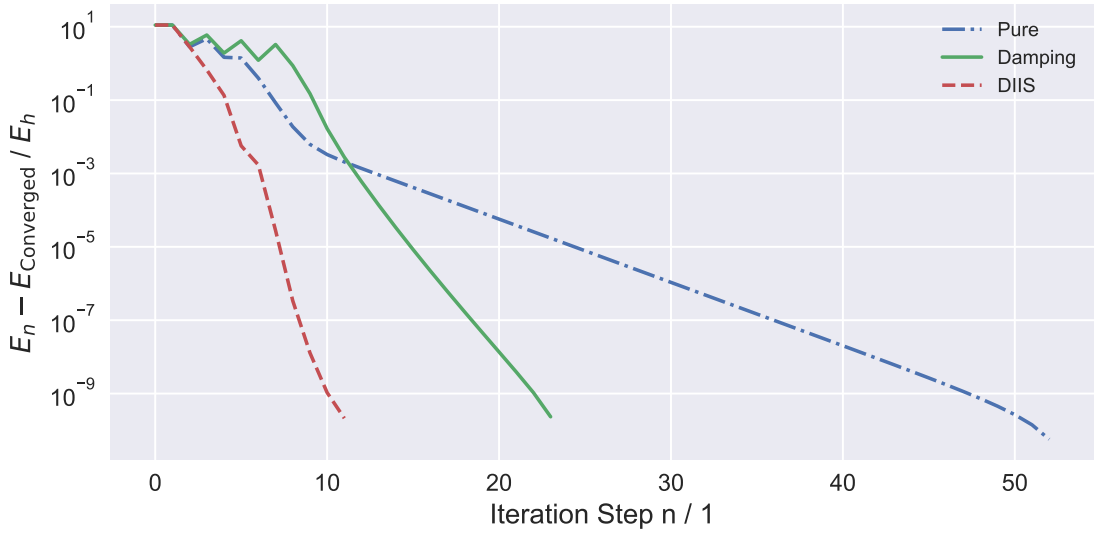


Figure 2.1.: Comparison of convergence enhancements for an ethyne molecule slightly off the equilibrium geometry. The difference between current energy E_n and the converged energy $E_{\text{Converged}}$ are plotted as a function of the step number n. As initial guess H_{core} was used. As criterion for convergence the difference to the previous step's energy must be smaller than $1 \times 10^{-9}\, E_\text{h}$. The line "Pure" depicts the convergence behavior without any enhancement, "Damping" is the result of an SCF algorithm employing a damping of $\omega = 0.3$, and "DIIS" refers to results obtained with DIIS.

2.2. Basis Sets

In Section 2.1.6, an abstract basis was introduced in (2.17). To do concrete computations these basis sets must be implemented in some way. The following describes a few ideas on how this is usually done[7].

[7] See also [1, Chapter 5, p. 192 *et seqq.*] for more information.

The form of the basis set function is often motivated by both theoretical and experimental results. Molecular orbitals are described using "atomic orbitals", which are modeled after results obtained from the atomic Schrödinger equation, and are often parametrized to reproduce empirical data as well as possible. The functions presented are all centered at the nuclei of the molecule, which indicates that electrons may be loosely associated with one atom, but they also make for a key challenge regarding computational effort. A substantial amount of integrals involving the basis functions has to be calculated in the SCF calculation, and this needs to be done as fast as possible to make Hartree-Fock and related techniques computationally feasible.

2.2.1. Slater Type Orbitals

As the Schrödinger equation has an exact solution for hydrogen and was solved in approximations for other atoms it makes sense to use similar functions as basis for molecular orbitals. Very notable are so-called Slater Type Orbitals (STOs), which are modeled after the solutions to the atomic Schrödinger equation and take the form

$$\phi_{\zeta,n,l,m}(r,\varphi,\theta) = NY_{lm}(\phi,\theta)r^{n-1}\exp\{-\zeta r\}, \tag{2.40}$$

where N is a normalization constant, n the principal quantum number, ζ a tuneable parameter and $Y_{lm}(\phi,\theta)$ the real spherical harmonics. With their exponential form they ensure rapid convergence regarding the number of functions used. However, they do not exhibit radial nodes and must be (e.g. linearly) combined to introduce the physically correct radial behavior. However, their biggest drawback is that the many two-electron integrals cannot be calculated analytically, making these orbitals a rather inefficient choice from a computational point of view.

2.2.2. Gaussians: STO-nG

A way to make basis functions more efficient is to use Gaussian functions, which allows analytical integration and therefore much faster evaluation. Orbitals of this type are called Gaussian Type orbitals or GTOs. While notably decreasing the computational effort of ab initio calculations GTOs by themselves, they also have a distinct disadvantage in comparison to STOs: their slope becomes zero at their nucleus which is unphysical. Therefore GTOs perform very badly at describing electronic behavior close to the atoms. A way to mitigate this problem is to "contract" two or more GTOs in a linear combination to approximate STOs. An example of these orbitals is the STO-nG group, where STOs are approximated by a sum of n Gaussians,

$$\phi_{\boldsymbol{\alpha}}(r) = N\sum_{i=1}^{n} c_i \exp\{-\alpha_i r^2\}, \tag{2.41}$$

with $\boldsymbol{\alpha} := \{\alpha_1,\ldots,\alpha_n\}$ as the centers of the Gaussian functions. In the equation above only the radial part of the orbital is given; the angular part is expressed using real spherical

harmonics once more. However, there is also an alternative form where GTOs are expressed in Cartesian coordinates[8].

Common choices for n are values between 2 and 6. In Fig. 2.2 STO-2G, STO-3G und STO-6G are compared to an STO. As explained above, the slopes near the nucleus and the form of the curves for great radii is inferior to STOs' even for large n.

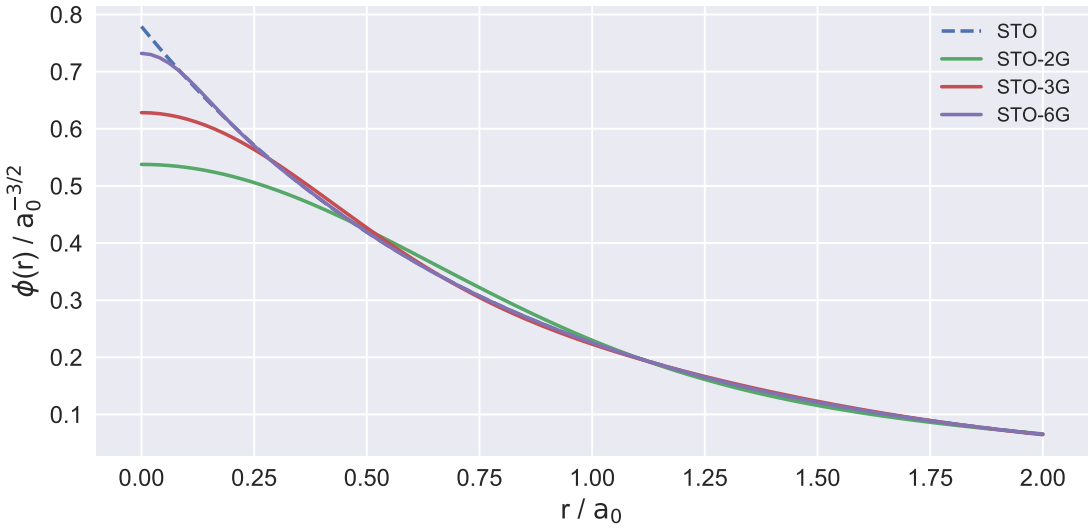


Figure 2.2.: STO($\zeta = 1.24$) and contracted GTOs (namely STO-nG for $n \in \{2, 3, 6\}$) for the 1s orbital of hydrogen. Parameter for ζ is taken from [14, p. 159], the STO-nG parameters from EMSL basis set library [21, 22]. The figure is modeled after Fig. 3.3. from [14, p. 158].

2.2.3. Split-Valence, Diffuse, and Polarized

The remaining question is now how many basis functions should be used. A minimal basis is a basis set with as few functions as necessary to distribute all electrons in the neutral atom/molecule. Surely this cannot be enough. Therefore, a first approach is to simply double the number of basis functions (Double Zeta or DZ bases). Here two STOs (approximated by contracted GTOs) are used for a single electron. However, the electrons close to the nuclei "core electrons" will usually behave very similar to how they would in an atom, while the valence electrons have a much greater impact on the chemical behavior of the atom in the molecule. A more economic ansatz is therefore to only double the basis for valence electrons creating a split-valence basis, which is a better tradeoff between computational cost and accuracy. Split valence Double Zeta bases are usually denoted as VDZ (for Valence Double Zeta). With greater computational abilities, this scheme can be repeated to create Triple Zeta (DZ) bases with three times and Quadruple Zeta (QZ) with four times as many orbitals as the minimal basis and so on.

[8] For more details see also [14, p. 153 *et seqq.*].

Another way to go is to specifically add higher angular momentum orbitals (e.g. extra p-orbitals to hydrogen, which in minimal basis would only have s-orbitals). These functions are then called polarization functions and are of particular importance for electron correlation. Another type are "diffuse" functions with very small exponents α_i. They extend very far from the nuclei and are used to describe loosely bound electrons.

2.2.4. Basis Set Superposition Error (BSSE)

Using a finite basis triggers a whole series of systemic errors. The fact that basis functions are usually centered around the nuclei of the system gives rise to the BSSE[23]. A basis of atom-centered basis functions does not always provide the same accuracy for different molecular geometries. The reason for this is that sometimes basis functions that belong to a second nucleus can improve the description of the electronic density at a first nucleus. This is of course only relevant if the two atoms are close enough, which introduces a geometry-dependent bias. When calculating energies for different geometries, this has to be accounted for.

2.3. Artificial Neural Networks

The core topic of this thesis concerns the applicability of neural networks to a very specific field of computational chemistry. In this section, we will take a more general look at neural networks and their components. The text loosely follows the book of J. Heaton[9].

2.3.1. Introduction

Inspired by the human brain, artificial neural networks (ANNs) are a collection of interconnected nodes ("neurons"), i.e. cells that can receive an input and output a signal to other cells that is in some relation to the input (e.g. it may only be sent if the input surpasses a certain threshold). The idea of ANNs is not new, with the first publications on this topic dating back to the 1950s. One of them was F. Rosenblatts Perceptron[25], a prototype of a neural network. Back then, they could hardly be realized due to technical limitations. However, in the last few years ANNs have seen a renaissance and are now widely used in many field and in many variations, as greater computational capabilities allow for complexer and thus more powerful models. They are used within a wide range from various image-processing applications (such as facial recognition) over acoustic engineering (e.g. natural language processing) to more or less complex regression problems.

2.3.2. Feed-Forward Networks

We will make use of the simplest type of ANNs, a so-called feed-forward network (FNN). They comprise multiple sets of nodes (called layers), in which each node is connected to

[9] Most notably Chapters 1, 4, 5, and 6[24], see p. 1-25 and p. 65-130.

every node of the previous and the following layer. Exceptions are the first (the input layer) and the last (output layer), which have only connections to the following and the previous, respectively. FNNs can be understood as directed graphs, with the direction going from the input, via the so called hidden-layers, to the output layer. In Fig. 2.3 an ANN is drawn in a very schematic way. The connections between the neurons are directed

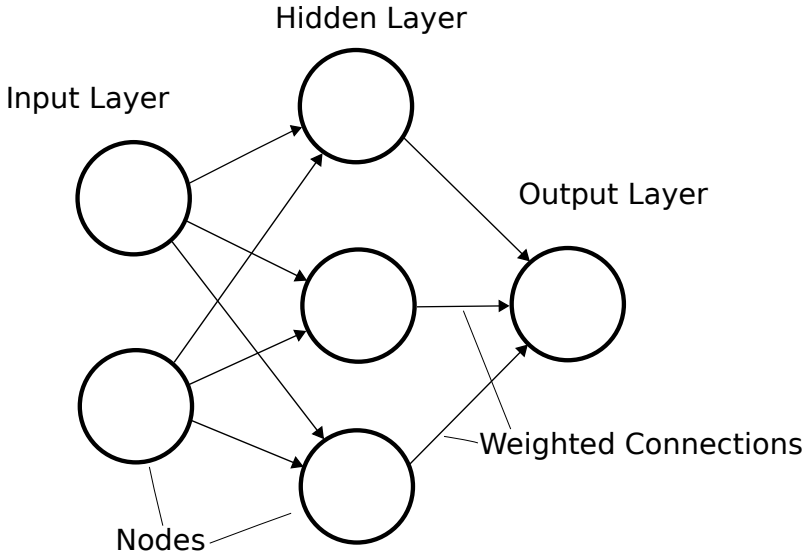


Figure 2.3.: Schematic drawing of an ANN, consisting of a single hidden layer.

(sending the input from the input layer to the output layer) and also weighted. We will now briefly explain the situation for the node i in the network layer j. Each connection from layer $j-1$ to a neuron in layer j represents the output y_k^{j-1} of the neuron k in the previous layer. In neuron i of layer j, all inputs from the neurons in the previous layer are weighted by weights w_{ki}^j and summed up to yield the "raw input"

$$\tilde{z}_i^j = \sum_k w_{ki}^j y_k^{j-1}. \tag{2.42}$$

By adding the bias b_i^j we form the "total input"

$$z_i^j = \tilde{z}_i^j + b_i^j = \sum_k w_{ki}^j y_k^{j-1} + b_i^j. \tag{2.43}$$

This can also be expressed in matrix notation:

$$\boldsymbol{z}^j = \boldsymbol{W}^j \boldsymbol{y}^{j-1} + \boldsymbol{b}^j. \tag{2.44}$$

In addition to the above, every node (or rather every layer) is associated with an activation function f, which determines how much output y there will be for a given total input from nodes from the previous layer.

$$y_i^j = f^j\left(z_i^j\right) = f^j\left(\sum_k w_{ki} y_k^{j-1} + b_i^j\right), \tag{2.45}$$

or in matrix notation

$$\boldsymbol{y}^j = \boldsymbol{f}^j\left(\boldsymbol{z}^j\right) = \boldsymbol{f}^j\left(\boldsymbol{W}^j\boldsymbol{y}^{j-1} + \boldsymbol{b}_i^j\right), \tag{2.46}$$

This is graphically captured in Fig. 2.4, where the neuron described above is shown. Note that the output of every node (except for the output layer) will be the input for nodes in the next layer.

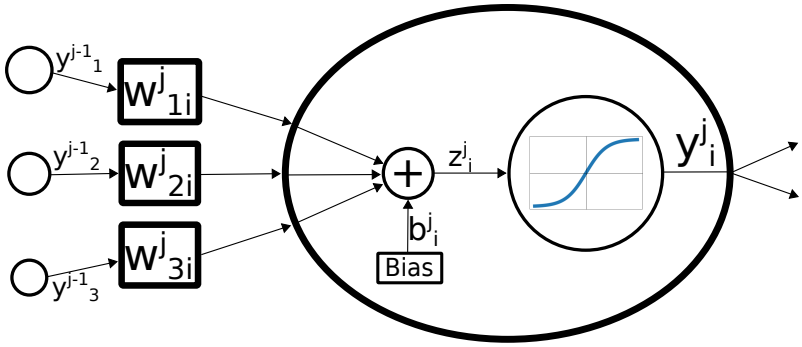


Figure 2.4.: A depiction of neuron i in layer j and all quantities involved in calculating its output.

Universal Approximation Theorem

FNNs are a very general class of non-linear mappings. They have been shown to be able to approximate any real continuous function with a single hidden layer containing sufficiently many nodes to arbitrary accuracy if a non-linear activation functions is used[26]. This shows how powerful they are in principle suggesting a broad spectrum of applications.

2.3.3. Network Training

Initialisation

As soon as the architecture of a network is decided, it needs to be be initialized. There are many different schemes to initialize a network in a suitable fashion for the task at hand; in this work, we will initialize the weights of new networks randomly with a suitable normal distribution and the biases set to zero unless stated otherwise.

Normalisation

A common pre-processing step is to "normalize" input data. This means that the data is transformed by a linear mapping to have a mean of zero and a variance of one in for every input feature. A normalization is applied in order to make different input features comparable to each other, and to let the network learn in an unbiased way which feature is more important.

Activation Functions

The form of the activation function can have rigorous effects on network training speed and performance. Classical choices were also influenced by human nerve cells, which remain inactive until a threshold is reached. In Fig. 2.5 a few common examples are given. As we will be doing regression, the activations of the last layer will be linear, to be able to reach all real numbers. Activation functions can also be used to introduce normalization for the inputs of inner layers and mitigate what is called the vanishing/exploding gradient problem. The more hidden layers the networks have, the harder they are to train, because the gradients that appear in the hidden layers tend to become very large or very small for classical activation functions such as e.g. the hyperbolic tangent. Recent research[27] has shown that this can be avoided if the activation function has certain properties (e.g. allowing for negative activation values, showing almost linear behavior for positive inputs and a saturation behavior for negative ones, etc.). A recently introduced function that provides all desired features is the exponential linear unit function (called "ELU")[27], which will be used throughout this thesis,

$$\mathrm{ELU}(x) = \begin{cases} x & \text{if } x > 0 \\ a\,(\mathrm{e}^x - 1) & \text{otherwise,} \end{cases} \tag{2.47}$$

where a is a tuneable parameter and must be positive.

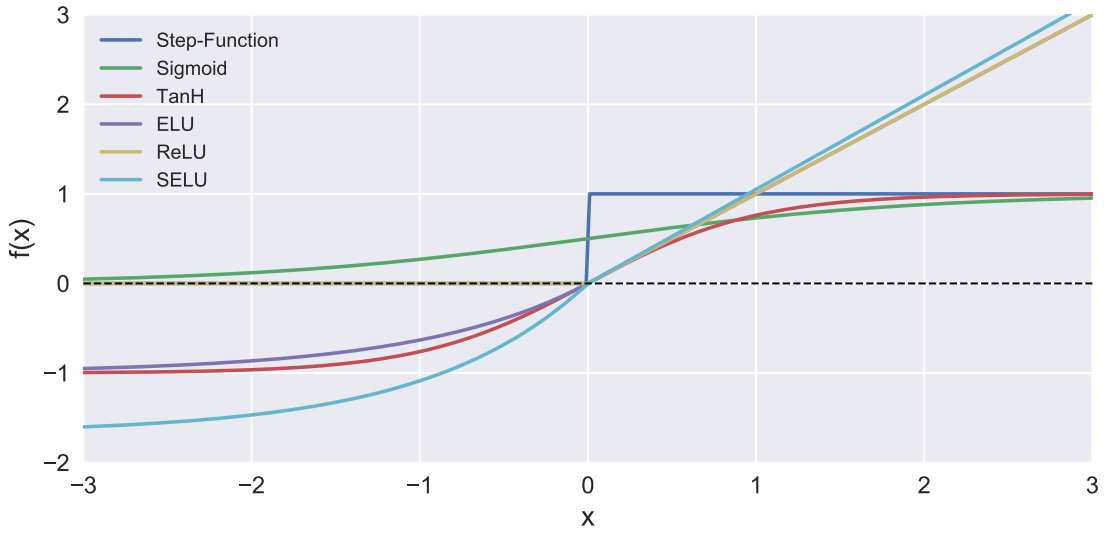


Figure 2.5.: A few examples of activation functions. the Heaviside step function (with a threshold of 0 and a step of 1 are displayed), the sigmoid function (very smooth, usually used for classification), the hyperbolic tangent function (very popular because of its smoothness and asymptotic behavior, which exhibits saturation effects), the exponential linear unit function (ELU), the rectified linear unit function [28] (ReLU, finds great application in image and natural language processing) and the scaled exponential linear unit function (SELU)[29].

Optimizers

After initialization the network needs to be trained. In other words, the weights and biases have to be optimized to minimize a cost of some sort. This cost function, which we shall denote as E, is usually a measure of how well the network performs, and can be as simple as a mean squared error[10]. Very often it is a combination of an error measure and other terms that penalize certain unfavorable properties (e.g. for regularization; see below for more details). Finding the minimum of this cost function is a classical optimization problem and there is a nifty solution to it: the backpropagation algorithm [30–32]. In principle, it can be described as a layer-wise unfolding of the ANN by applying the chain-rule of differential calculus[33]. The process is presented in Algorithm 2.

Algorithm 2: The backpropagation algorithm

Data: Initilized network components: weights, biases etc. (denoted by $\boldsymbol{w}$)
Data: Traning data, a set of 2-tuples of the form $(\text{input}, \text{output})$
Data: Error or cost function E
Data: Lerning rate η
begin
  while *stopping criterion not met* **do**
    foreach *sample* (x, y) *in the test dataset* **do**
      foreach *network variable* $w \in \boldsymbol{w}$ **do**
        Calculate the backpropagation error $\delta = \frac{\partial E(f_{\tilde{w}}(\tilde{x}),\tilde{y})}{\partial \tilde{w}}\Big|_{\tilde{x}=x,\tilde{y}=y,\tilde{w}=w}$;
        Update the variable $w = w - \eta\delta$;
      end
    end
  end
end

Algorithm 2 is a form of the Gradient Descent algorithm[11] and requires the gradients of the cost function. The computation of these gradients is one of the most complicated parts of the training, due to the large number of parameters in a neural network. The term backpropagation, in a more narrow sense, thus often only refers to the efficient calculation of the gradients via the chain-rule. The time needed to calculate these gradients has a large effect on the training speed. Therefore, it is important to choose and implement cost function, activation functions etc. in a way that they can be easily (preferably analytically) calculated and evaluated.

Another way to train ANNs is the Adam algorithm[34] which uses adaptive "momenta" to further accelerate the learning (hence the name). It also features an adaptive learning rate which gets larger for steep parts of the cost function and smaller closer to a minimum.

[10] The sum of squared differences between network predictions and target values.

[11] In Gradient Descent, a (local) minimum of a real, continuous, scalar function $E : \mathbb{R}^m \to \mathbb{R}, \boldsymbol{x} \mapsto E(\boldsymbol{x})$ of m variables is found iteratively, starting from a given initial guess $\boldsymbol{x}_0$, by repeatedly "moving" in the direction of the negative gradient, $\boldsymbol{x}_{n+1} = \boldsymbol{x}_n - \eta\nabla E(\boldsymbol{x}_n)$, with a step size $\eta \in \mathbb{R}$.

Algorithm 3: The Adam (adaptive moments) algorithm. δ is a small constant introduced to ensure numerical stability, i.e. to avoid division by zero.

Data: Initialized network components: weights, biases etc. (denoted $\boldsymbol{w}$)
Data: Traning data, a set of 2-tuples of the form $(\text{input}, \text{output})$
Data: Error or cost function E
Data: Step size ϵ and two decay rates $\rho_1, \rho_2 \in [0, 1)$. Default: $\rho_1 = 0.9, \rho_2 = 0.999$
begin
  Initialize the two momentum variables: $\boldsymbol{m}_1 = 0, \boldsymbol{m}_2 = 0$;
  Initialize the training step counter: $t = 0$;
  while *stopping criterion not met* **do**
    foreach *sample* (x, y) *in the test dataset* **do**
      Update counter: $t = t + 1$
      Calculate the backpropagation error: $\boldsymbol{\delta} = \nabla_{\boldsymbol{w}} E(f_{\boldsymbol{w}'}(\tilde{x}), \tilde{y})\big|_{\tilde{x}=x, \tilde{y}=y, \tilde{w}=w}$;
      Update the first momentum estimate: $\boldsymbol{m}_1' = \rho_1 \boldsymbol{m}_1 + (1 - \rho_1)\boldsymbol{\delta}$;
      Update the second momentum estimate: $\boldsymbol{m}_2' = \rho_2 \boldsymbol{m}_2 + (1 - \rho_2)\boldsymbol{\delta} * \boldsymbol{\delta}$;
      Correct bias in first momentum: $\boldsymbol{m}_1 = \frac{\boldsymbol{m}_1'}{1 - \rho_1^t}$;
      Correct bias in second momentum: $\boldsymbol{m}_2 = \frac{\boldsymbol{m}_2}{1 - \rho_2^t}$;
      Compute the update: $\delta \boldsymbol{w} = -\epsilon \frac{\boldsymbol{m}_1}{\sqrt{\boldsymbol{m}_2} + \delta}$;
      Update the network variable: $\boldsymbol{w} = \boldsymbol{w} + \delta \boldsymbol{w}$;
    end
  end
end

As opposed to Algorithm 2, the updates for the network variables $\boldsymbol{w}$ are all calculated simultaneously in Algorithm 3. The multiplication of the backpropagation error $\boldsymbol{\delta}$ with itself must be understood as element-wise, as well as the square root of the second weight $\sqrt{m_2}$ in the calculation of the update vector. Due to this, there is also an individual learning rate for every parameter.

Learning Rate

Obviously, one must be careful when choosing a certain learning rate η (in Algorithm 2) or ϵ (in Algorithm 3). If it is too large, it may become difficult to converge, but if it is too small, one is likely to get stuck in a local minimum. Since this is a difficult choice, it is a common practice to start with a larger rate and decrease it during training. This is called learning rate decay. However, some optimizers adapt their equivalent of a learning rate automatically. An example for this behavior is Algorithm 3.

Mini-Batching

While it is of course possible to use all of the available training data in every training step (see e.g. Algorithm 2), research has shown that this bears the increased risk of getting stuck in local minima of the cost function. To reduce this risk, mini-batching was

introduced, where the dataset is subdivided into (often not exclusive) subsets over which a single optimization step is performed. The randomness in the selection of the subset elements used in an optimization step helps to avoid overfitting and makes it easier to overcome local minima. The batch size used in training is a "hyperparameter", i.e. a fundamental, often manually chosen parameter of the ANN implementation, that must be optimized for every application individually.

Regularization

Another widely used trick to avoid overfitting is the introduction of a penalty for large weight or bias values. This is called weight decay[35] or regularization, and is very popular in the machine learning community. For example, a given cost function $E(\boldsymbol{x})$ for an input $\boldsymbol{x}$ could be augmented with the Euclidean norm[12] of the "vector" of all weights $\boldsymbol{w}$ of a network trained with this function,

$$E(\boldsymbol{x})' := E(\boldsymbol{x}) + \lambda \|\boldsymbol{w}\|, \tag{2.48}$$

where λ is the coupling parameter which has to be chosen in a way to make sure that the regularization term is of a suitable order of magnitude.

2.3.4. Neural Networks in Quantum Chemistry - Descriptors

Machine leaning, and in particular ANNs, are already widely used in quantum chemistry nowadays. As mentioned above, a very common application is the interpolation of the potential energy surface or the prediction of any other molecular quantity based on a dataset of ab initio calculated examples (see e.g. Behler and Parinello [36]). Whatever the task in detail is, the atomic environment has to be captured somehow. It has to be encoded in a format that can be fed into ANNs[13]. This "finger print" of a local molecular environment is a vector or matrix, which we shall call generically "symmetry vector", throughout this thesis. It is the result of what we will call "descriptors" or "symmetry functions", which map features of the environment (such as positions and species of the atoms in a molecule, ionization energies, etc.) to an abstract vector, which rarely has any physical meaning anymore. This is illustrated in Fig. 2.6. While there are many ways to set up these mappings, they need to reflect certain physical principles such as given symmetries and invariances, otherwise unphysical predictions will follow. An important example is the fact that the representation of a molecular system must not depend on the order the atoms are stored in. Computers can in principle only work in sequences. They store only a representation of the molecule by remembering the atoms and their positions in the molecule. Unfortunately, this representation is not unique, while physical quantities such as the energy are. If a model gave a different prediction for, say, the energy

[12] Regularization is not restricted to the Euclidean norm. In fact, the absolute-value norm is also very popular in this context in machine learning.

[13] It is worth mentioning that an ANN ansatz does not imply that a single network is doing all the work. Behler and Parinello for instance used multiple "atomic networks" that would estimate the contribution of a single atom to the total energy of a molecule based on a description of the environment of that molecule [36]).

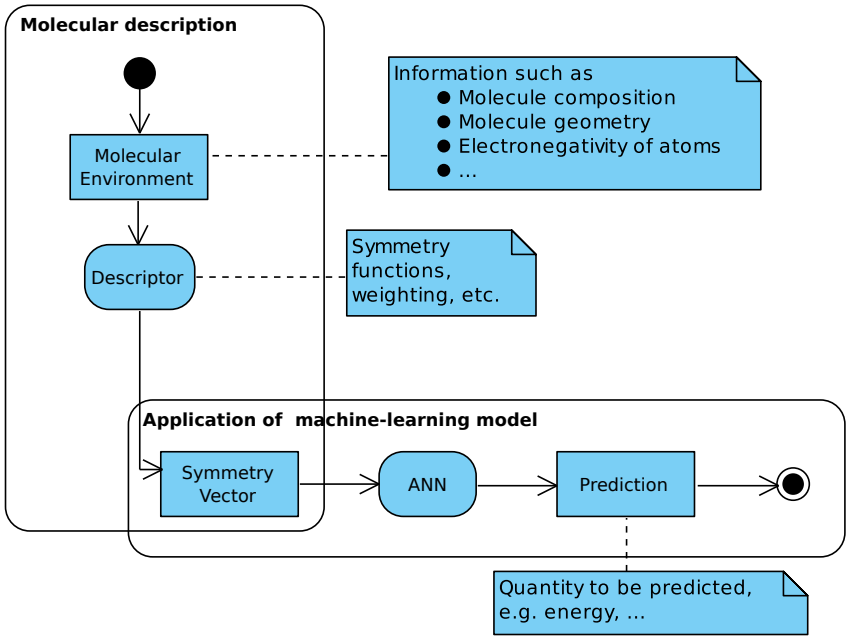


Figure 2.6.: The path from atomic information to neural network input (symmetry vector) to neural network prediction.

for molecules of equal geometry, because they were stored differently, this would lead to an immediate contradiction.

As of now, the scientific community is mostly focussing on estimating scalar properties, such as the molecular energy or the norm of the dipole moment. Higher dimensional quantities that depend on the orientation of the molecule, such as e.g. the vector of the dipole moment, have not drawn much attention so far. A reason for this is the fact that a lot of symmetries and thus physical information that can be coded into the descriptors can not be applied anymore. If a network is supposed to estimate a scalar, the descriptor mapping should probably be invariant to permutation of the atoms in the molecule as well as to rotation and translation or even exchanging them (among related species). Otherwise, equivalent cases would end up with two different inputs, yielding a redundant and possibly contradictory dataset. All these constraints - once they are met - might also simplify the usage of "divide and conquer"-schemes for increased flexibility regarding the types of systems that can be dealt with. A comparison of some selected descriptors is provided in a recent article of Bartók et al. [37]. An example of such a descriptor is the Coulomb matrix [38],

$$M_{ij} = \begin{cases} \frac{1}{2} Z_i^{2.4} & i = j \\ \frac{Z_i Z_j}{\|\boldsymbol{R}_i - \boldsymbol{R}_j\|} & i \neq j \end{cases}, \tag{2.49}$$

which was used in a model estimating atomization energies of molecules.

3. From Overlap to Density: a First Approach

3.1. Introduction

The generation of an initial guess should be computationally inexpensive. It is not the goal to go as far as to create a nearly converged solution for the guess if it is more expensive than a single point SCF calculation itself. This would defeat the purpose of the SCF procedure. To keep the computational expense of our guess as small as possible, it would be very practical if it could work with an input that is simple to produce from the information that is already at hand. However, it is crucial that all necessary information about the atomic environment is fully captured. The calculation of the core Hamiltonian matrix $\boldsymbol{H}^{\text{core}}$ and the overlap matrix $\boldsymbol{S}$ are mandatory steps of any SCF algorithm (cf. Algorithm 1). As described in Section 2.1.9, the core Hamiltonian has already been used to generate guesses by approximating the Fock matrix:

- H_{core} uses the the core Hamiltonian itself as an estimate to the Fock matrix.
- GWH employs the diagonal of the core Hamiltonian in combination with the overlap matrix to make a guess for the Fock matrix.

The core Hamiltonian only contains information about the situation of a single electron in the field of the nuclei. Therefore, H_{core} can only give a qualitative idea of the elements in the density and Fock matrix. The fact that it does not include any information on electron-electron interaction is a huge disadvantage. The overlap matrix, on the other side, does include this information (at least to some extent). If two atoms are closer, the basis functions centered on them will overlap more strongly. The number of electrons in a system is also encoded in the number of basis functions used. In a GWH guess, $\boldsymbol{H}^{\text{core}}$ provides formation on the nature of the atoms in the molecule and the overlap matrix specifies the geometry. The question at hand is now if other properties, such as the density matrix, can also be deduced from information encoded in the overlap matrix. In the following chapter we will investigate whether there is enough information and whether it is possible to find a mapping from the overlap matrix to the density matrix $S_{\mu\nu} \mapsto P_{\mu\nu}$.

The size of the density matrix depends on the number of basis functions used, which in turn depends on the basis set, and most importantly, on the number of atoms and electrons in the system. Therefore, the computational treatment is restricted to a single molecule for now. Our molecule of choice should therefore be a system that does not consist of too many electrons but still shows a non-trivial electronic structure. We choose the molecule 1,3-butadiene, an organic molecule with the empirical formula C_4H_6. The overlap matrix of butadiene (close to equilibrium geometry) is shown in Fig. 3.1.

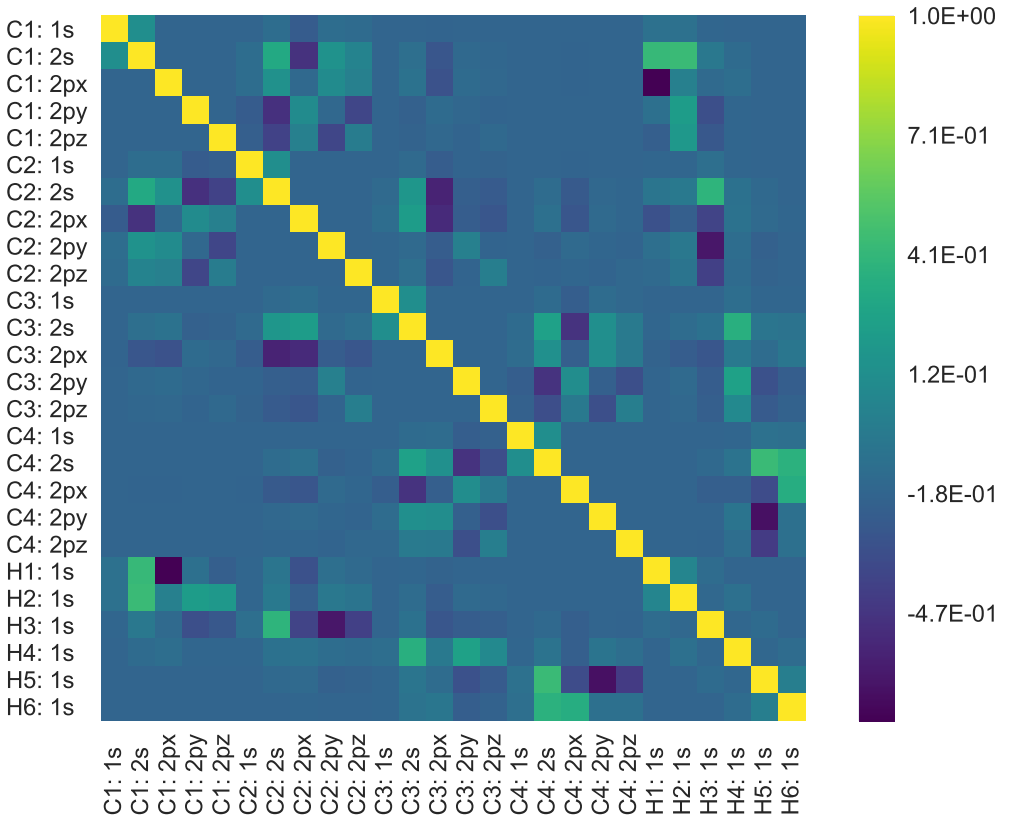


Figure 3.1.: The overlap matrix of a butadiene molecule (near equilibrium geometry) as obtained with the STO-3G basis[39]. The basis functions are labeled according to the atom they are centered at (cf. Fig 3.2).

3.2. A Neural Network Guess

We will now investigate a suitable model. This involves finding the right network architecture, training parameters and a few other factors. Finally we will train our model and see how it performs. Fig. 3.2 depicts a butadiene molecule.

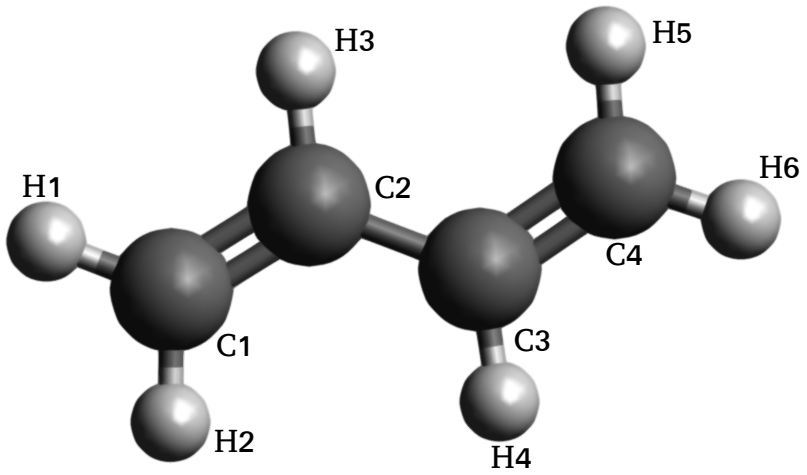
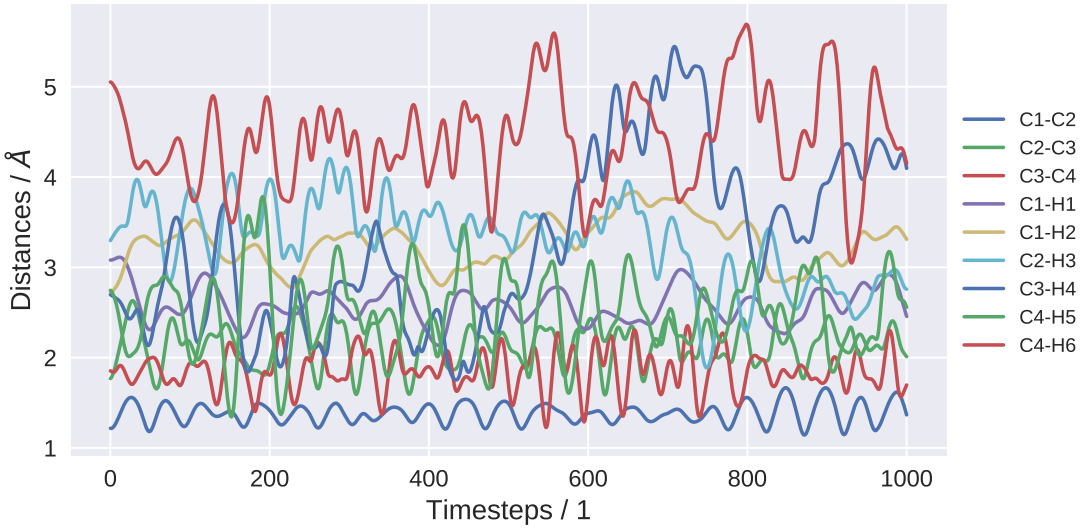


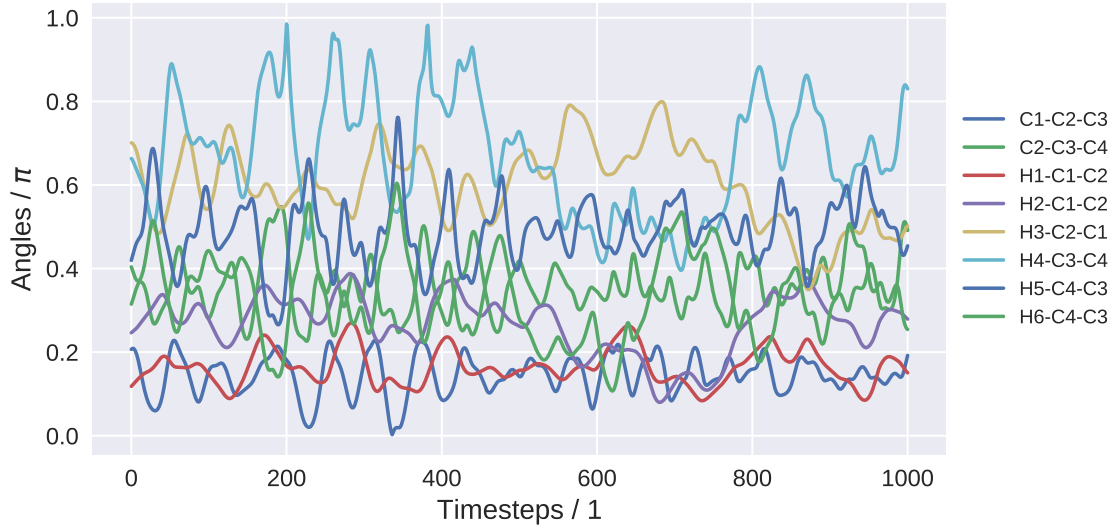
Figure 3.2.: A 3D view of a butadiene molecule, very close to its equilibrium geometry. Created and captured with Avogadro.

3.2.1. The Dataset

Training of any model requires data. A very common approach in the quantum chemistry community is to sample data using molecular dynamics (MD). This method has also been chosen for data generation in this work. Starting from a specially tweezed starting geometry (not too close to the equilibrium geometry) 1000 simulation steps of about 0.5 fs are executed at 450 K. This small time step was necessary in order to provide an appropriate resolution of molecular motion at this high temperature. The high temperature in turn is chosen so that a large volume of the phase space of the system can be explored and more geometries further away from equilibrium can be sampled. From the sampled geometries the overlap matrix and density matrix were calculated in STO-3G basis[39]. The latter is a minimal basis, but leads to matrices with 676 elements already for our system of choice. To illustrate the width of the variation of geometries in the dataset the evolution of the values for distances and angles in the molecule during the MD simulation is presented in Fig. 3.3.



(a)



(b)

Figure 3.3.: Molecule geometry parameters over time. For orientation, the atoms have been labeled (cf. Fig. 3.2). (a) shows distances and (b) angles between nearest-neighbor atoms.

Another way to analyze the diversity of the dataset is to take a look at the energies that appear in it. If we look at the histogram of the energy distribution in Fig. 3.4 we can see that highest and lowest energies are more than $0.35\,E_\mathrm{h} \approx 9.5\,\mathrm{eV}$ apart.

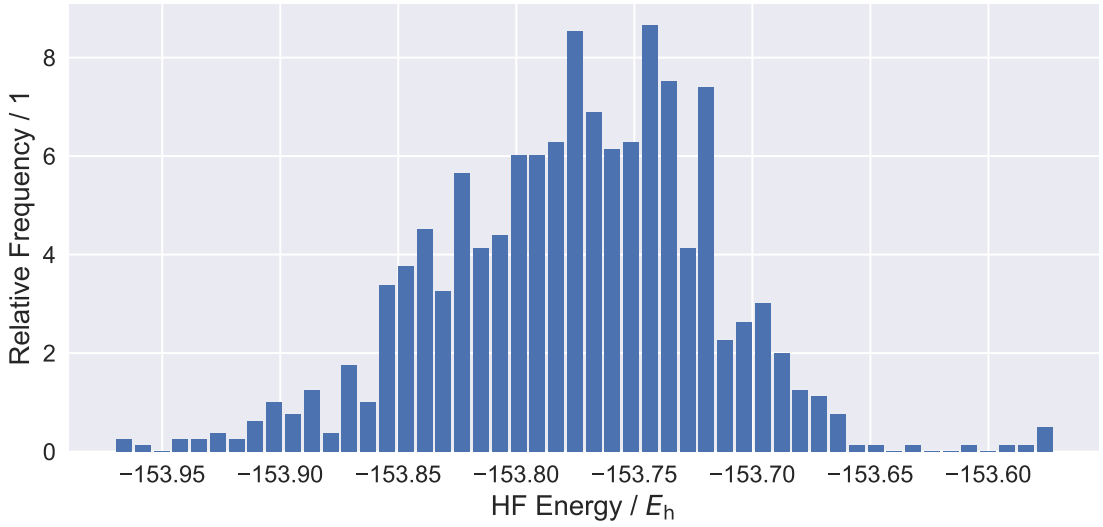


Figure 3.4.: Distribution of energies in the dataset.

Since both overlap and density matrix are symmetric by construction, almost half the elements (e.g. those of the lower triangular half, below the diagonal) only contain redundant information. In the ANN mapping they can be omitted and reconstructed in post processing, which leaves us with 351 elements. The final dataset will be split into three subsets. 20 % are reserved for testing the trained networks. Of the remaining samples 20 % are used for validation and the rest for training.

3.2.2. Plain Neural Network Fit

Network Architecture and Hyperparameters

At the beginning, a few networks with varying size are trained with a simple cost function comprising the mean squared error and a regularization term (coupled with $\lambda = 1 \times 10^{-6}$). The network weights are initialized with a truncated normal distribution, which is centered around 1×10^{-2} with a variance of 1×10^{-2}, while the biases are initialized to zero. For our training we use a special algorithm in which the learning rate is altered. The process is outlined in Algorithm 4. In principle it works as follows: As described in Section 2.3.3, the learning rate is decayed if no further progress can be made. When the learning rate gets too small it is reset to its initial value.

Algorithm 4: Learning rate decay used to train the initial guess networks. Training is performed until a given number of resets is reached. During all of the training steps, whenever a new "best" model is found, it gets stored, thus ensuring that no progress is lost, even if the cost goes up again due to a learning rate reset.

```
Data: Inital learning rate η_0, minimum learning rate η_min, reduction factor f < 1
Data: Two thresholds θ_1, θ_2, where θ_1 < θ_2
begin
    Initialize reset counter;
    while maximum number of resets not reached do
        (Re-)Initialize learning rate η = η_0 ;
        while change in validation cost is not lower than θ_1 do
            while change in validation cost is not lower than θ_2 do
                Train the network with the current learning rate;
                if validation cost reaches new minimum then
                    Store the current network;
                end
            end
            if updated learning rate not too small, i.e. fη > η_min then
                Reduce the learning rate η = fη;
            end
        end
        Advance reset counter;
    end
end
```

A typical example of the cost evolution in such a training is displayed in Fig. 3.5, together with the evolution of the learning rate.

To test which architecture makes the most sense a few models are trained using Algorithm 4 with a fixed number of 3 resets. This is done for each model with three different learning rates[1] and repeated 5 times for each learning rate to account for statistical fluctuation. The result is shown in Fig. 3.6.

[1] Higher learning rates typically improve the training performance in small models but can be ineffective in larger ones.

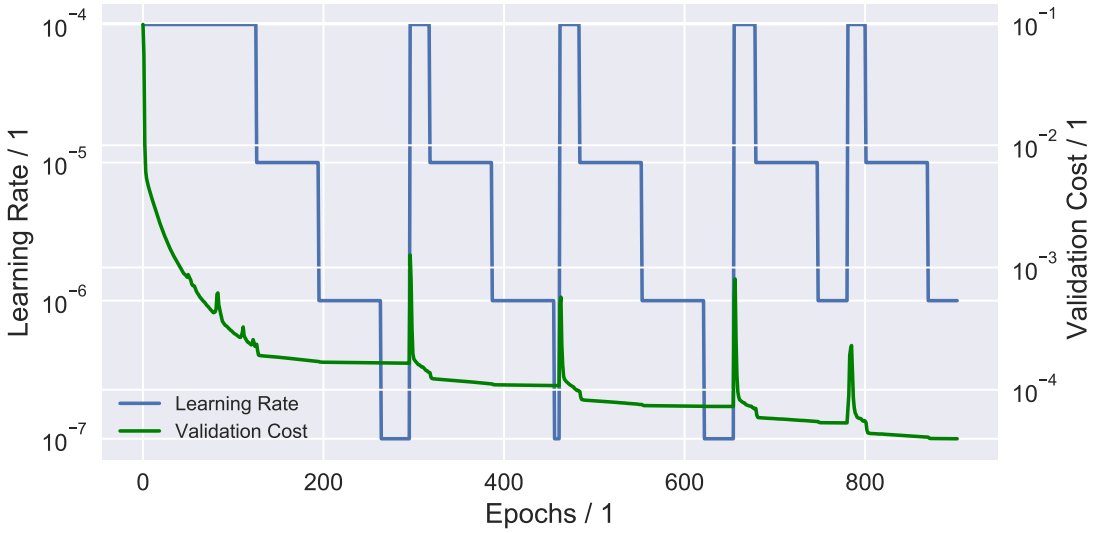


Figure 3.5.: Evolution of the costs (MSE) and the learning rate during the training of an ANN with Algorithm 4.

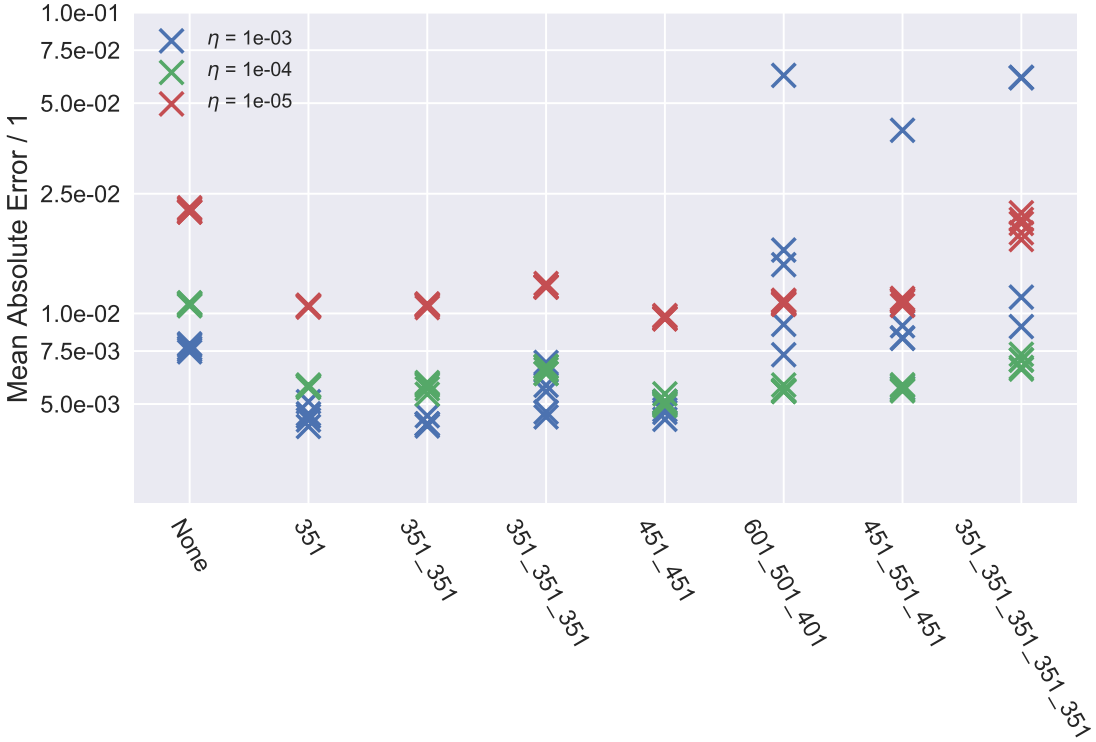


Figure 3.6.: Various network architectures in comparison. The number of neurons per hidden layer are noted along the abscissa, separated by underscores.

Four of the analyzed models (2^{nd}-5^{th}) are fairly close for all learning rates, making it hard to decide on the final structure from this data. Therefore, some additional manual testing is required. Note first that there is a certain bias. Smaller models can usually be trained a lot faster than larger, deeper models (especially if Algorithm 4 is employed). Evidence of this is the bad performance of the larger models for lower learning rates. This difference in training speed is a significant factor in parameter studies as the one above, because the true power of the deeper networks cannot always unfold in the short time of just three resets. There is always a tradeoff not only between training time and accuracy, but also between minimum training time and maximum achievable accuracy. With this in mind, and after some further testing, in spite of the results from Fig. 3.6, a network with three hidden layers in a wedge form (neurons per hidden layer: 601-501-401) proved to achieve the lowest error.

Analysis of Network Performance

We will now compare the guess of our chosen network to classical initial guess schemes. As explained in Section 2.3, neural networks are basically a nonlinear fit with many variables. Thus, there are many local minima to every network training, all of which imply different prediction capabilities for the trained network. All values listed below may vary therefore for equal network structures trained with the same algorithm on the same dataset. A first look at the absolute errors achieved by the network fit in comparison to the classical guesses (Tab. 3.1) seems quite promising.

Table 3.1.: Average (over all matrix elements and over all samples of the test set) of mean absolute error (MAE) of initial guess schemes. The value of the neural network is denoted by NN.

	$\mathbf{H_{core}}$	**SAD**	**GWH**	**NN**
MAE / 1×10^{-3}	300 ± 190	96 ± 31	56 ± 8	0.3 ± 0.1

In order to deepen our insights into the prediction capabilities of out network we compare the network and the classical predictions to the corresponding values in the converged density matrix in Fig. 3.7.

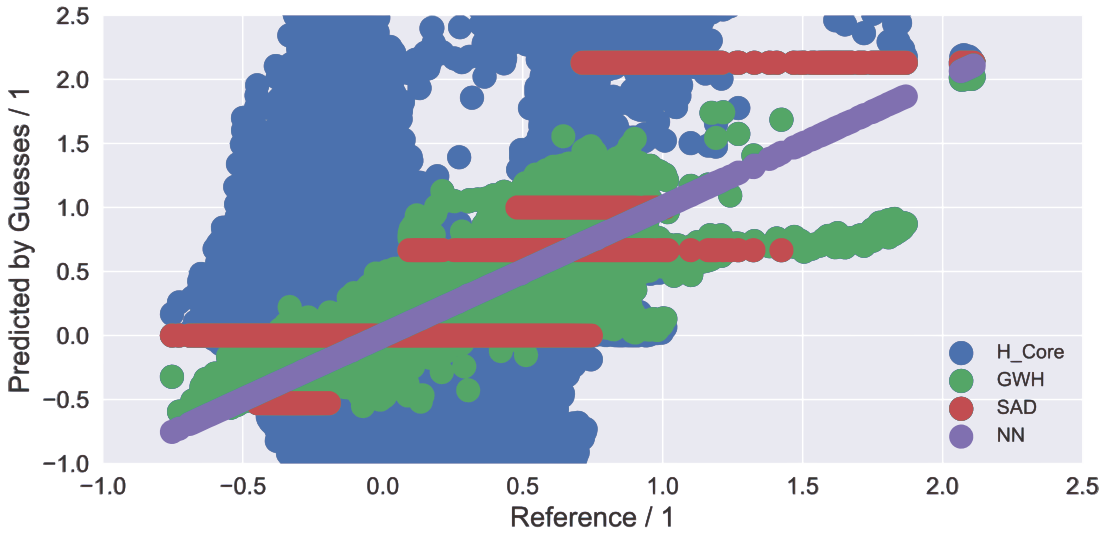


Figure 3.7.: The network predictions for elements of the density matrix against their reference values.

In Fig. 3.7 all values are compared the same way. However, this is only fair to some extent as some do not contribute as much to observables such as e.g. the total energy as others. Examining the error for each element of the density matrix shown in Fig. 3.8, we see that it is rather distributed. However, some weaker areas are e.g. in the areas corresponding to basis functions that are centered at hydrogen atoms. This not surprising, since the hydrogen atoms definitely moved around more easily than the carbon atoms during the MD simulation (as the hydrogen atoms are much lighter), causing higher variance in these regions of the density matrices in our dataset. Strongly varying targets are always harder to learn, thus the difference in achieved performance. Nevertheless, the difference is fairly small and the overall performance solid. To test whether the difference in quality for different regions of the matrix will have distorting side effects, we will take a look at how well our guess is able to predict the HF energy directly in comparison to our classical guesses, which do not show this bias. In Tab. 3.2 a comparison of the neural network estimates to the results of classical guesses is presented.

Table 3.2.: Estimates for the HF energy for the neural network guess denoted by NN and classical schemes. All values are in mE_h

$\mathrm{H_{core}}$	SAD	GWH	NN
$17\,000 \pm 1500$	590 ± 340	700 ± 140	11 ± 9

Not only are no side effects visible, but the error of the guess of the ANN is almost two orders of magnitudes smaller! This means that the difference in quality regarding the different matrix elements is not a problem as the overall error is small enough.

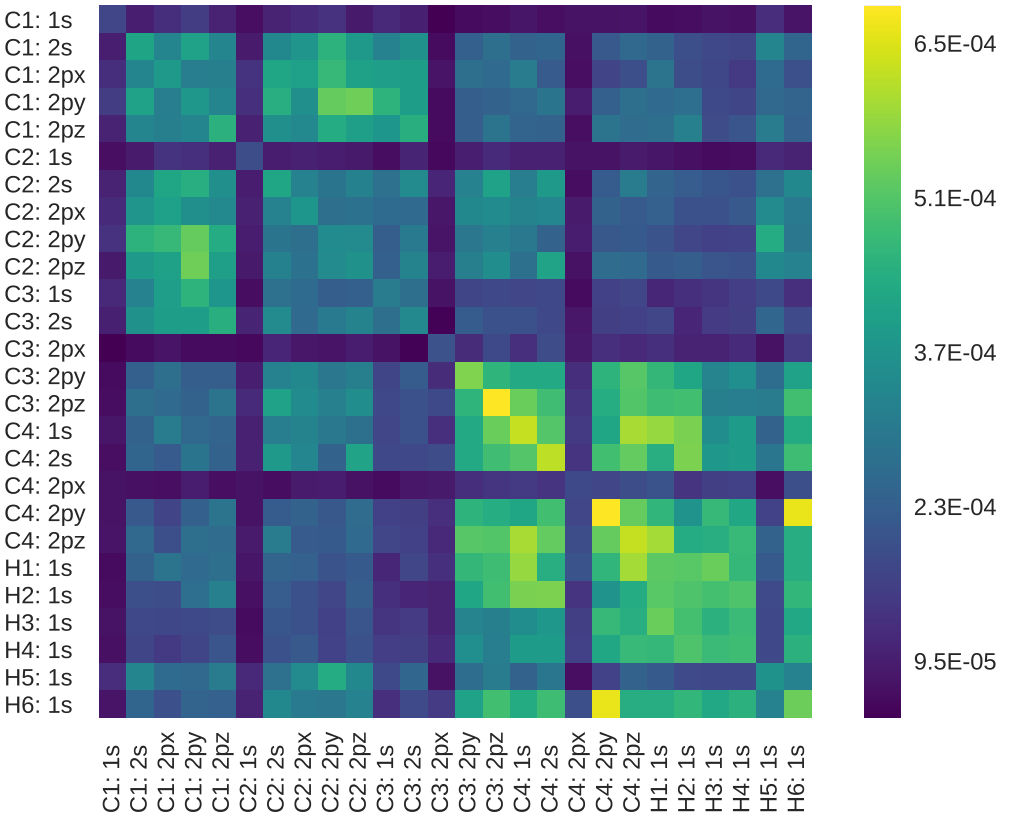


Figure 3.8.: A heatmap plot of the average error of the neural network prediction for each element of the density matrix.

This is a promising start. We will now see how this affects the SCF calculation. Tab. 3.3 lists the iterations which are required by an SCF calculation to converge. Of course, not all samples will converge for all guesses, which is why the percentage of samples which exceeded the maximum number of allowed iterations (which was 100 iterations) are stated as well. The average number of iterations are calculated using only the results from the converged samples, as the ones that did not converge would contribute with the maximum number of allowed iterations and distort the result. We find improvements of up to half the number of iterations (Pure) and almost a third for DIIS. Also, regarding the number of samples that did not converge, NN is clearly taking the lead.

Table 3.3.: Average number of iterations needed to converge the calculation for the cases of no convergence enhancement (Pure), damping (Damped) and with DIIS (DIIS) are shown for H_{core}, SAD, GWH and the neural network guess (NN). Also shown is the percentage of samples that did not converge for the respective combination of enhancement method and initial guess scheme "Not Conv.". For the H_{core} guess no samples could reach convergence with Pure or Damped, which is why no values are listed.

Method	**Quantity**	$\mathbf{H_{core}}$	**GWH**	**SAD**	**NN**
Pure	Iterations / 1	-	37 ± 14	39 ± 25	21 ± 20
	Not Conv. / %	100.0	13.4	10.9	0.0
Damped	Iterations / 1	-	28 ± 11	23 ± 10	10 ± 3
	Not Conv. / %	100.0	2.9	0.5	0.0
DIIS	Iterations / 1	15 ± 2	12 ± 1	11 ± 1	7 ± 1
	Not Conv. / %	0.0	0.0	0.0	0.0

In Fig. 3.9 the results from Tab. 3.3 are presented graphically. We can see the number of samples that could not reach convergence, which is largest for Pure calculations (blue bars).

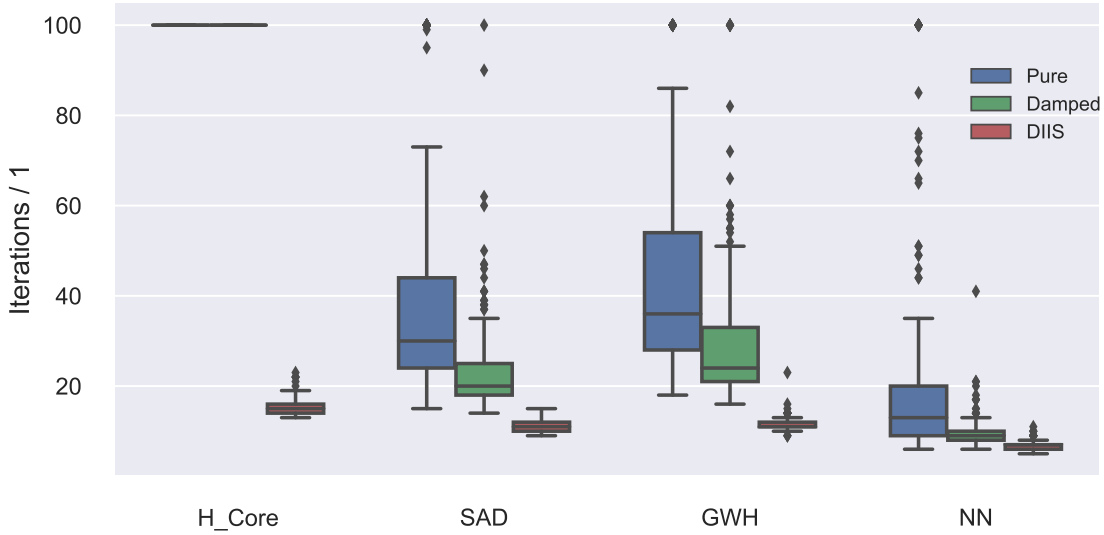


Figure 3.9.: The iterations needed to reach convergence for different guess schemes and convergence enhancement methods.

3.3. Improving the Network Guess

We want to improve our scheme even further. In Section 2.1.7, a few attributes of the density matrix were introduced. Most notably, we talked about the number of electrons

and charge population analysis and the idempotence with respect to the overlap matrix. We will investigate as to how far our neural network guess already fulfills these constraints. With the network from Section 3.2.2 we receive the results listed in Tab. 3.4.

Table 3.4.: Orders of errors of attributes of the neural network guess and the classical guesses in comparison to the converged result over the test dataset. Reference denotes converged density matrix.

	Idempotence	**Occupancy**
H_{core}	$\mathcal{O}(10^{-16})$	$\mathcal{O}(10^{-}15)$
SAD	$\mathcal{O}(10^{0})$	$\mathcal{O}(10^{-15})$
GWH	$\mathcal{O}(10^{-16})$	$\mathcal{O}(10^{-15})$
Reference	$\mathcal{O}(10^{-16})$	$\mathcal{O}(10^{-15})$
NN	$\mathcal{O}(10^{-4})$	$\mathcal{O}(10^{-3})$

This is our starting point. We see that the converged matrices (as well as most of the classical guesses) show very little error, while our NN guess performs rather poorly. An exception is the SAD guess, which is non-idempotent by construction (cf. Section 2.1.9). It might be helpful for us to improve on these attributes. We will do this using matrix transformations, which are applied in a post-processing step.

3.3.1. Improving the Idempotence

We will start with idempotence as this should be easy to fix, due to a matrix transformation proposed by McWheeny[40]. It is generally referred to as the McWheeny purification. The idea is to transform a matrix $\boldsymbol{A}$ to obtain a more idempotent matrix

$$\boldsymbol{A}' = 3\boldsymbol{A}^2 - 2\boldsymbol{A}^3, \tag{3.1}$$

where $\boldsymbol{A}$ and $\boldsymbol{A}'$ are represented in an orthonormal basis, a luxury we unfortunately do not have. For our application the transformation thus becomes

$$\boldsymbol{P}' = \frac{3}{2}\boldsymbol{PSP} - \frac{1}{2}\boldsymbol{PSPSP}. \tag{3.2}$$

The purification is an iterative process and must be applied repeatedly, generating a more and more idempotent matrix (i.e. exhibiting a smaller idempotence error) every time. We will apply the transformation once (McW-1) and five times (McW-5) to see the gradual effects. This leaves us with the values listed in Tab. 3.5. Interestingly, not only the idempotence is improved, but also the HF energy error and even the absolute error of the guess.

Table 3.5.: Properties of the neural network guess (NN) and its transformed versions (McWheeny purification applied once, "McW-1", and five times, "McW-5"). MAE denotes the mean absolute error of the matrix elements. "Not Conv. (Pure)" denotes the percentage of not converged calculations without any convergence enhancements (Pure). The iterations for the Pure case are given without non-converged samples. In the calculations with damping and DIIS all samples converged for all guesses.

	NN	**McW-1**	**McW-5**
MAE / 1×10^{-4}	3 ± 1	2 ± 1	2 ± 1
HF energy error / mE_h	11 ± 9	0.13 ± 0.07	0.010 ± 0.008
Idempotence / 1	$(5 \pm 1) \times 10^{-4}$	$(2 \pm 1) \times 10^{-6}$	$(7.6 \pm 0.6) \times 10^{-17}$
Occupancy / 1	$(4 \pm 3) \times 10^{-3}$	$(1.0 \pm 0.7) \times 10^{-4}$	$(2 \pm 1) \times 10^{-15}$
Iterations (Pure) / 1	17 ± 14	13 ± 10	13 ± 10
Not Conv. (Pure) / %	5.0	3.5	3.5
Iterations (Damped) / 1	10 ± 3	9 ± 3	9 ± 3
Iterations (DIIS) / 1	7 ± 1	7 ± 1	7 ± 1

After only applying the purification once, accuracy in the values and the HF energy calculated from the matrices and prediction of the total number of electrons increases. After applying it five times, the idempotence is approximately on the same level as the converged density matrix. Most notable is the incredibly small error in HF energy. Regarding iterations, unfortunately only a small advantage could be gained by the purification. For the cases done with DIIS no improvement is visible at all. Therefore, we must conclude that a McWheeny purification does improve the guess, but only to a rather small degree.

3.3.2. Improving the Charge Analysis

Next, we will target the ability of the guess to reproduce the correct number of electrons. Due to the linearity of the trace,

$$c \operatorname{tr}(\boldsymbol{P}\boldsymbol{S}) = \operatorname{tr}((c\boldsymbol{P})\boldsymbol{S}) \qquad \forall c \in \mathbb{R}, \tag{3.3}$$

we can rescale the density matrix guesses to yield exactly the number of electrons in the charge population analysis. We define our rescaling transformation,

$$\boldsymbol{P}' = \boldsymbol{P} \frac{N}{\operatorname{tr}(\boldsymbol{P}\boldsymbol{S})}, \tag{3.4}$$

and apply it to the output of the ANN. It must be noted that we can gain at most a single iteration, because after the first SCF step the occupation should be of the order of the reference. The results are listed in Tab. 3.6. While the prediction of the number of electrons was improved drastically all other properties remained widely unchanged. However, the number of iterations in the Pure case improved slightly. Unfortunately, at the same time, the percentage of not converged samples increased, which leads us to believe that this

improvement is just a consequence of a few difficult geometries not reaching converge and thus falling out of the statistics.

Table 3.6.: Properties of the neural network guess (NN) and its rescaled version. MAE denotes the mean absolute error of the matrix elements. Not Conv. (Pure) denotes the percentage of not converged calculations without any convergence enhancements. The iterations given for the Pure case are given without non-converged samples. In the calculations with damping and DIIS all samples converged.

	NN	**Rescaled**
MAE / 1×10^{-4}	3 ± 1	3 ± 1
HF energy error / mE_h	11 ± 9	7 ± 6
Idempotence / 1	$(5 \pm 1) \times 10^{-4}$	$(5 \pm 1) \times 10^{-4}$
Occupancy / 1	$(4 \pm 3) \times 10^{-3}$	$(3 \pm 2) \times 10^{-15}$
Iterations (Pure) / 1	17 ± 14	16 ± 13
Not Conv. (Pure) / %	5.0	8.9
Iterations (Damped) / 1	10 ± 3	10 ± 3
Iterations (DIIS) / 1	7 ± 1	7 ± 1

3.4. Conclusion

In this chapter we estimated the density matrix of butadiene molecules using their overlap matrix as input. Not only did we manage to produce a very robust and accurate guess, we also improved it using the McWheeny transformation. The NN guess and its improved versions McW-1 and McW-5 performed far superior to the classical guess schemes.

On the other hand the presented scheme is very rigid. It can only produce guesses for a single molecule in a single basis set. Therefore, a few substantial modifications must be introduced to make the NN guess applicable to arbitrary systems.

4. Generic Descriptor Schemes

4.1. Introduction

In Chapter 3 we used the overlap matrix as input for our neural network. This "descriptor scheme" was computationally cheap and delivered the required information, but was also very rigid. Now we will work on making the input side of the network more flexible. The goal is to derive a descriptor scheme that describes the situation of every atom in a molecule independently, thus being able to perform independently of the actual molecule composition. (In other words: it must not have any restrictions regarding molecular composition.) The description of a molecule will be assembled from the descriptions of the respective environments of the atoms in such a scheme.

We need to make sure that the scheme to be derived fulfills our requirements (e.g. regarding symmetry and invariances) but is still able to encode all relevant information. As mentioned in Section 2.3.4, estimating non-scalar quantities is still mostly unexplored territory. Thus, we need to define a few physical ground rules. First, we will define our requirements for our descriptor scheme:

- It should be able to distinguish different elements. Oxygen will no doubt have a different impact on the electron density than a helium atom would in the same position.
- It must to be able to distinguish close and distant atoms, i.e. be sensitive to bond lengths and geometry in general.
- The description should be independent of the order the atoms in the geometry specification of the molecule are given in (see also Section 2.3.4).
- As mentioned above, it should remain applicable to any molecule, regardless of its composition.

Second, we will consider certain invariances of our preferred descriptor scheme:

- Translational invariance: The absolute coordinates of the atoms are irrelevant; only their relative positions are important.
- A fixed orientation in space: This is a direct consequence of the fact that basis functions (Section 2.2) have a fixed orientation. Therefore, all matrices in this basis (such as e.g. the density matrix) will be different for differently oriented, yet identical molecules. Thus, it will be essential for our descriptor to capture how a sample is positioned in space.

4.1.1. The Scheme

The starting point is the vector from atom i to atom j,

$$\boldsymbol{R}_{ij} = \boldsymbol{R}_i - \boldsymbol{R}_j,$$

where $\boldsymbol{R}_i$ denotes the position of atom i in spherical coordinates. Our descriptor for the environment of atom i, denoted by $\boldsymbol{G}_i$, will be composed of atomic contributions of all atoms $j \neq i$, denoted as

$$\boldsymbol{G}_{ij} = \boldsymbol{S}(\boldsymbol{R}_{ij}) \tag{4.1}$$

with $\boldsymbol{S} = (\{S_k\})^\top$ as the vector of symmetry functions S_k, where k is the index of the element or the symmetry vector corresponding to the symmetry function and does not refer to an atom in the molecule. They are combined in a weighted sum,

$$\boldsymbol{G}_i = \sum_j w_{ij}(\boldsymbol{G}_{ij}), \tag{4.2}$$

with w_{ij} denoting a weighting function to be defined later. These atomic contributions should each for themselves be an encoding of the position of atom j with respect to atom i, as well as additional physical information such as which chemical element atom j corresponds to (this is the purpose of w_{ij}). The role of the symmetry functions in this context is to yield a description of the local environment of atom i in a compact and (in combination with the weighting) unique way. Unique in this context refers to the fact that a descriptor must produce different outputs for different molecules[1], thus making it possible to distinguish them. However, it is not required that molecules can be identified by the descriptor output. If two different molecules would exhibit the same density matrix (regardless of whether this is possible at all), may result in the same descriptor output. The resulting process is depicted in Fig. 4.1.

The goal of this chapter is to find suitable sampling functions S_k. A set of such functions will be called a "descriptor model". For simplicity, we will now drop the atom indices in e.g. $\boldsymbol{R}_{ij} \to \boldsymbol{R}$, as our discussions in the next pages will focus on an abstract pair of atoms, i-j. We must encode $\boldsymbol{R}$ in a way that allows us to combine it with the contribution from the other atoms in a simple way. In analogy to Behler and Parinello[36], we will express $\boldsymbol{R}$ in spherical coordinates as

$$\boldsymbol{R} =: (r, \varphi, \theta)^\top \tag{4.3}$$

and sample its components with our descriptor functions. The difference is, however, that the symmetry functions used in this project show far less symmetries. While the radial part is mostly the same, the difference lies in the angular part. The descriptors used by Behler and Parinello sample the angles between three atoms[2]. We will use the azimuthal and the polar components, φ and θ, of the vector $\boldsymbol{R}$ instead. One could say that Behler and Parinello sample internal coordinates, which ensure all the required symmetries, while we use a mixture of internal (atom distances) and spatially fixed (angles) coordinates. This mixture preserves some symmetries but does not fully decouple our description from our reference coordinate system.

[1] Different in the sense that these molecules have differing density matrices.

[2] An example of such an angle is "HCH" in Fig. 4.2. The time evolution of the angles between nearest neighbor atoms of a butadiene molecule during an MD simulation can be seen in Fig. 3.3b.

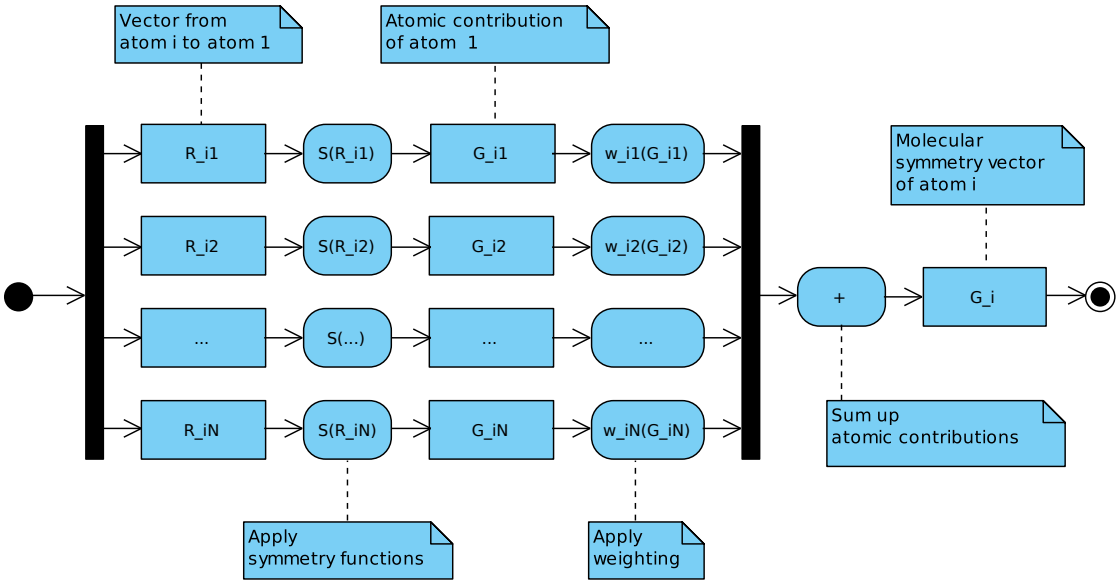


Figure 4.1.: Schematic illustration of the calculation of the symmetry vector that describes the environment of atom i in the molecule. For relative positions, atom contributions are calculated via symmetry and weighting functions, which are finally added up to yield the symmetry vector.

4.1.2. The Dataset

In this chapter we will illustrate a few things using the small olefin ethene. While being similar to butadiene, it is more convenient due to its smaller size. It is depicted in Fig. 4.2.

4.2. Radial Description

In this section, we will discuss how to sample r, the radial component of the vector $\boldsymbol{R}$ (i.e. $r = \|\boldsymbol{R}\|$) that connects the two atoms of the pair (i, j). Our tool of choice will be Gaussian functions of varying width η,

$$f_{\text{Gaussian}}(r, \eta, r_\text{s}) = \mathrm{e}^{-\eta(r-r_\text{s})^2}, \tag{4.4}$$

centered at r_s. Behler and Parinello used Gaussians centered around zero[42] as well as models with distributed Gaussians[36]. We will also try other models for reasons explained below.

4.2.1. Activation

We will now introduce a practical tool for the visualization of our models: activation. The aim is to extract the information encoded in a symmetry vector and visualize it. This way

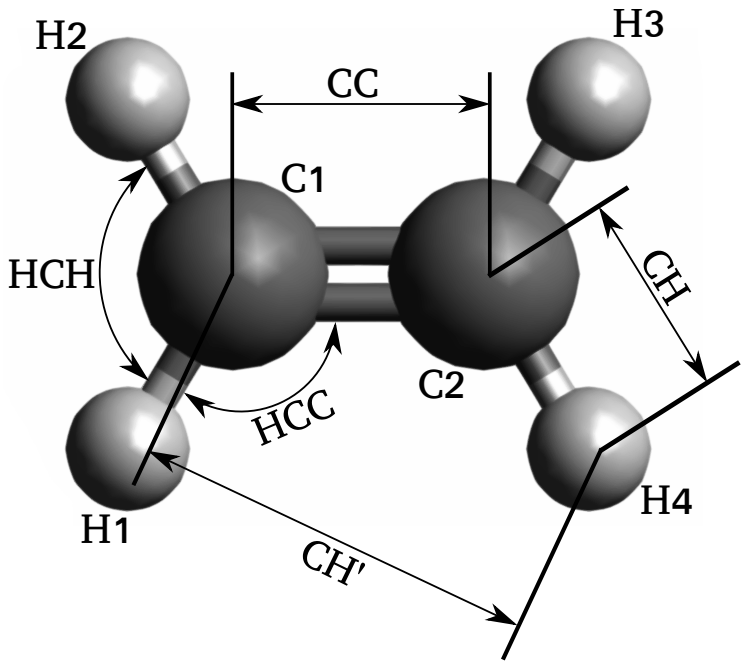


Figure 4.2.: Ethene molecule near optimum geometry. A few characteristic distances and angles are marked in the drawing. Their values are approximately "CC" 1.33 Å, "CH" 1.08 Å, "CH′" 2.11 Å (distances) and "HCC" 121.4°, "HCH" 117.2° for an ethene molecule in equilibrium geometry[41].

we can limit the parameter space for a more systematic parameter optimization and make sure that the information relevant to us is actually captured. For a given symmetry vector $\boldsymbol{G}$ with the elements G_i we can retain the encoded information by using it as weighting for the corresponding symmetry functions S_i which are evaluated at (r, φ, θ). This way we obtain an "activation"

$$a(r, \varphi, \theta, \boldsymbol{G}) = \sum_i G_i S_i(r, \varphi, \theta). \tag{4.5}$$

If we restrict the symmetry functions and vector values to those that correspond to radial description, i.e. to all $i \in I$, with I the set of indices that correspond to radial descriptors, we can obtain a radial activation

$$a_{\text{radial}}(r, \boldsymbol{G}) = \sum_{i \in I} G_i S_i(r). \tag{4.6}$$

In Fig. 4.3 the radial activation of a few descriptor models for the C1 atom is plotted, together with the typical lengths of the system: the C-C double bond bonding length ("CC"), the C-H bonding length ("CH") and even a peak for the remote H atoms ("CH′"), which are naturally less pronounced due to the cut-off (see Section 4.4). The values are also marked in Fig. 4.2. Included in the test is a model containing 50 Gaussians, all centered at zero (Origin-50), several models of uniformly distributed Gaussians (Unif-N, where N is the number of Gaussians) and a manually tweaked model of 50 Gaussians, with

higher densities in the areas of the characteristic distances (Man-50). All these models are presented in detail in Appendix B.1. As can be seen in Fig. 4.3, the uniformly distributed models have difficulties capturing the characteristic distances with fewer functions, while the Man-50 does without effort. This is obviously not very surprising as it has many sampling functions in this area and can thus sample with a very high resolution. Origin-50 is not able to resolve any of the distances in a way that would be visible for a human spectator. However, this is also the catch with this tool: activation allows for a quick

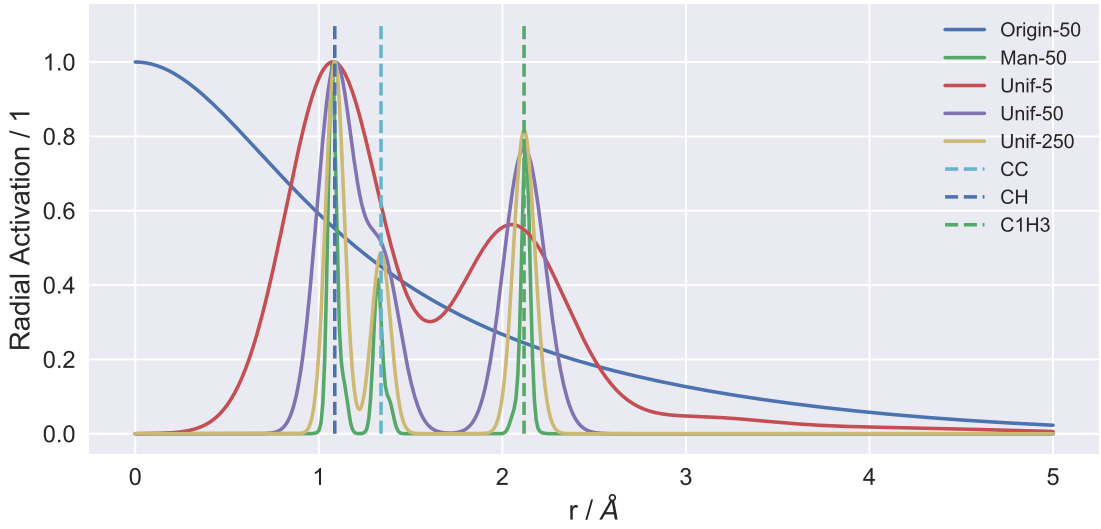


Figure 4.3.: The radial activation of the description of atom C1 in ethene for a few descriptor models together with characteristic distances of the system. See Fig. 4.2 for the distances and Appendix B.1 for details on the models.

visualization, but only gives a rough idea of how information is encoded. A neural network can read much more out of the symmetry vectors and, even if no physical information could be reconstructed, as is the case with Origin-50, the network might still be able to draw sufficient information to make meaningful and accurate predictions. It merely shows where the emphasis of a descriptor model lies. This leads us to the second lesson to be learned here: the number of descriptor functions needed for a required accuracy can be significantly reduced if they are fine-tuned to sample in relevant regions.

4.3. Angular Description

Now we will encode the angular component of $\boldsymbol{R}$, i.e. the azimuthal angle φ and the polar angle θ. As opposed to the radial component, they need a description that is periodic. Plain Gaussians will therefore not work. We suggest two possible solutions: a periodic adaption of Gaussians (called periodic Gaussians), in which polar and azimuthal angle are sampled separately, and spherical harmonics (which of course already have the desired

periodicity). A periodic Gaussian takes the form

$$g_{\text{periodic}}(\varphi, \varphi_s, \eta, p) := \max\left\{ \mathrm{e}^{-\eta((\varphi-\varphi_s)(\text{mod } p))^2}, \mathrm{e}^{-\eta((\varphi-\varphi_s)(\text{mod } p)-p)^2} \right\} \tag{4.7}$$

and is centered around φ_s, with a width defined by η and a periodicity of p. The function "mod" denotes the modulo operation extended to real numbers, e.g. for $x, y \in \mathbb{R}$:

$$x(\text{mod } y) = x - k * y, \qquad k = \text{floor}(x/y).$$

(Even though φ and φ_s were used in (4.7) this function is of course also used to sample the polar angle.) Note that the "Gaussian form" can only be realized if the width η of the Gaussians involved is large enough in comparison to the period. Their activation is calculated analogously to the activation of non-periodic Gaussians.

For the spherical harmonics, the real and the imaginary parts of the complex values are used. We define

$$g_{\text{SPH}}(\varphi, \theta, l, m, \sigma) := \begin{cases} \Re(\mathrm{Y}_l^m(\varphi, \theta)) & \text{if } \sigma = 1 \\ \Im(\mathrm{Y}_l^m(\varphi, \theta)) & \text{if } \sigma = -1 \end{cases} \tag{4.8}$$

In a symmetry vector that was created using spherical harmonics for the angular part, two values at a time will correspond to the same l and m: one with $\sigma = +1$ and the other with $\sigma = -1$. The activation is calculated by taking the absolute value of the complex number that would result from combining the corresponding values of the symmetry vector again:

$$a_{\text{SPH}}(\varphi, \theta, \boldsymbol{G}) = \sum_{(i,j)\in I} |G_i S_i(\varphi, \theta) + \mathrm{i} G_j S_j(\varphi, \theta)|, \tag{4.9}$$

with I denoting the family of pairs of indices that belong to the angular part of the symmetry vector and correspond to the symmetry function with matching l and m but differing σ. In Fig. 4.4, the azimuthal activation resulting from ethene for a few angular models is shown, for a fixed polar angle. The models compared are two spherical harmonics models SPH-N (where N denotes the highest l to be used) and PG-M which denotes a model of M evenly distributed periodic Gaussians. The peak at $\varphi = 0$ may be confusing at first, but is merely a result of the peak at $\varphi = 2\pi$ and the periodic nature of the symmetry functions. Once again, it can be seen that greater resolution can be achieved with more functions.

4.4. Weighting and Cut-Off

Now that an atomic contribution $\boldsymbol{G}_{ij}$ can be calculated, we need to specify how to combine them and thus calculate the weighting function w_{ij}. The weighting is a great opportunity to include physical information such as mass, charge etc. of atom j in relation to the respective quantities of atom i.

Also, electrostatics should not be neglected; taking into account a dependence on the distance is probably advantageous. This is done via so called cut-off functions as used by

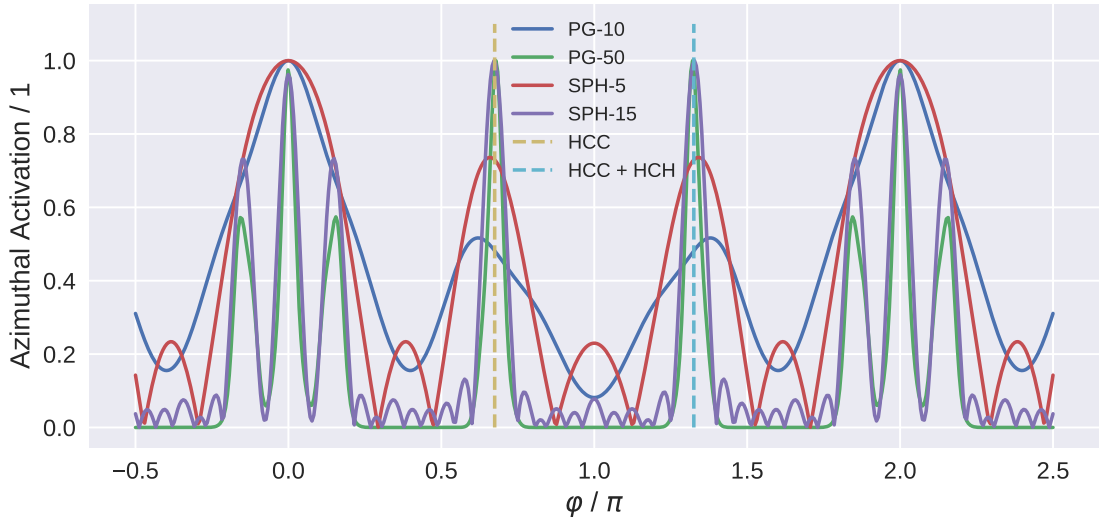


Figure 4.4.: The azimuthal activation for ethene in the x-y plane, together with characteristic angles (cf. Fig. 4.2). θ is fixed at $\frac{\pi}{2}$.

Behler and Parinello[36]. We will employ one of the presented functions, which we shall call "Behler1"[3] in this work. It is of the form

$$f_{\text{Behler1}}(r) = \begin{cases} \frac{1}{2}\left[\cos\left(\frac{\pi r}{R_\text{c}}\right) + 1\right] & r \leq R_\text{c} \\ 0 & r > R_\text{c} \end{cases}, \tag{4.10}$$

with the cut-off distance R_c. We will also evaluate the feasibility of a simple exponential damping

$$f_{\text{exponential}}(r) = \mathrm{e}^{-\frac{r}{\tau}}, \tag{4.11}$$

where τ denotes the half-life, taking to role of the cut-off distance. The two candidate cut-off functions are shown in Fig. 4.5. With the first cut-off function, as it goes to zero at and beyond the cut-off distance R_c, linear scaling in the calculation of symmetry functions can be reached for sufficiently large molecules, because atom pairs (ij) with a distance $r > R_\text{c}$ can simply be neglected as they do not yield any contribution to the atomic descriptor. Note, however, that this is not possible for (4.11).

We will choose the weighting functions to be a product of a physical quantity W_j of atom j and a cut-off function f:

$$w_{ij}(\boldsymbol{R}_{ij}) = W_j f(\|\boldsymbol{R}_{ij}\|). \tag{4.12}$$

The reason for not using e.g. the ratio of the property $\frac{W_j}{W_i}$ is that it W_i would be the same for all w_{ij} and thus not make a difference. In Fig. 4.6, the radial activation of the C1 atom is shown for a model consisting of uniformly distributed Gaussians, weighted by various physical quantities.

[3] Following a similar naming scheme used by Behler a review article[42].

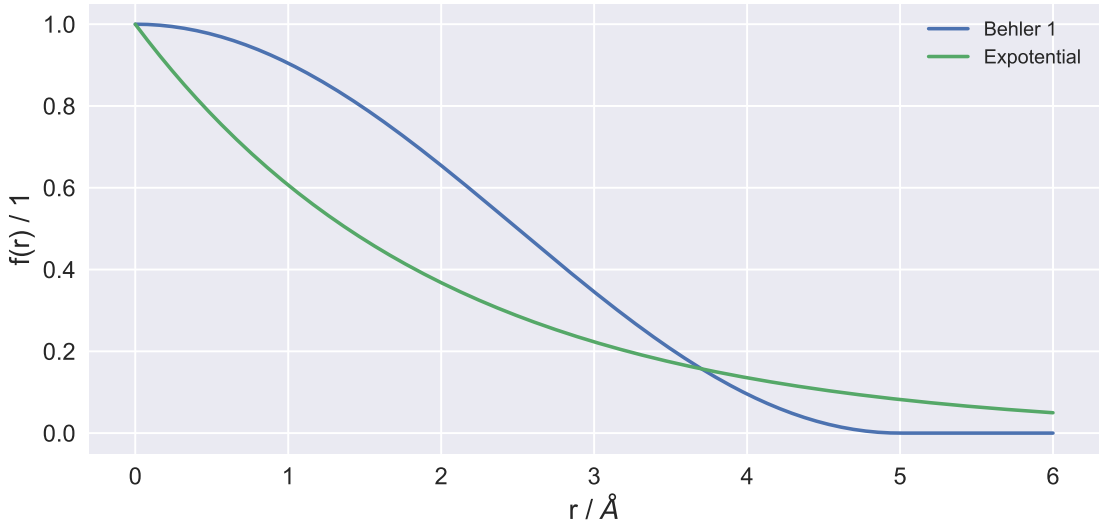


Figure 4.5.: The cut-off functions analyzed in this project. For Behler1 a cut-off distance of 5 Å and for the exponential decay a half-life of 2 Å.

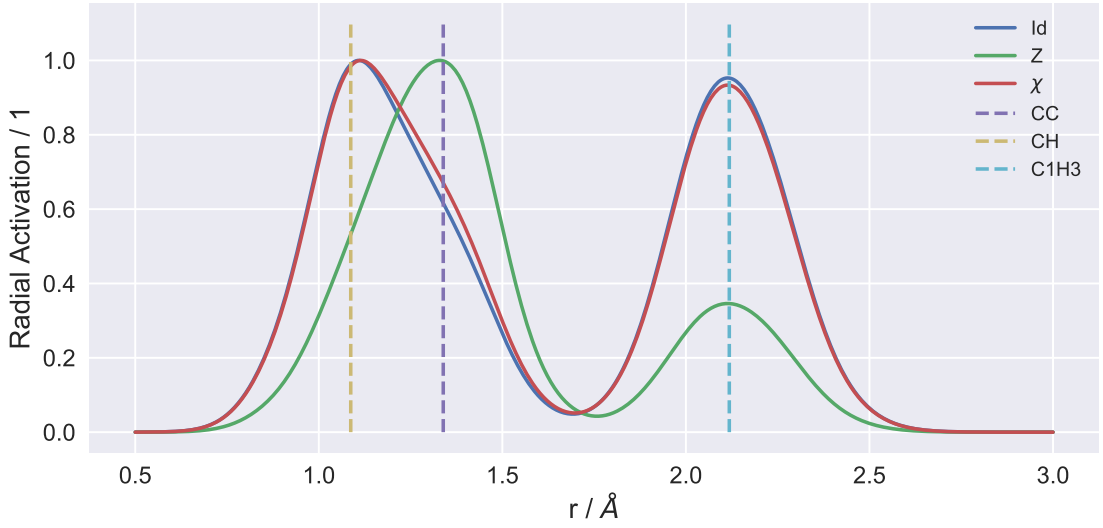


Figure 4.6.: Radial activation of the C1 atom in an ethene molecule, for the radial model Unif-25 (cf. Appendix B.1.3) in connection with different quantities used for weighting. For a better comparison, no cut-off was applied. The curves are all rescaled to their respective maximum value. The quantities used are the atomic number Z, the electronegativity χ, and a factor of 1 (i.e. no weighting factor applied) denoted by "Id".

We observe that using the electronegativity has a subtle effect at best, and is very close to the non-weighted activation ("Id"). Neither χ nor Id seem to promote distinguishability of the atom in molecule. Using the atomic number Z, on the other hand, really brings out

the CC peak, i.e. emphasizes the C2 carbon. This is probably a favorable behavior.

Finally, to give an impression of the effect of the weighting function, a set of Gaussians sampling the distance r, multiplied by a cut-off function, is shown in Fig. 4.7. It is worth noting that the direction of the cut-off and the direction of the sampling fall together, yielding this special result. For the angles, the cut-off is perpendicular, and plotting them the same way would make no sense.

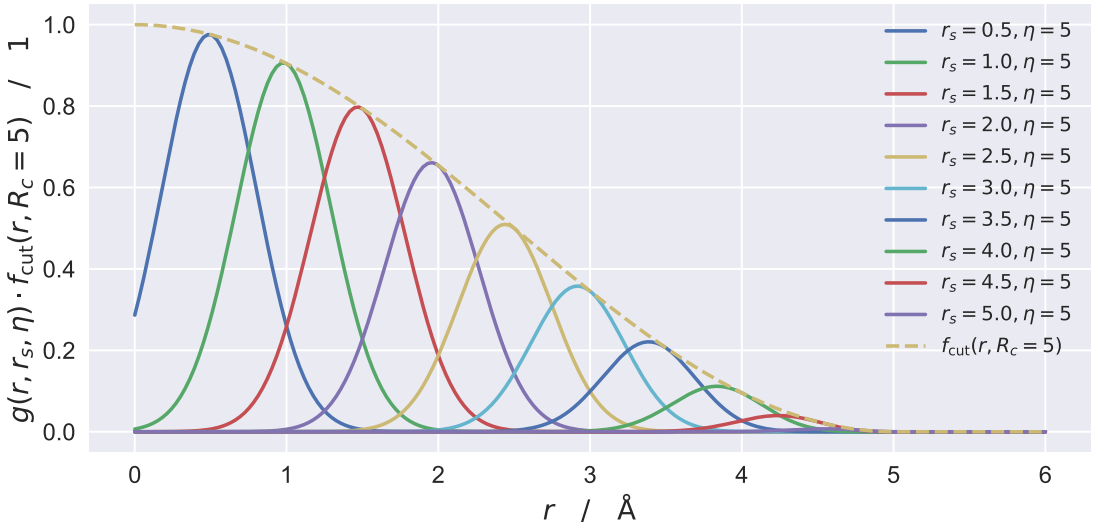


Figure 4.7.: Gaussians (denoted by $g(r, r_s, \eta)$) centered at r_s with a width of $(2 * \eta)$, weighted by the cut-off function Behler1 (denoted in the plot as f_{cut}), as radial part of descriptor values would be.

4.5. Raw Selection

Now that we have gathered a few candidate descriptor models, we shall do a quick parameter study to assess which of them are the most suitable. Our model task will be to estimate the full density matrix of ethene. As input we will calculate descriptors for every atom in the molecule and concatenate them to an input that describes the whole molecule. We will train a single layer (i.e. linear) ANN for every combination of candidates for radial and angular descriptor models, cut-off function, and physical quantity W to be used in the weighting and compare the reached training and validation errors. Each network is initialized and trained independently five times to account for possible disadvantages resulting from an unfortunate initialization or slow learning behavior.

4.5.1. Dataset

Similarly to Chapter 3, the geometries of the dataset for this part of the thesis were also created via an MD simulation with QChem[10]. Again, 1001 geometries were sampled with 1000 time steps of 0.5 fs at 450 K. This resulting energy distribution of the dataset is shown in Fig. 4.8. For each of the sampled geometries the Fock and the density matrix

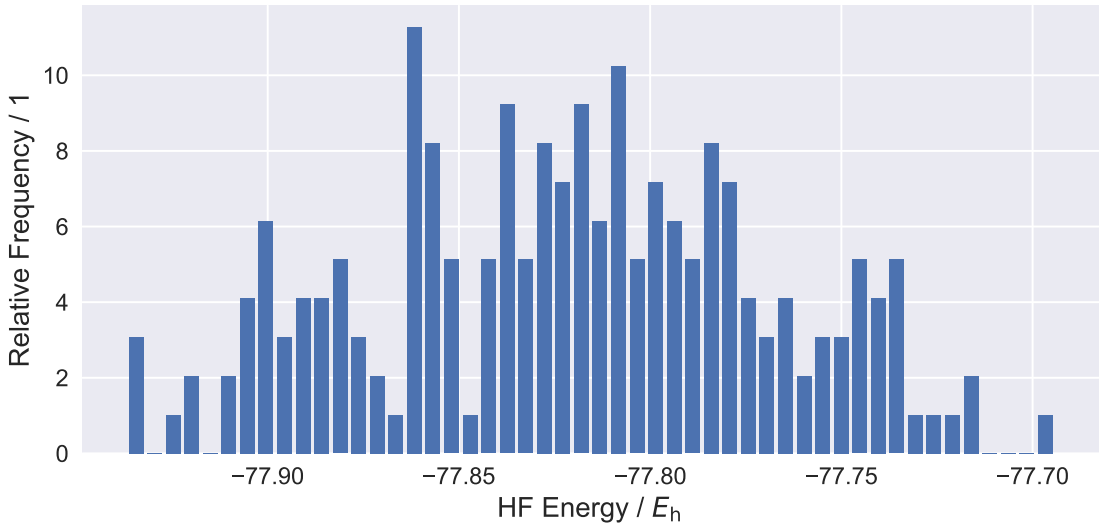


Figure 4.8.: Distribution of energies in the dataset. The lowest and the highest energies lie about $0.2\,E_h \approx 5.5\,\text{eV}$ apart.

are calculated in the 6-311++G** basis[43]. 80 % of the samples were used for training, the remaining make up the validation dataset.

4.5.2. Descriptor Model Candidates

Our descriptor models will be composed of models for the radial part:

"Man-50" 50 Gaussians with manually tweaked positioning and width. They also become wider the further away from 0 they are, but have the highest density in the range 0.7-2.5 Å, where we would expect most of the interaction.

"Unif-N" N uniformly distributed (regarding centers and widths) Gaussians. They become broader with increasing distance from zero.

For the angular part we will use the following selection:

"PG" Periodic Gaussians for azimuthal and polar angles, both uniformly distributed. 20 Gaussians for azimuthal and 10 for the polar part.

"SPH-3" The real and imaginary part of spherical harmonics, up to a maximum angular momentum number of $l_{\text{max}} = 3$.

"SPH-6" Similar to SPH-3, but including spherical harmonics up to a maximum angular momentum number of $l_{\max} = 6$.

The models using Gaussians or periodic Gaussians are depicted in Appendix B.1. The candidates for quantities used as weight W in the weighting function are:

"Id" No specific weighting at all. $\forall j : W_j = 1$.

"Z" The atomic number Z_j.

"χ" The electronegativity χ_j.

Finally, the cut-off will be modeled by the two functions shown in Fig. 4.5: Behler1 (with a cut-off distance of 2.5 and 5 Å) and the exponential decay "Exponential" (with a parameter of 2 Å).

4.5.3. Results

The detailed composition of the models is listed in Tab. 4.1 and the result of the comparison is shown in Fig. 4.9.

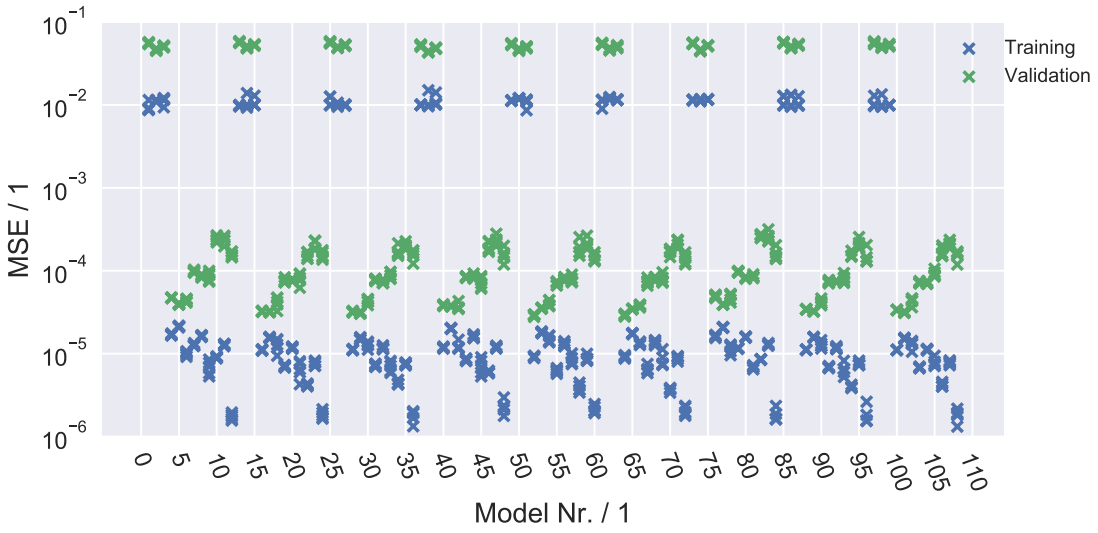


Figure 4.9.: Comparison of different descriptors as inputs for a linear ANN. The figure shows training and validation mean square error. The detailed configuration of the models is shown in Tab. 4.1.

The following conclusions can be drawn:

- The weighting seems to have little influence. This seems to be true both for constants used as well as the choice of cut-off functions. However, since only two types of atoms appear in the dataset, we cannot be sure if this is also the case for systems with more

elements. From what can be seen here, there is a very small edge in favour of χ, but the difference is too small to be statistically relevant.

- The radial descriptor definitely makes a difference. Surprisingly, the hand-crafted candidate, Man-50, performed worst. Apparently, there was a lot of relevant information at greater distances. We also note that for the uniform model Unif-N the lowest training errors are achieved with the largest N, while the opposite is true for the validation error. This raises the suspicion that an important decisive factor is not how well information is captured, but rather the additional number of parameters, resulting from the larger input. In other words, Unif-N, while proving to be superior to Man-50, is prone to over-fitting for large N, which can arguably be seen already for Unif-25, but definitely for Unif-50. The boundary between "too large" and "large enough" will surely vary from atom to atom.
- The largest impact, as is seems, had the angular description. Here, SPH-6 Exhibited some drastic advantages over SPH-3, which in turn performed slightly worse than the periodic Gaussian model. This is surprising, as the PG model should have been large enough to capture the environment very well, while not being larger than the SPH-3.

Table 4.1.: List of the descriptor model configurations tested in this Section.

Model Nr.	**Weighting**	**Cut-Off**	**Radial**	**Angular**
1	Id	Behler1(2.5)	Man-50	PG
2	Id	Behler1(2.5)	Man-50	SPH-3
3	Id	Behler1(2.5)	Man-50	SPH-6
4	Id	Behler1(2.5)	Unif-5	PG
5	Id	Behler1(2.5)	Unif-5	SPH-3
6	Id	Behler1(2.5)	Unif-5	SPH-6
7	Id	Behler1(2.5)	Unif-25	PG
8	Id	Behler1(2.5)	Unif-25	SPH-3
9	Id	Behler1(2.5)	Unif-25	SPH-6
10	Id	Behler1(2.5)	Unif-50	PG
11	Id	Behler1(2.5)	Unif-50	SPH-3
12	Id	Behler1(2.5)	Unif-50	SPH-6
13	Id	Behler1(5.0)	Man-50	PG
14	Id	Behler1(5.0)	Man-50	SPH-3
15	Id	Behler1(5.0)	Man-50	SPH-6
16	Id	Behler1(5.0)	Unif-5	PG
17	Id	Behler1(5.0)	Unif-5	SPH-3
18	Id	Behler1(5.0)	Unif-5	SPH-6
19	Id	Behler1(5.0)	Unif-25	PG
20	Id	Behler1(5.0)	Unif-25	SPH-3
21	Id	Behler1(5.0)	Unif-25	SPH-6
22	Id	Behler1(5.0)	Unif-50	PG
23	Id	Behler1(5.0)	Unif-50	SPH-3

Model Nr.	Weighting	Cut-Off	Radial	Angular
24	Id	Behler1(5.0)	Unif-50	SPH-6
25	Id	Exponential(2.0)	Man-50	PG
26	Id	Exponential(2.0)	Man-50	SPH-3
27	Id	Exponential(2.0)	Man-50	SPH-6
28	Id	Exponential(2.0)	Unif-5	PG
29	Id	Exponential(2.0)	Unif-5	SPH-3
30	Id	Exponential(2.0)	Unif-5	SPH-6
31	Id	Exponential(2.0)	Unif-25	PG
32	Id	Exponential(2.0)	Unif-25	SPH-3
33	Id	Exponential(2.0)	Unif-25	SPH-6
34	Id	Exponential(2.0)	Unif-50	PG
35	Id	Exponential(2.0)	Unif-50	SPH-3
36	Id	Exponential(2.0)	Unif-50	SPH-6
37	Z	Behler1(2.5)	Man-50	PG
38	Z	Behler1(2.5)	Man-50	SPH-3
39	Z	Behler1(2.5)	Man-50	SPH-6
40	Z	Behler1(2.5)	Unif-5	PG
41	Z	Behler1(2.5)	Unif-5	SPH-3
42	Z	Behler1(2.5)	Unif-5	SPH-6
43	Z	Behler1(2.5)	Unif-25	PG
44	Z	Behler1(2.5)	Unif-25	SPH-3
45	Z	Behler1(2.5)	Unif-25	SPH-6
46	Z	Behler1(2.5)	Unif-50	PG
47	Z	Behler1(2.5)	Unif-50	SPH-3
48	Z	Behler1(2.5)	Unif-50	SPH-6
49	Z	Behler1(5.0)	Man-50	PG
50	Z	Behler1(5.0)	Man-50	SPH-3
51	Z	Behler1(5.0)	Man-50	SPH-6
52	Z	Behler1(5.0)	Unif-5	PG
53	Z	Behler1(5.0)	Unif-5	SPH-3
54	Z	Behler1(5.0)	Unif-5	SPH-6
55	Z	Behler1(5.0)	Unif-25	PG
56	Z	Behler1(5.0)	Unif-25	SPH-3
57	Z	Behler1(5.0)	Unif-25	SPH-6
58	Z	Behler1(5.0)	Unif-50	PG
59	Z	Behler1(5.0)	Unif-50	SPH-3
60	Z	Behler1(5.0)	Unif-50	SPH-6
61	Z	Exponential(2.0)	Man-50	PG
62	Z	Exponential(2.0)	Man-50	SPH-3
63	Z	Exponential(2.0)	Man-50	SPH-6
64	Z	Exponential(2.0)	Unif-5	PG
65	Z	Exponential(2.0)	Unif-5	SPH-3
66	Z	Exponential(2.0)	Unif-5	SPH-6
67	Z	Exponential(2.0)	Unif-25	PG

Model Nr.	Weighting	Cut-Off	Radial	Angular
68	Z	Exponential(2.0)	Unif-25	SPH-3
69	Z	Exponential(2.0)	Unif-25	SPH-6
70	Z	Exponential(2.0)	Unif-50	PG
71	Z	Exponential(2.0)	Unif-50	SPH-3
72	Z	Exponential(2.0)	Unif-50	SPH-6
73	χ	Behler1(2.5)	Man-50	PG
74	χ	Behler1(2.5)	Man-50	SPH-3
75	χ	Behler1(2.5)	Man-50	SPH-6
76	χ	Behler1(2.5)	Unif-5	PG
77	χ	Behler1(2.5)	Unif-5	SPH-3
78	χ	Behler1(2.5)	Unif-5	SPH-6
79	χ	Behler1(2.5)	Unif-25	PG
80	χ	Behler1(2.5)	Unif-25	SPH-3
81	χ	Behler1(2.5)	Unif-25	SPH-6
82	χ	Behler1(2.5)	Unif-50	PG
83	χ	Behler1(2.5)	Unif-50	SPH-3
84	χ	Behler1(2.5)	Unif-50	SPH-6
85	χ	Behler1(5.0)	Man-50	PG
86	χ	Behler1(5.0)	Man-50	SPH-3
87	χ	Behler1(5.0)	Man-50	SPH-6
88	χ	Behler1(5.0)	Unif-5	PG
89	χ	Behler1(5.0)	Unif-5	SPH-3
90	χ	Behler1(5.0)	Unif-5	SPH-6
91	χ	Behler1(5.0)	Unif-25	PG
92	χ	Behler1(5.0)	Unif-25	SPH-3
93	χ	Behler1(5.0)	Unif-25	SPH-6
94	χ	Behler1(5.0)	Unif-50	PG
95	χ	Behler1(5.0)	Unif-50	SPH-3
96	χ	Behler1(5.0)	Unif-50	SPH-6
97	χ	Exponential(2.0)	Man-50	PG
98	χ	Exponential(2.0)	Man-50	SPH-3
99	χ	Exponential(2.0)	Man-50	SPH-6
100	χ	Exponential(2.0)	Unif-5	PG
101	χ	Exponential(2.0)	Unif-5	SPH-3
102	χ	Exponential(2.0)	Unif-5	SPH-6
103	χ	Exponential(2.0)	Unif-25	PG
104	χ	Exponential(2.0)	Unif-25	SPH-3
105	χ	Exponential(2.0)	Unif-25	SPH-6
106	χ	Exponential(2.0)	Unif-50	PG
107	χ	Exponential(2.0)	Unif-50	SPH-3
108	χ	Exponential(2.0)	Unif-50	SPH-6

4.6. Application to Butadiene

Having decided on a descriptor model, we will apply it now to our system from Chapter 3 to compare the results. As in Section 4.5, the inputs for the ANN will consist of the concatenated symmetry vectors, which are calculated for all atoms in the molecule individually via the respective descriptors. For 100 radial descriptors (Model "Man-50", Fig. B.2) and 98 angular functions (SPH-4) at 10 atoms in the molecule, we end up with an input layer size of 1980 neurons! For the rest of the neural network a structure of three hidden layers (700, 500 and 400 neurons) was selected. Results are presented in Tab. 4.2 and 4.3.

Table 4.2.: Properties of the guess produced by the neural network with the new descriptors as input ("NN-Descriptors") for the test dataset samples from Section 3.2.1 together with the results from Section 3.2.2 "NN" . MAE denotes the mean absolute error of the matrix elements.

	NN	**NN-Descriptors**
MAE / 1×10^{-3}	0.3 ± 0.1	1.0 ± 0.5
HF energy error / mE_h	11 ± 9	60 ± 68
Idempotence / 1×10^{-4}	5 ± 1	15 ± 6
Occupancy / 1×10^{-3}	4 ± 3	17 ± 19

Table 4.3.: Number of iterations required to reach convergence using the descriptor-based approach "NN-Descriptors" for the test dataset samples from Section 3.2.1 together with data of the results from the overlap matrix based approach "NN". The averages for the number of iterations do not contain the results from not converged samples. The percentage of samples that did not converge are listed in the rows labeled "Not Conv.".

Method	**Quantity**	**NN**	**NN-Descriptors**
Pure	Iterations / 1	17 ± 14	22 ± 15
	Not Conv. / %	5.0	6.0
Damped	Iterations / 1	10 ± 3	13 ± 7
	Not Conv. / %	0.0	0.0
DIIS	Iterations / 1	7 ± 1	8 ± 1
	Not Conv. / %	0.0	0.0

As can be seen from the tables, the results are fairly close, with a slight edge towards "NN", the guess that used the overlap matrix as input. A reason for the differences may be rooted in the large number of inputs, which resulted in an expansion of the overall network dimensions and thus made the training more complex. Furthermore, the network geometry for "NN" was optimized more rigorously (Section 3.2.2), whereas for this application only trial and error-based manual tuning was done, and not as exhaustive

as for "NN". The result also shows that there is still room for improvement regarding the descriptor system. For example, the same descriptor model was used for carbon as well as hydrogen, even though their requirements are most likely different. Besides a more optimized configuration of the used symmetry functions (e.g. the distribution of radial Gaussians) one could perhaps also reduce the number of descriptors for hydrogen.

4.7. Conclusion

In this chapter we introduced a descriptor scheme flexible enough to be used in "divide and conquer"-approaches, which can give a description of the environment of one atom of the molecule at a time while still being able to capture substantial information about the environment.

That information is indeed captured was made sure on multiple levels. A tool called activation was presented to make resulting symmetry vectors more easily interpretable to humans. Fig. 4.3 and Fig. 4.4 show this for an ethene molecule. The two figures also emphasized the obvious fact that the result can be greatly enhanced by choosing the descriptor functions based on physical knowledge, e.g. for the radial sampling functions to be of the highest density in areas of typical atomic distances. In Section 4.5 a systematic comparison of descriptor functions was carried out. The result was surprising as it implicated that the weighting was rather unimportant. However, it did emphasize the importance of the angular descriptions.

5. Density Construction Schemes

5.1. Introduction

Now that the input side of our neural network guess process has been made more flexible, the output side must be treated in a similar way. We will again pursue a "divide and conquer"-approach. This means that, using ANNs, we will guess components (e.g. parts of the density matrix) independently, and assemble the full density matrix in a second step from these components. Another option is to estimate the Fock matrix this way and then use it to calculate the density matrix. We will call algorithms to assemble the density matrix "construction schemes" (regardless if it is done via construction and subsequent diagonalization of the Fock matrix or by constructing the density matrix directly). While many construction schemes exist, not all of them work sufficiently well. This chapter is dedicated to the presentation and comparison of a few selected examples. Since we are about to discuss construction schemes that involve estimating parts of either the density or the Fock matrix we shall introduce the abstract term "target matrix" to refer to the respective matrix being estimated.

5.1.1. The Best Achievable Performance

Not all of the construction schemes presented in this chapter will estimate all elements of their respective target matrices. Some will only guess parts and interpolate or fill the remaining elements of the matrix with a simpler classical guess scheme. In order to observe the effects of approximating the density matrix via a construction scheme, we shall assume "perfectly trained" ANNs in those schemes, that are able to reproduce the reference values[1] for the components of the target matrix exactly. This is done by skipping any ANNs that would appear in the guess process and directly using the reference values instead. The remaining error exhibited by an initial guess will thus be due to the construction scheme, allowing us to investigate the feasibility of the scheme independently of any ANN performance.

5.1.2. Dataset and Classical Guess Performance

We need to perform ab initio calculations in a large basis to see the full "wrath" of the side effects stemming from the assembly of the target matrices in our construction schemes from independently estimated parts. To keep computation time at an acceptable level, we

[1] The ANNs would be trained to predict values of the converged density or Fock matrix, depending on what the target matrix in the construction scheme is.

will fall back on the ethene molecule and recycle the dataset used in Section 4.1.2. The performance of the classical guess schemes on this dataset is documented in Tab. 5.1 and 5.2.

Table 5.1.: Properties of the initial guesses produced by classical schemes for the dataset samples. MAE denotes the mean absolute error of the matrix elements.

	$\mathbf{H_{core}}$	**GWH**	**SAD**
MAE / 1×10^{-2}	2.8 ± 0.5	1.4 ± 0.2	1.2 ± 0.1
HF energy error / E_h	20 ± 1	4.9 ± 0.5	0.5 ± 0.3
Idempotence / 1×10^{-18}	50 ± 14	25 ± 5	$(117 \pm 6) \times 10^{14}$
Occupancy / 1×10^{-15}	5 ± 4	4.0 ± 0.3	4.0 ± 0.4

Table 5.2.: Number of iterations required to reach convergence from the initial guesses produced by classical schemes for the dataset samples. The average numbers of iterations do not contain the results from not converged samples. The percentages of samples that did not converge are listed in the rows labeled "Not Conv.".

Method	**Quantity**	$\mathbf{H_{core}}$	**GWH**	**SAD**
Pure	Iterations / 1	48 ± 19	44 ± 16	29 ± 13
	Not Conv. / %	96.3	80.0	4.7
Damped	Iterations / 1	38 ± 1	36 ± 10	25 ± 8
	Not Conv. / %	99.8	96.6	0.0
DIIS	Iterations / 1	15 ± 1	14 ± 1	11 ± 1
	Not Conv. / %	0.0	0.0	0.0

$\mathrm{H_{core}}$ and GWH perform very poorly. In addition to a high number of iterations required to reach convergence, the number of samples that did not converge at all is also very large for both guesses. In some cases this value is close to 100 %, implying a total failure. This is a manifestation of the fact that most of the samples are rather far away from equilibrium due to the high temperature used when sampling the geometries. SAD, on the other hand, performs surprisingly well, with most of the samples reaching convergence in comparably few iterations.

5.1.3. Matrix Regions

We need to define a few terms regarding regions of the overlap, density and Fock matrix. As an example, all of the regions to be discussed are illustrated for an ethene molecule in Fig. 5.1. The regions of the overlap matrix, where basis functions overlap that are centered around the same nucleus, we shall call "self-overlap" regions. They form diagonal blocks, marked as "HH Self-Overlap" and "CC Self-Overlap" in the example. Regions related to basis functions centered at two different atoms of the same atomic species, we will name

regions of "homonuclear" overlap ("HH Overlap" and "CC Overlap"). Similarly, we will refer to regions corresponding to the overlap of basis functions centered at different cores of atoms of differing elements as "heteronuclear" overlap (marked as "CH Overlap" in Fig. 5.1). Since overlap, Fock, and density matrix are all represented in the same basis, we can use this terminology to refer to analogous regions in these matrices.

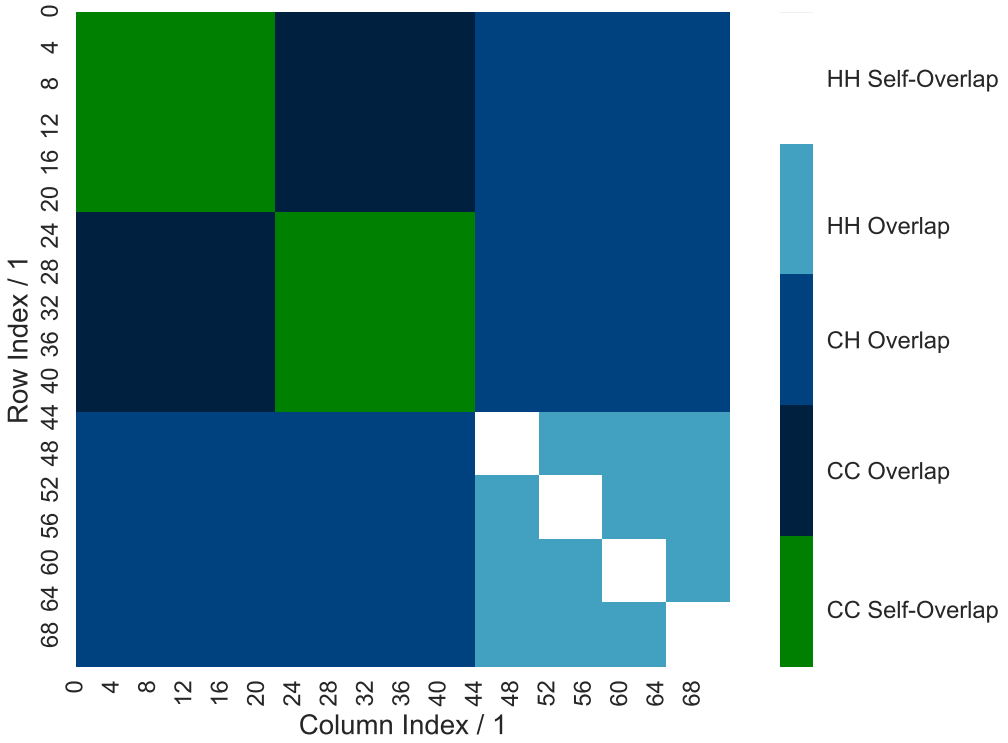


Figure 5.1.: Regions of the overlap matrix of ethene in 6-311++G** basis. The contraction schemes are $(6s, 1p) \rightarrow [4s, 1p]$ for hydrogen and $(12s, 6p, 1d) \rightarrow [5s, 4p, 1d]$ for carbon.

5.2. Wolfsberg-Helmholz 2.0: E-GWH

In Section 2.1.9 we outlined a classical guess scheme that approximates the diagonal of the Fock matrix by the diagonal of the core Hamiltonian. It further relies on the Wolfsberg-Helmholz scheme (c.f. (2.34)) to estimate the off-diagonal elements via.

$$F_{\mu\nu} = KS_{\mu\nu}\frac{H^{\text{core}}_{\mu\mu} + H^{\text{core}}_{\nu\nu}}{2},$$

with $K = 1.75$ usually [18]. To improve this ansatz, we will simply use a diagonal estimated by ANNs instead of $\boldsymbol{H}^{\text{core}}$. ANNs means in this case, that we will use a separate network for each chemical element, and assemble the diagonal according to molecular composition. This gives us the desired flexibility. A look at the average error in Fock

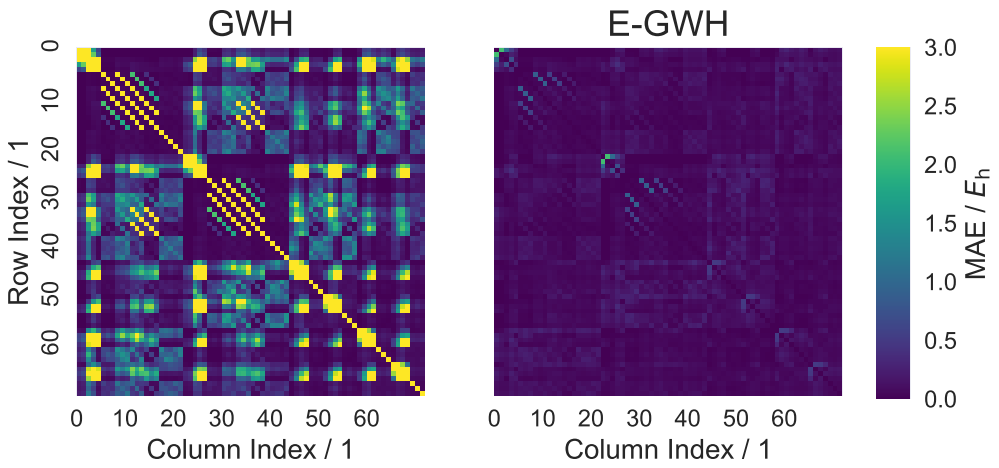


Figure 5.2.: Mean absolute error (MAE) of elements of the Fock matrix for GWH and E-GWH, clipped at an error of 3.0 E_h .

matrix in Fig. 5.2 reveals that it also leads to tremendous improvements. Just as with the classical pendant, the resulting guess for the Fock matrix is then used to construct a density matrix. The results are listed in Tab. 5.3 and 5.4. While there is definitely an improvement to GWH regarding the rate of samples that did not converge (a reduction by almost 50 %), it is still very high. A slight reduction in the number of iterations is also noticeable. However, the error in HF energy is enormous and even larger than for the classical GWH. The reason for this is not entirely clear and contradictory to the obvious improvements regarding the estimate of the Fock matrix as shown in Fig. 5.2.

Table 5.3.: Properties of the E-GWH initial guess produced for the dataset samples. MAE denotes the mean absolute error of the matrix elements.

	E-GWH
MAE / 1×10^{-1}	2.8 ± 2.8
HF energy error / E_h	25 ± 2
Idempotence / 1×10^{-16}	10 ± 5
Occupancy / 1×10^{-15}	7 ± 5

Table 5.4.: Number of iterations required to reach convergence from the initial guesses produced by the E-GWH scheme for the dataset samples. The average numbers of iterations do not contain the results from not converged samples. The percentages of samples that did not converge are listed in the rows labeled "Not Conv.".

Method	Quantity	E-GWH
Pure	Iterations / 1	36 ± 16
	Not Conv. / %	43.2
Damped	Iterations / 1	29 ± 10
	Not Conv. / %	48.9
DIIS	Iterations / 1	13 ± 1
	Not Conv. / %	0.0

5.3. The Center Blocks: E-GWH+ and Embedded GWH

The next construction scheme to be introduced is based on the E-GWH scheme from Section 5.2. We shall call it "E-GWH+". The Fock matrix estimate of E-GWH is improved by replacing the center blocks (self-overlap regions; cf. Fig. 5.1) with an ANN prediction for the values in these regions, similarly to how the density matrix is set up in the SAD guess. In practice, this of course does not mean that two separate networks for diagonal and center blocks are required. Rather than that, the center block is guessed by a network and the diagonal of this block is used to fill the outer regions (both homonuclear and heteronuclear overlap regions). The resulting Fock matrix is subsequently diagonalized to calculate the density matrix, which can finally be used as initial guess. In addition, because of its easy accessibility, we will also analyze the result of embedding the center blocks of a Fock matrix that was generated via the classical GWH scheme (which uses the diagonal of the one-electron Hamiltonian matrix, $\boldsymbol{H}^{\text{core}}$). We refer to this second scheme as "Embedded GWH". Results are compiled in Tab. 5.5 and Tab. 5.6.

Table 5.5.: Properties of the initial guesses produced for the dataset samples. MAE denotes the mean absolute error of the matrix elements.

	E-GWH+	Embedded GWH
MAE / 1×10^{-2}	103 ± 107	5 ± 4
HF energy error / E_h	24 ± 4	4.4 ± 0.3
Idempotence / 1×10^{-17}	$(1.4 \pm 1.1) \times 10^3$	7.5 ± 1.8
Occupancy / 1×10^{-14}	1.6 ± 1.5	0.4 ± 0.3

We notice that both mean absolute errors and HF energy errors are very large once again. A scatter plot (see Fig. 5.3) is employed to check for systematic deviations. Unfortunately, it reveals a complete failure of all our Fock matrix-based approaches, as was to be expected due to the HF energy error. Even worse, it does not show any characteristic pattern, which

Table 5.6.: Number of iterations required to reach convergence from the initial guesses produced for the dataset samples. The average numbers of iterations do not contain the results from not converged samples. The percentages of samples that did not converge are listed in the rows labeled "Not Conv.".

Method	Quantity	E-GWH+	Embedded GWH
Pure	Iterations / 1	37 ± 17	37 ± 16
	Not Conv. / %	42.6	30.0
Damped	Iterations / 1	28 ± 9	27 ± 7
	Not Conv. / %	35.4	25.4
DIIS	Iterations / 1	13 ± 2	13 ± 1
	Not Conv. / %	0.0	0.0

could point at a potential source of the large HF energy errors. It is noteworthy, however, that Embedded GWH, which is much closer to the classical GWH scheme, exhibits a much lower HF energy error, thus raising the suspicion that off-diagonal estimates of the Fock matrix based on a more accurately estimated diagonal are not as good as expected. Despite this issue, the robustness of the guesses was improved significantly in comparison to the classical GWH scheme. The same is true for the number of iterations.

5.4. Superposition of Atomic Neural Network Densities (SANND)

In the previous sections we introduced construction schemes that sought to improve upon guesses for the Fock matrix. In a final effort, we will try a brute-force approach, i.e. a direct estimate of the density matrix. We will use ANNs to estimate parts of the density matrix (e.g. self-overlap blocks) and embedded them in the output of a simpler classical guess, namely GWH, to improve it. This should give us an edge to SAD, which is very similar to our approach (it estimates self-overlap blocks and embeds them in a matrix of zeros). To emphasize this similarity, the scheme is called Superposition of Atomic Neural Network Densities (SANND). We will distinguish between the following flavors of SANND:

Center: Here we will guess the center blocks of the density, similarly to SAD. This corresponds to the self-overlap areas in Fig. 5.1.

Homo: Here we will guess the blocks of the homonuclear overlap regions in addition to the center blocks.

Hetero: Here we will guess the blocks that correspond to heteronuclear overlap in addition to the center blocks.

Note that a combination of Homo and Hetero would imply to estimate the full density matrix. We will first guess a "raw" density, which is used to build a "raw" Fock matrix. Finally, this "raw" Fock matrix is diagonalized to build a proper density matrix, ready to

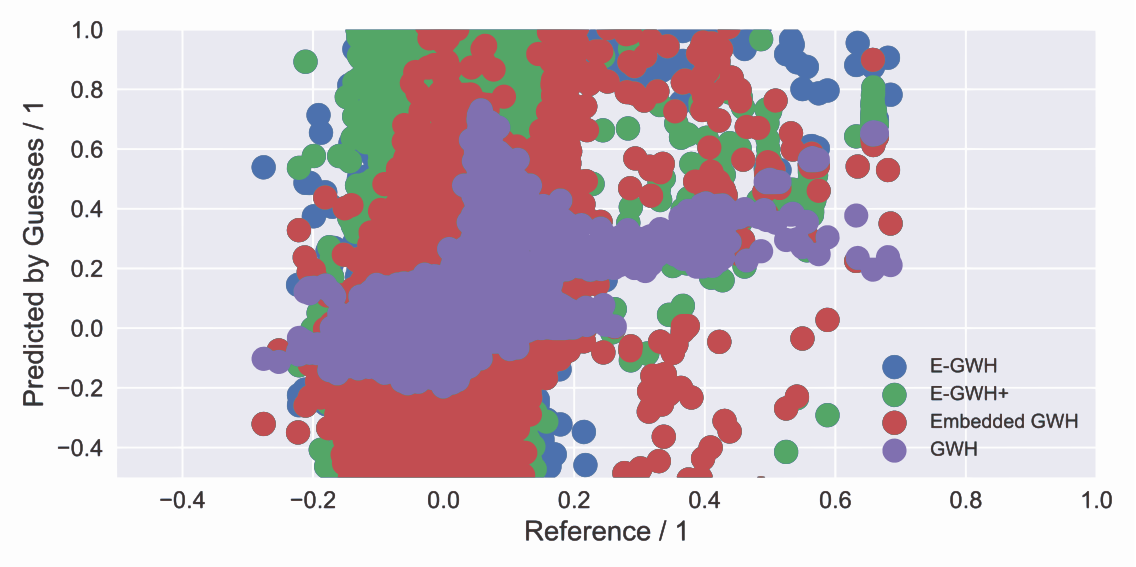


Figure 5.3.: Elements of the density matrix as predicted by Fock matrix-based guesses, plotted against the reference values from the converged density matrices. Since the product of number of elements, samples, and displayed guesses yield a fairly big number, not all samples are included in the making of this plot. Instead, 30 examples have been chosen randomly from the dataset for visualization.

be used as initial guess. This extra step of building and diagonalizing a "raw Fock matrix" helps to smoothen the density matrix. The performances of the various SANND guesses are summarized in Tab. 5.7 and Tab. 5.8.

Table 5.7.: Properties of the SANND initial guess produced for the dataset samples. MAE denotes the mean absolute error of the matrix elements.

	Center	**Homo**	**Hetero**
MAE / 1×10^{-3}	6 ± 10	5 ± 6	0.03 ± 0.73
HF energy error / E_h	0.17 ± 0.08	0.16 ± 0.07	0.06 ± 0.20
Idempotence / 1×10^{-17}	2.8 ± 0.8	2.8 ± 0.6	490.0 ± 784.4
Occupancy / 1×10^{-15}	4 ± 3	5 ± 4	6 ± 20

Table 5.8.: Number of iterations required to reach convergence from the initial guesses produced by the SANND schemes for the dataset samples. The average numbers of iterations do not contain the results from not converged samples. The percentages of samples that did not converge are listed in the rows labeled "Not Conv.".

Method	**Quantity**	**Center**	**Homo**	**Hetero**
Pure	Iterations / 1	30 ± 13	30 ± 14	27 ± 13
	Not Conv. / %	5.1	4.8	5.0
Damped	Iterations / 1	25 ± 7	23 ± 6	24 ± 6
	Not Conv. / %	0.0	0.0	0.3
DIIS	Iterations / 1	12 ± 1	11 ± 1	11 ± 1
	Not Conv. / %	0.0	0.0	0.0

All SANND guesses show a similar performance. Homo is slightly more robust than Hetero, while the latter requires slightly less iterations on average. All guesses are very close to the classical SAD scheme. Hetero even surpasses SAD regarding the number of iterations needed in Pure and Damped settings. Homo performs similarly well, beating SAD for Damped by two iterations, but requiring one additional iteration for the Pure case. The SANND guesses also perform well regarding matrix properties. This is especially true for Hetero, which exhibits a very low mean squared error and HF energy error; The high error in idempotence, thus seems not to be a problem. As already stated, the differences between the SANND guesses and SAD are not too large and they obviously vary greatly with the dataset.

The outcome clearly shows that both Homo and Hetero are an improvement to Center. There is, however, a downside to the improvements that Homo and Hetero bring. Note that both come with a great increase in effort because of the combinatorics of possible overlaps: the more elements appear in a molecule, the more distinct types of overlap regions will appear in the density matrix. For each type of overlap, a *separate* ANN is required. While this results in only one ANN per element for Center, an additional ANN per element is required to make the extension from Center to Homo. The step from Center to Hetero is even worse, as another ANN for each possible combination of two elements is needed. (Strictly speaking, only for each ordered combination, e.g. the heteronuclear overlap region of the combination C-H may be processed via the same ANN as the combination H-C.) For a molecule build from atoms of N different elements, this means that N ANNs are required for Center, $2N$ for Homo, and a painful $\frac{1}{2}\binom{N}{2}$ for Hetero.

5.5. Comparison and Conclusion

In a final comparison we want to look at the predictions for the electron density in the molecule. The respective predictions produced by a selection of guess schemes is shown in Fig. 5.4. Here we can see the spherical atomic regions produced by SAD. In spite of the fact that it is actually geometry-independent, the produced electron density is fairly

close to the converged reference. GWH, on the other side, which should have geometry dependence, even fails to reproduce an at least qualitatively correct electron density in the area of the hydrogens. The guesses produced by our Fock matrix-based construction schemes are also not very convincing: The density produced by Embedded GWH remotely resembles the correct "form", but performs badly regarding details near the hydrogens. On the other hand, E-GWH seems to overestimate the density in these areas, which causes strange, unrealistic holes in other places (e.g. between the two carbon nuclei). The only guess that comes close to the quality of SAD is SANND. In many details it even surpasses the classical guess. An example is the bulge in the density in the centers of the triangles spanned by the hydrogens and the carbon next to them, which SAD can not provide due to its spherical averaging.

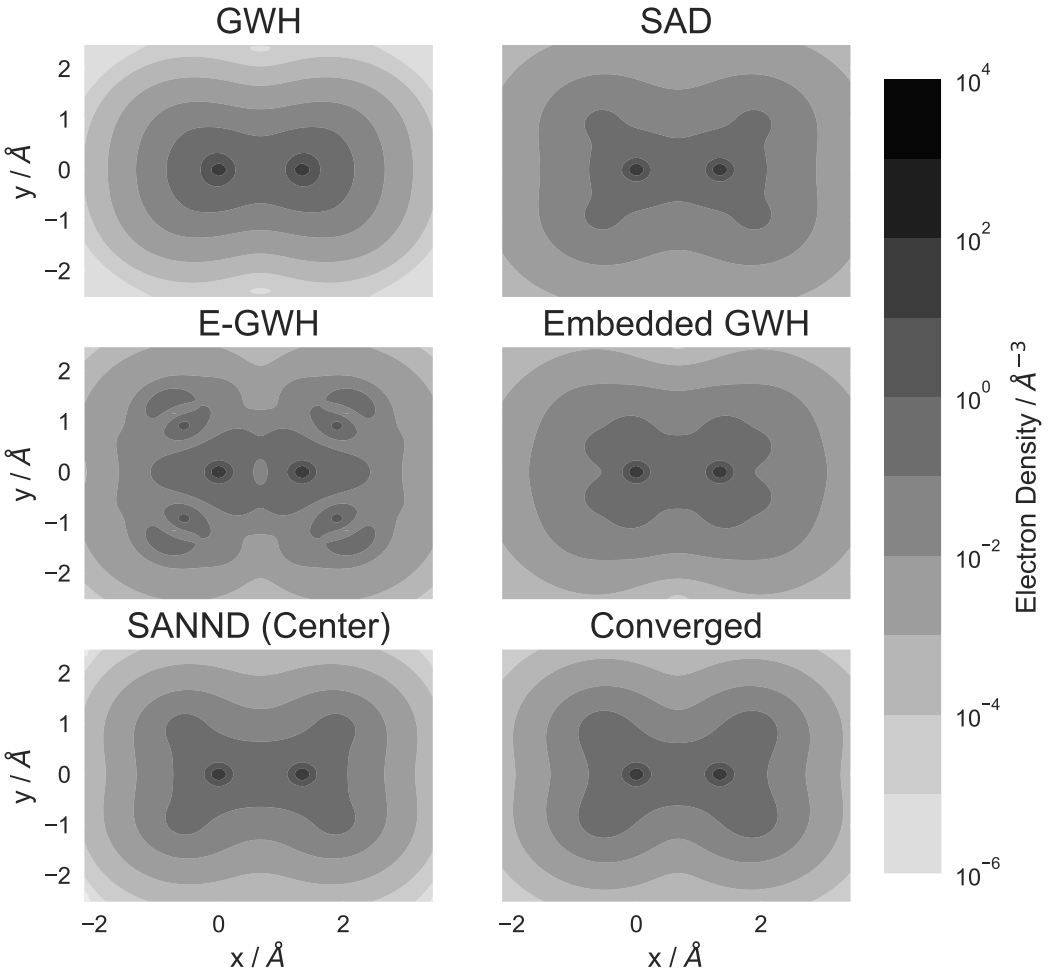


Figure 5.4.: Cut at $z = 3\,\text{Å}$ through the electron density of ethene, lying in the x-y plane ($z = 0\,\text{Å}$), as obtained for various guesses, and compared to the converged result.

5.5.1. Conclusion

Before concluding, a word of caution must be repeated: tests such as the ones carried out above show a considerable dependence on the dataset. This can already be seen by the poor performance of H_{core} and GWH, which are simply not designed for the rather challenging off-equilibrium geometries found in the dataset at hand.

Out of all the presented schemes, the SANND approach, especially with the more complex forms Homo and Hetero, appears to be the most promising. Its convergence behavior (speed and robustness) is much better than that of GWH, which could hardly converge any of the samples, and rather close to SAD, performing very similarly (with a very slight edge to SANND). We did not include a combination of Homo and Hetero in the tests of this chapter, because the resulting guesses (using best achievable performance; cf. Section 5.1.1), would be identical to the converged density matrices. However, after having seen the performance of Homo and Hetero, it is fair to assume that a combination of the two would make for an even better guess. On the other hand, it should be noted that Homo and Hetero are probably very impractical to implement, because the number of ANNs to train increases quickly with the number of chemical elements involved.

The poor performance of Fock matrix-based approaches is very disappointing. While they proved to be significantly better (especially regarding robustness) than the classical GWH guess, none of them was even close to being a threat to SAD or SANND.

In conclusion, the SANND approaches (Homo and Hetero) showed to be most promising. This ansatz is much more accurate than GWH and very close to (partly even better than) SAD. A combination of Homo and Hetero surely has great potential.

6. Application

6.1. Introduction

After having experimented with various ways to formulate a description of the molecular environment in Chapter 4, and having investigated a few schemes to generate a density matrix from a guess made using this descriptions, it is finally time to put the results of this research to work and apply them to a test dataset.

We choose a dataset containing off-equilibrium geometries of carbohydrate molecules (ethane, ethene, and ethyne). Since there are only two elements involved (carbon and hydrogen), the effort to implement SANND is minimal.

6.2. SANND Architecture and Network Training Conditions

We will employ the SANND guess scheme from Section 5.4 in the form of a combination of SANND (Homo) and SANND (Hetero), which we shall call "SANND" (without trailing specification in parentheses). In SANND, all elements of the density matrix are estimated with ANNs. Two networks are needed for SANND (Center), two more for the homonuclear overlap regions (C-C) and (H-H), and another one for the heteronuclear regions (C-H). Obviously, there will be molecules in which an element appears more than once. Therefore, the number of samples does not match the number of molecules anymore, because these molecules will provide as many samples as they contain atoms of that specific element. A consequence is that not all ANNs have the same amount of information available for training. This is all the more troubling as heavier atoms with more electrons, such as e.g. carbon, lead to larger regions in the density matrix. Therefore larger networks are required, which are also more difficult to train because these heavier atoms typically do not appear as often in molecules as e.g. hydrogen, which results in fewer training samples.

6.3. The Dataset

The dataset contains various geometries of ethane, ethene and ethyne. They are sampled using multiple MD simulations each, all at temperatures between 273 K and 450 K. This leads to geometries being far from their equilibrium. In addition to this, the initial geometries are hand-crafted to minimize similarity between the runs and to be able to sample broad regions of the phase space. The time step used in the simulations is 0.5 fs, and the number of steps varies from 400 to 1000 steps. Occasionally, molecules tend to disassemble

during the simulation due to the high temperatures. Affected structures are discarded as soon as the distance between any carbon and any hydrogen atom in the molecule exceeds 10 Å.

The dataset consists of 11 070 structures in total, adding up to 19 617 carbon and 43 738 hydrogen atoms. Usage assignment is as follows: 20 % of the molecules are reserved for testing, 20 % of the remaining molecules are used for validation and early stopping and the rest for network training[1]. Fig. 6.1 shows the distribution of C-H distances in the dataset to illustrate its structural variety.

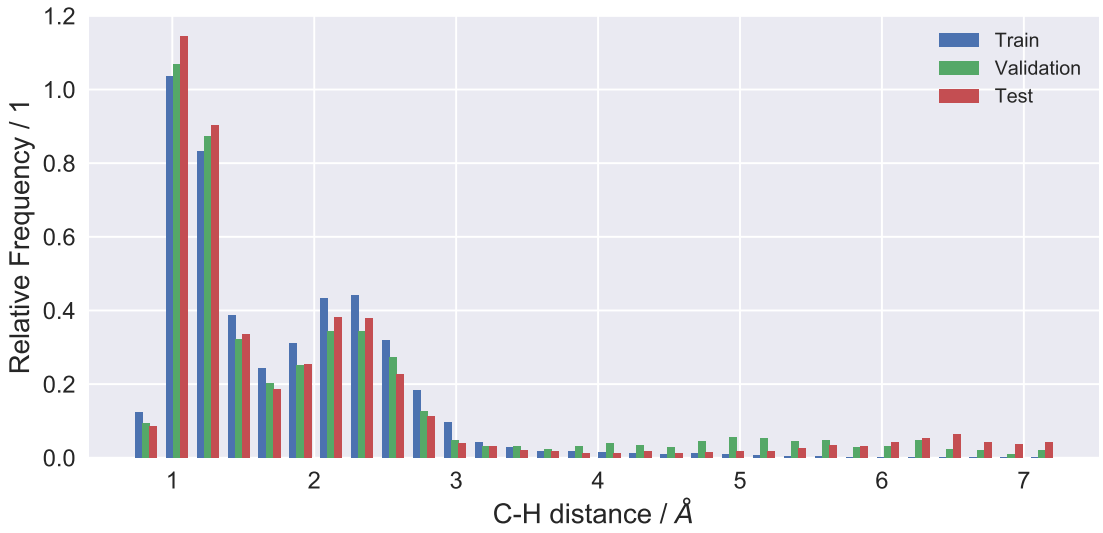


Figure 6.1.: Relative frequency of carbon-hydrogen (C-H) bond lengths.

6.4. Descriptor Details and Results

As a descriptor we will use our Gaussians with the respective models for the radial part and for the angular spherical harmonics. As introduced in Section 4.5 weighting will consist of a cut-off function times a physical quantity (as specified in (4.12)). The cuff-off function Behler1 (see equation (4.10)) with a cut-off of $R_c = 5$ Å and for the physical quantity the atomic number Z is chosen. For the networks describing homo- and heteronuclear overlap regions, combinations of the descriptors used for the center blocks (listed in Tab. 6.1) are employed. The resulting network structures can be found in Tab. 6.2.

[1] Reminder: since a molecule can consist of multiple samples, the number of molecules is smaller than the number of samples.

Table 6.1.: Configuration of the used descriptors. The column "Radial" lists the radial model used (see Section B.1) and the column "Angular" the angular model. "SPH-X" denotes spherical harmonics with $l_{\max} = X$.

Species	**Radial**	**Angular**
H	Unif-25	SPH-3
C	Unif-25	SPH-5

Table 6.2.: Configuration of the atomic networks for different combinations of overlap region type and chemical species involved. In the column "Structure" the number of nodes in the layers of the network used are listed (including input and output; layers are separated by a hyphen).

Type of Overlap	**Species**	**Structure**
Self	H	57-50-50-28
Self	C	75-100-200-253
Homonuclear	H-H	114-100-70-49
Homonuclear	C-C	150-200-350-484
Heteronuclear	C-H	132-200-200-154

Table 6.3.: Properties of initial guesses produced by various guess schemes for the dataset samples. MAE denotes the mean absolute error of the matrix elements.

	GWH	**SAD**	**SANND (Center)**	**SANND**
MAE / 1×10^{-2}	1.3 ± 0.3	1.1 ± 0.2	0.5 ± 0.2	0.7 ± 0.3
HF energy error / E_h	6 ± 2	0.3 ± 0.2	0.2 ± 0.4	0.5 ± 0.7
Idempotence / 1×10^{-17}	2.4 ± 0.7	$(1.1 \pm 0.2) \times 10^{15}$	2.5 ± 0.8	2.8 ± 0.8
Occupancy / 1×10^{-15}	4 ± 4	4 ± 3	5 ± 4	4 ± 4

The results are listed in Tab. 6.3 and 6.4 together with the results of GWH and SAD. As expected, both SANND schemes perform better than GWH and SANND performs slightly worse than SAD. More interesting is the fact that SANND (Center) yields a better guess than SANND. The differences are small, but almost always in favour of SANND (Center). In Fig. 6.2 we can see why: the estimates for homo- and heteronuclear overlap areas are of lower quality. This can only be a result of poor ANN performance. Similarly to the values of Tab. 6.3 and 6.4 the differences are small but significant. Most noticeable are the errors in the self and homonuclear overlap regions of the carbon atoms in the ethene molecules. This was to be expected, since, as described in the introduction of this chapter, all carbon related ANNs are larger and fewer samples are available for their training. The difference in error in the self overlap blocks is a consequence of the diagonalization of the "raw" density (cf. Section 5.4), during which the errors seem to diffuse and spread to other regions, e.g. from the C1-C2 homonuclear overlap to the C1 and C2 self overlap regions.

Table 6.4.: Number of iterations required to reach convergence from the initial guesses produced various guess schemes. The average number of iterations does not contain the results from not converged samples. The percentage of samples that did not converge are listed in the row labeled "Not Conv.".

Method	Quantity	GWH	SAD	SANND (Center)	SANND
Pure	Iterations / 1	28 ± 15	26 ± 17	30 ± 18	31 ± 17
	Not Conv. / %	60.1	11.2	19.6	26.0
Damped	Iterations / 1	27 ± 11	25 ± 12	28 ± 15	28 ± 14
	Not Conv. / %	71.7	1.3	9.3	14.7
DIIS	Iterations / 1	14 ± 4	11 ± 2	11 ± 4	12 ± 4
	Not Conv. / %	0.0	0.0	0.0	0.0

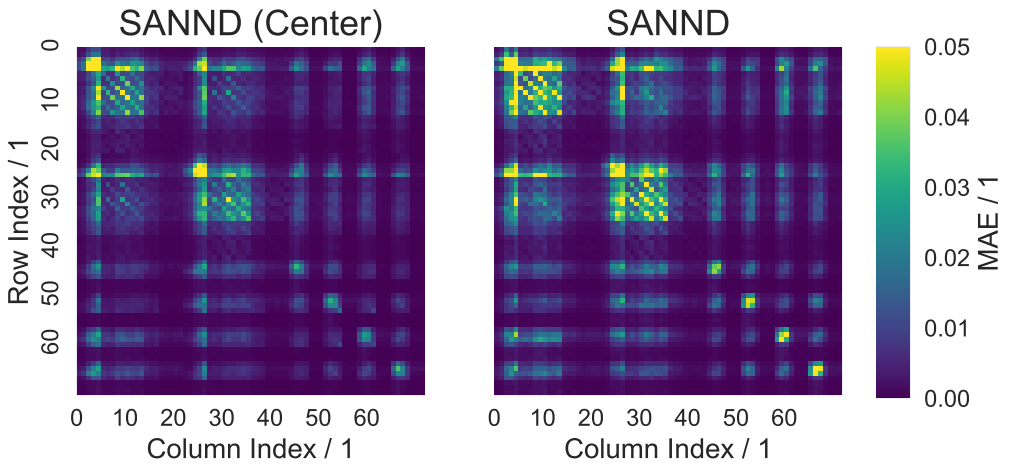


Figure 6.2.: Average mean absolute error (MAE) of elements in the density matrix produced by SANND (Center) and SANND for all ethene molecule samples of the test dataset, clipped at an error of 0.05 (dimensionless).

6.5. Conclusion

Our guesses SANND (Center) and SANND performed worse than the classical approaches. This is not entirely surprising as the performances of SAD and SANND (Center) were fairly close in Section 5.4 already where the converged reference values had been used directly instead of a fit. With the ANN fit as an additional source of error, SANND (Center) is less accurate. Originally, there was some hope that SANND, being the combination of SANND (Homo) and SANND (Hetero), would yield an improvement. Sadly, the homo- and heteronuclear overlap regions were even harder to learn for the ANNs than the center

blocks, which resulted in even worse predictions. Yet, the gap is not too large and on a different dataset things may look different.

7. Conclusion and Outlook

The aim of this thesis was to investigate new ways to generate initial guesses for SCF calculations using machine learning techniques and ANNs in particular.

In Chapter 3 we started with a very simple approach, trying to find a mapping from the overlap matrix to the density matrix of a molecule. Inspired by Hückel Theory and the generalized Wolfsberg-Helmholz scheme, the question arose whether a mapping of this form exists. According to the results from Section 3.2.2 this assumption can be confirmed. In this process we also showed that it is possible to capture the molecular environment using the overlap matrix and to estimate the density matrix via ANNs. The results could be improved even further using the McWheeny purification, to give a very low error in HF energy.

However, this approach showed limited flexibility. Therefore, we constructed more general descriptor schemes in Chapter 4, which finally allowed for a treatment of arbitrary molecules on the input side of the guessing process. This was achieved by an ansatz similar to Behler and Parinello's work on Neural Network Potentials, where the description of the environment of an atom is achieved by a combination of contributions from all surrounding atoms in the molecule. Several ways to define these contributions and their combinations have been suggested. We find that a spherical harmonics-based approach for the angular dependency in combination with Gaussians for the radial part is most promising, while the actual weighting in the combination seems to have little influence on the final outcome.

Additionally, several schemes have been presented to make the output side of our guess process independent of molecular composition. Some of these "construction schemes", as we called them, were again inspired by GWH, but the most convincing approach was SANND, a group of schemes similar to SAD.

In this preliminary work, our models have been applied to carbon and hydrogen-based structures exclusively. The performance on this limited and highly specific test set was slightly worse than that of the classical guesses.

7.1. Outlook

Next steps should aim at a further improvement of the ANN performance in the implementations of the SANND scheme to leverage its full potential. The scheme could also be applied to other, more diverse datasets, featuring molecules that consist of other elements than just hydrogen and carbon.

Another issue concerns the fixed spatial orientation of the density matrix. Obviously, the

electron density of a molecule (as was discussed for the energy in Section 2.3.4) is the same regardless of how the atom is rotated (if external fields are absent). Due to the fact that the density matrix is represented in a set of spatially fixed basis functions, equivalent densities (differing only by e.g. a rotation) have different representations. This is highly undesired for network training because now two identical samples are treated separately, resulting in numerous redundancies. Training speed and convergence could be greatly improved if identical structures could be identified and processed as identical examples. For instance, they could be mapped to an intermediate input, which is used by the ANN to generate an intermediate output, which is in turn mapped to the density matrix in the spatially fixed representation.

A. Mathematical Details

A.1. Born-Oppenheimer Approximation

The Born-Oppenheimer approximation, sometimes also referred to as the adiabatic approximation, makes use of the different masses in a molecular system and separates the nuclear and the electronic motion to obtain two decoupled equations for the electronic and the nuclear energies, respectively. This derivation follows the one given by A. C. Hurley[44, p. 1 *et seqq.*]. Starting with the full many-body Hamiltonian from (2.2), we realize that it consists of kinetic and potential contributions:

$$\begin{aligned} H = &\underbrace{-\frac{1}{2}\sum_{i=1}^{N}\nabla_i^2}_{T_\mathrm{e}} \underbrace{-\frac{1}{2}\sum_{A=1}^{M}\frac{1}{M_A}\nabla_A^2}_{T_\mathrm{N}} \\ &\underbrace{-\sum_{i=1}^{N}\sum_{A=1}^{M}\frac{Z_A}{r_{i,A}}}_{V_\mathrm{Ne}} + \underbrace{\sum_{i=1}^{N}\sum_{j>i}^{N}\frac{1}{r_{ij}}}_{V_\mathrm{ee}} + \underbrace{\sum_{A=1}^{M}\sum_{B>A}^{M}\frac{Z_A Z_B}{R_{AB}}}_{V_\mathrm{NN}}, \end{aligned} \tag{A.1}$$

T_e and T_N denote electronic and nuclear kinetic terms, respectively, while V_Ne, V_ee, V_NN denote the nuclear-electron, electron-electron and nuclear-nuclear potential terms. Assuming constant nuclear coordinates $\boldsymbol{R} := \{R_A : 1 \leq A \leq 3M\} \overset{!}{=} \text{const.}$ we obtain the electronic Hamiltonian (2.3), which gives us the electronic Schrödiger equation, in the form

$$\left[T_\mathrm{e} + V(\boldsymbol{r}, \boldsymbol{R})\right]\psi_i(\boldsymbol{r}, \boldsymbol{R}) = E_i(\boldsymbol{R})\psi_i(\boldsymbol{r}, \boldsymbol{R}), \tag{A.2}$$

with $\boldsymbol{r} := \{r_i : 1 \leq i \leq 3N\}$ as the coordinates of the positions of the electrons and $\boldsymbol{R}$ as a parameter. The electronic eigenfunctions $\psi(\boldsymbol{r}, \boldsymbol{R})$ form a complete, orthonormal basis of the set of functions of the electronic coordinates $\boldsymbol{r}$. This allows us to expand the total molecular wave function in terms of the following:

$$\Psi(\boldsymbol{r}, \boldsymbol{R}) = \sum_i \phi_i(\boldsymbol{R})\psi_i(\boldsymbol{r}, \boldsymbol{R}). \tag{A.3}$$

Inserting (A.3) into the Schrödinger equation (2.1) yields

$$\left[T_\mathrm{e} - T_\mathrm{N} + V(\boldsymbol{r}, \boldsymbol{R}) - \mathcal{E}\right]\sum_i \phi_i(\boldsymbol{R})\psi_i(\boldsymbol{r}, \boldsymbol{R}) = 0 \tag{A.4}$$

where $\mathcal{E}$ represents the total energy of the system. Multiplication by ψ_j^* and integration over the whole space of electronic coordinates yields

$$\sum_i \int \psi_j^*(\boldsymbol{r}, \boldsymbol{R})\left[T_\mathrm{e} + T_\mathrm{N} + V(\boldsymbol{r}, \boldsymbol{R}) - \mathcal{E}\right]\phi_i(\boldsymbol{R})\psi_i(\boldsymbol{r}, \boldsymbol{R})\,\mathrm{d}\boldsymbol{r} = 0. \tag{A.5}$$

Combining the above with (A.2) and using the orthonormality of the electronic eigenfunctions $\psi(\boldsymbol{r}, \boldsymbol{R})$ we obtain

$$\sum_i \left[\int \psi_j^*(\boldsymbol{r}, \boldsymbol{R}) T_\mathrm{N} \psi_i(\boldsymbol{r}, \boldsymbol{R}) \, \mathrm{d}\boldsymbol{r} + (E_i(\boldsymbol{R}) - \mathcal{E}) \delta_{ij} \right] \phi_i(\boldsymbol{R}) = 0. \tag{A.6}$$

Now we apply T_N to ψ and ϕ. We obtain (again using orthonormality)

$$\begin{aligned} \int \psi_j^* T_\mathrm{N} \psi_i \phi_i \, \mathrm{d}\boldsymbol{r} = &- \sum_A \frac{1}{2M_A} \int \psi_j^* \frac{\partial^2 \psi_i}{\partial R_A^2} \phi_i \, \mathrm{d}\boldsymbol{r} \\ &- 2 \sum_A \frac{1}{2M_A} \int \psi_j^* \frac{\partial \psi_i}{\partial R_A} \frac{\partial \phi_i}{\partial R_A} \, \mathrm{d}\boldsymbol{r} \\ &- \sum_A \frac{1}{2M_A} \frac{\partial^2 \phi_i}{\partial R_A^2} \delta_{ij}, \end{aligned} \tag{A.7}$$

which - after inserting (A.7) into (A.6) - leads us to

$$\begin{aligned} &\left[- \sum_A \frac{1}{2M_A} \frac{\partial^2}{\partial R_A^2} + E_i(\boldsymbol{R}) - \mathcal{E} \right] \psi_j(\boldsymbol{R}) = \\ &\sum_i \sum_A \frac{1}{M_A} \int \psi_j^* \frac{\partial \psi_i}{\partial R_A} \frac{\partial \phi_i}{\partial R_A} \, \mathrm{d}\boldsymbol{r} \sum_i \sum_A \frac{1}{2M_A} \int \psi_j^* \frac{\partial^2 \psi_i}{\partial R_A^2} \phi_i \, \mathrm{d}\boldsymbol{r} \, . \end{aligned} \tag{A.8}$$

Finally, we define the coupling operator

$$\Lambda_{nm} := \sum_A \frac{1}{M_A} \int \mathrm{d}\boldsymbol{r} \, \psi_n^* \left[\frac{\partial}{\partial R_A} \psi_m \right] \frac{\partial}{\partial R_A} + \sum_A \frac{1}{2M_A} \int \mathrm{d}\boldsymbol{r} \, \psi_n^* \left[\frac{\partial^2}{\partial R_A^2} \psi_m \right] \tag{A.9}$$

to shorten the expression in (A.8), which finally yields the compressed equation:

$$\left[T_\mathrm{N} + E_j(\boldsymbol{R}) - \mathcal{E} \right] \phi_j(\boldsymbol{R}) = \sum_i \Lambda_{ji} \phi_i(\boldsymbol{R}). \tag{A.10}$$

The Born-Oppenheimer approximation is now to ignore the coupling in (A.10), i.e.

$$\Lambda_{ji} \stackrel{!}{=} 0, \tag{A.11}$$

which gives us two decoupled sets of equations, one for the nuclear part

$$\left[T_\mathrm{N} + E_n(\boldsymbol{R}) - E_{n\nu} \right] \phi_{n\nu}(\boldsymbol{R}) = 0 \tag{A.12}$$

and (A.2) for the electronic part. In the nuclear part, the total energy was relabeled $\mathcal{E} \to E_{n\nu}$ to express that there is a different set of electronic states (label by n) for each nuclear state (labeled by ν). The total wave function is now a product of an electronic and a nuclear wave function:

$$\Psi_{n\nu}(\boldsymbol{r}, \boldsymbol{R}) = \psi_n(\boldsymbol{r}, \boldsymbol{R}) \phi_{n\nu}(\boldsymbol{R}). \tag{A.13}$$

As for every approximation, the question of the boundaries of validity arises immediately. In the context of perturbation theory a criterion for validity can be obtained

$$\frac{|\langle\phi_{n\nu}|\Lambda_{nm}|\phi_{m\nu'}\rangle|}{|E_{n\nu}-E_{m\nu'}|} \ll 1 \quad \forall\nu,\nu',n,m:\nu\neq\nu',n\neq m. \tag{A.14}$$

The condition can be interpreted in the following way: the matrix elements of the coupling operator have to be small compared to the difference in total energy of the system. Except for when the matrix elements vanish due to symmetry, this will no longer be the case when electronic energy differences approach the order of magnitude of vibronic frequencies.

A.2. Practical Notations for Common Integrals

Since writing out all integrals for our derivations is very tiring we will define a few shortcuts to save space and time. Very frequent are two-electron integrals of r_{ij}^{-1} (with r_{ij} the distance between two electrons i and j). Therefore, we define

$$\langle ij|kl\rangle := \langle\phi_i\phi_j|\phi_k\phi_l\rangle := \int \phi_i^*(\boldsymbol{x}_1)\phi_j^*(\boldsymbol{x}_2)\frac{1}{r_{ij}}\phi_k(\boldsymbol{x}_1)\phi_l(\boldsymbol{x}_2)\mathrm{d}\boldsymbol{x}_1\mathrm{d}\boldsymbol{x}_2 \tag{A.15}$$

and

$$\langle ij||kl\rangle := \langle\phi_i\phi_j||\phi_k\phi_l\rangle := \int \phi_i^*(\boldsymbol{x}_1)\phi_j^*(\boldsymbol{x}_2)\left(\frac{1}{r_{ij}}-\mathcal{P}_{12}\right)\phi_k(\boldsymbol{x}_1)\phi_l(\boldsymbol{x}_2)\mathrm{d}\boldsymbol{x}_1\mathrm{d}\boldsymbol{x}_2 \tag{A.16}$$

with $\mathcal{P}_{nm}$ as the operator that exchanges electron n and m. Note that

$$\langle ij||kl\rangle = \langle ji||lk\rangle \tag{A.17}$$
$$\langle ij||kl\rangle = \langle kl||ij\rangle^* \tag{A.18}$$
$$\langle ij||kk\rangle = 0. \tag{A.19}$$

The orbitals ϕ above include spin. Therefore, the integrals are over the product space of spin spaces and coordinate spaces. We also define notations for purely spacial integrals,

$$(ij|kl) := (\varphi_i\varphi_j|\varphi_k\varphi_l) := \int \varphi_i^*(\boldsymbol{r}_1)\varphi_j^*(\boldsymbol{r}_2)\left(\frac{1}{r_{ij}}-\mathcal{P}_{12}\right)\varphi_k(\boldsymbol{r}_1)\varphi_l(\boldsymbol{r}_2)\mathrm{d}\boldsymbol{r}_1\mathrm{d}\boldsymbol{r}_2, \tag{A.20}$$

two of which receive a special name:

$$J_{ij} := (ii|jj) \qquad \text{(Coulomb integrals)} \tag{A.21}$$
$$K_{ij} := (ij|ji) \qquad \text{(Exchange integrals).} \tag{A.22}$$

There are also shorter notations for integrals involving one-electron operators h

$$\langle i|h|j\rangle := \int \phi_i^*(\boldsymbol{x}_1)h(\boldsymbol{r}_1)\phi_j(\boldsymbol{x}_1)\mathrm{d}\boldsymbol{x}_1 \qquad \text{(with spin)} \tag{A.23}$$
$$(i|h|j) := \int \varphi_i^*(\boldsymbol{r}_1)h(\boldsymbol{r}_1)\varphi_j(\boldsymbol{r}_1)\mathrm{d}\boldsymbol{r}_1 \qquad \text{(just spacial)} \tag{A.24}$$

and we shall denote its elements by

$$h_{ij} = \langle i|h|j\rangle = (i|h|j)\,. \tag{A.25}$$

The last equality is due to fact that the one-electron Hamiltonian only acts on the spatial component of orbitals.

We can now use these rules to express the Coulomb and the exchange operator ((2.9) and (2.10))

$$\langle \phi_a(1)|J_b(1)|\phi_a(1)\rangle = \langle ab|ab\rangle \tag{A.26}$$
$$\langle \phi_a(1)|K_b(1)|\phi_a(1)\rangle = \langle aa|bb\rangle \tag{A.27}$$

If we take the electronic Hamiltonian (2.3), use slater determinants $|\Psi\rangle = |\phi_1 \dots \phi_N\rangle$ from (2.4) to build the expectation value $E_0 = \langle \Psi|H|\Psi\rangle$ and apply our notations from above, we obtain a very compact expression for the energy, namely (2.13),

$$E_0 = \sum_a \langle a|h|a\rangle + \frac{1}{2}\sum_{ab} \langle ab||ab\rangle \,,$$

where h denotes the one-electron Hamiltonian from (2.11).

A.3. Variation of Orbitals

In Section 2.1.4 we mention that we apply the variational method to derive (2.8), i.e. the Hartree-Fock equations. This is what we will do now. The Section follows the derivation by A. Szabo and N. S. Ostlund[14, p. 117-122].

We want to minimize the energy with respect to the orbitals while maintaining their orthonormality $\langle \phi_a|\phi_b\rangle = \langle a|b\rangle = \delta_{ab}$. We use Langrange multipliers to ensure this and thus receive the following target function to minimize

$$\begin{aligned} \mathcal{L}\left[\{\phi_a\}\right] = E_0[\{\phi_a\}] - \sum_{a=1}^{N}\sum_{b=1}^{N} \epsilon_{ab}\left(\langle a|b\rangle - \delta_{ab}\right) = \\ \sum_a \langle a|h|a\rangle + \frac{1}{2}\sum_{ab} \langle ab||ab\rangle - \sum_{a=1}^{N}\sum_{b=1}^{N} \epsilon_{ab}\left(\langle a|b\rangle - \delta_{ab}\right) \end{aligned} \tag{A.28}$$

We introduce a variation in the orbitals $\phi \to \phi + \delta\phi$ and apply it to (A.28) to obtain $\mathcal{L} \to \mathcal{L} + \delta\mathcal{L}$ with

$$\delta\mathcal{L} = \delta E_0 - \sum_{a=1}^{N}\sum_{b=1}^{N} \epsilon_{ab}\,\delta\left(\langle a|b\rangle - \delta_{ab}\right). \tag{A.29}$$

To minimize $\mathcal{L}$ we must set its variation zero, i.e.

$$\delta\mathcal{L} \stackrel{!}{=} 0, \tag{A.30}$$

but before we do that we will simplify the expressions variation of the energy and of our constraint. We will use the notation $\langle \delta a|b\rangle = \langle \delta\phi_a|\phi_b\rangle$, $\langle \delta(a)\, b|cd\rangle = \langle \delta(\phi_a)\,\phi_b|\phi_c\phi_d\rangle$ etc. as extension to the one from Appendix A.2. By writing out all of the terms for variation

of the energy and renaming some of the indices (details see Szabo & Oslund) we find for that some of the terms match. This gives us

$$\delta E_0 = \sum_{a=1}^{N} \langle \delta a | h | a \rangle + \frac{1}{2} \sum_{a=1}^{N} \sum_{b=1}^{N} \langle \delta(a)\, b | ab \rangle - \langle \delta(a)\, b | ba \rangle + \quad \text{c.c.} \quad , \tag{A.31}$$

where "c.c." denotes the complex conjugated version of whichever term precedes it. For the constraint we find

$$\begin{aligned} \sum_{ab} \epsilon_{ab} \left(\langle \delta a | b \rangle + \langle a | \delta b \rangle \right) = & \quad \textstyle\sum_{ab} \epsilon_{ab} \langle \delta a | b \rangle + \sum_{ab} \epsilon_{ab} \langle a | \delta b \rangle & \text{(A.32)} \\ = & \quad \textstyle\sum_{ab} \epsilon_{ab} \langle \delta a | b \rangle + \sum_{ab} \epsilon^*_{ab} \langle \delta a | b \rangle^* & \text{(A.33)} \\ = & \quad \textstyle\sum_{ab} \epsilon_{ab} \langle \delta a | b \rangle + \quad \text{c.c.} & \text{(A.34)} \end{aligned}$$

If we plug these results back into (A.30) and replace our short notations with the actual integrals we obtain

$$\begin{aligned} \delta \mathcal{L} = & \sum_{a=1}^{N} \int \delta \phi_a^*(1) \left[h(1) \phi_a(1) + \sum_{b=1}^{N} \left(J_b(1) - K_b(1) \right) \phi_a(1) - \sum_{b=1}^{N} \epsilon_{ab} \phi_b(1) \right] \mathrm{d}\boldsymbol{x}_1 \\ & + \text{ c.c.} \stackrel{!}{=} 0 \end{aligned} \tag{A.35}$$

Considering that the variation $\delta\phi_a(1)$ and thus $\delta\phi_a^*(1)$ is arbitrary, (A.35) can only hold if the expression in the brackets vanishes. Thus we receive

$$\left[h(1) + \sum_{b=0}^{N} \left(J_b(1) - K_b(1) \right) \right] \phi_a(1) = \sum_{b=1}^{N} \epsilon_{b=1}^{N} \phi_b(1) \quad \forall a \in \{1, \ldots, N\}. \tag{A.36}$$

As this is already very close to (2.8) we realize the term in brackets is the Fock operator from (2.12). To get from this generalized eigenvalue problem to an ordinary eigenvalue problem problem we must apply a rotation.

We now investigate how things change if the orbitals are rotated $\phi'_a = \sum_b \phi_b U_{ba}$. This is equivalent to rotating the matrix before taking the determinant in (2.4) (the Slater-Determinant). Due to the rules of determinants of a matrix product, $\det\{(AB)\} = \det\{A\} \det\{B\}$, we see that

$$\left| \Psi'_0 \right\rangle = \det\{\boldsymbol{U}\} \left| \Psi_0 \right\rangle . \tag{A.37}$$

Determinants of rotation matrices (as they are orthogonal and real) can only be ± 1. This means the rotated slater determinant only differs by a constant phase, which is irrelevant

to us[1]. Next we check the Coulomb operator,

$$\begin{aligned}\sum_a J'_a(1) &= \sum_a \int \phi'_a(2)^* \frac{1}{r_{12}} \phi'_a(2)\, \mathrm{d}\boldsymbol{x}_2 \\ &= \sum_{bc} \underbrace{\sum_a U^*_{ba} U_{ca}}_{(\boldsymbol{U}\boldsymbol{U}^\dagger)_{bc}=\delta_{bc}} \int \phi^*_b(2) \frac{1}{r_{12}} \phi_c(2) \mathrm{d}\boldsymbol{x}_2 \\ &= \sum_b \int \phi^*_b(2) \frac{1}{r_{12}} \phi_b(2)\, \mathrm{d}\boldsymbol{x}_2 \\ &= \sum_b J_b(1),\end{aligned} \tag{A.38}$$

and see that it does not change at all. The same is true for the Exchange operator. We have shown that the Fock operator is invariant under rotation, and does not change regardless of how we rotate the orbitals. Therefore, we will choose a rotation that will cause the Fock matrix (whose elements in the basis of the orbitals are the Lagrange multipliers, $\langle \phi_c | F | \phi_a \rangle = \sum_{b=1}^{N} \epsilon_{ba} \langle \phi_c | \phi_b \rangle = \epsilon_{ca}$) to be diagonal, i.e. we will choose a rotation $\boldsymbol{U}$, so that $\boldsymbol{\epsilon'} = \boldsymbol{U}^\dagger \boldsymbol{\epsilon} \boldsymbol{U}$ is a diagonal matrix with the diagonal elements $\epsilon_i,\ i \in \{1, \ldots, N\}$. The orbitals that result from this rotation are called canonical orbitals. What follows is (2.7).

A.4. Probability of Finding an Electron

The probability to find an electron at position $\boldsymbol{r}$ in our system of N electrons can be expressed via the density matrix $\boldsymbol{P}$. It is defined in our closed shell formalism (ψ_i are spatial orbitals) by

$$\rho(\boldsymbol{r}) = 2 \sum_{i=1}^{\frac{N}{2}} |\psi_i(\boldsymbol{r})|^2. \tag{A.39}$$

[1] A global phase of a state may be neglected as all physical properties are calculated using expectation values, where these phases cancel.

One can easily see that $\int \rho(\boldsymbol{r})\,\mathrm{d}\boldsymbol{r} = N$. If we plug in our basis expansion from (2.17), we obtain

$$\rho(\boldsymbol{r}) = 2\sum_{i=1}^{\frac{N}{2}} \psi_i^*(\boldsymbol{r})\psi_i(\boldsymbol{r}) \tag{A.40}$$

$$= 2\sum_{i=1}^{\frac{N}{2}} \sum_{\nu} C_{\nu i}^* \phi_\nu^*(\boldsymbol{r}) \sum_{\mu} C_{\mu i}\phi_\mu(\boldsymbol{r}) \tag{A.41}$$

$$= \sum_{\mu,\nu} \left(2\sum_{i=1}^{\frac{N}{2}} C_{\mu i} C_{\nu i}^* \right) \phi_\mu(\boldsymbol{r})\phi_\nu(\boldsymbol{r})^* \tag{A.42}$$

$$= \sum_{\mu,\nu} P_{\mu\nu}\phi_\mu(\boldsymbol{r})\phi_\nu(\boldsymbol{r})^*. \tag{A.43}$$

$$\tag{A.44}$$

In the last step we used the definition of the density matrix (2.26).

A.5. Expressing the Fock Matrix in Terms of the Density Matrix

Using the notation from Appendix A.2 we can write the Fock matrix as

$$F_{\mu\nu} = H_{\mu\nu}^{\mathrm{core}} + \sum_{i=1}^{\frac{N}{2}} \int \phi_\mu^*(1)\left[2J_i(1) - K_i(1)\right]\phi_\nu(1)\,\mathrm{d}\boldsymbol{r}_1 \tag{A.45}$$

$$= H_{\mu\nu}^{\mathrm{core}} + \sum_{i=1}^{\frac{N}{2}} 2(\mu\nu|ii) - (\mu i|i\nu) \tag{A.46}$$

$$= H_{\mu\nu}^{\mathrm{core}} + \sum_{i=1}^{\frac{N}{2}} \sum_{\lambda\sigma} C_{\lambda i} C_{\sigma i}^* \left[2(\mu\nu|\sigma\lambda) - (\mu\lambda|\sigma\nu)\right] \tag{A.47}$$

$$= H_{\mu\nu}^{\mathrm{core}} + \sum_{\lambda\sigma} P_{\lambda\sigma}\left[(\mu\nu|\sigma\lambda) - \frac{1}{2}(\mu\lambda|\sigma\nu)\right] \tag{A.48}$$

$$=: H_{\mu\nu}^{\mathrm{core}} + G_{\mu\nu}. \tag{A.49}$$

$$\tag{A.50}$$

Here we used the basis set expansion (2.17) and finally the definition of the density matrix (2.26).

B. Descriptor Models

In Chapter 4 we introduced symmetry functions, which are used to sample information regarding the molecular environment of an atom. A certain group of these functions, Gaussians and periodic Gaussians, are parametrized; in this chapter we shall depict the various parametrization models used in this thesis.

B.1. Radial Models

This section shows the models consisting of ordinary Gaussians, introduced in Section 4.2, and used to describe the radial part of the relative position of two atoms. A few of these models are characterized as "N equidistant points in an interval $[x, y]$". This describes a list $\{z_i : i \in \{1, \ldots, N\}, z_i = x + (y - x)i/N\}$, with x and y as real numbers. Note that $x > y$ is explicitly allowed. All numbers for widths and centers are in Å^{-1} and Å, respectively.

B.1.1. Origin Centered

The models in this section consist of Gaussians, that are all centered at 0.

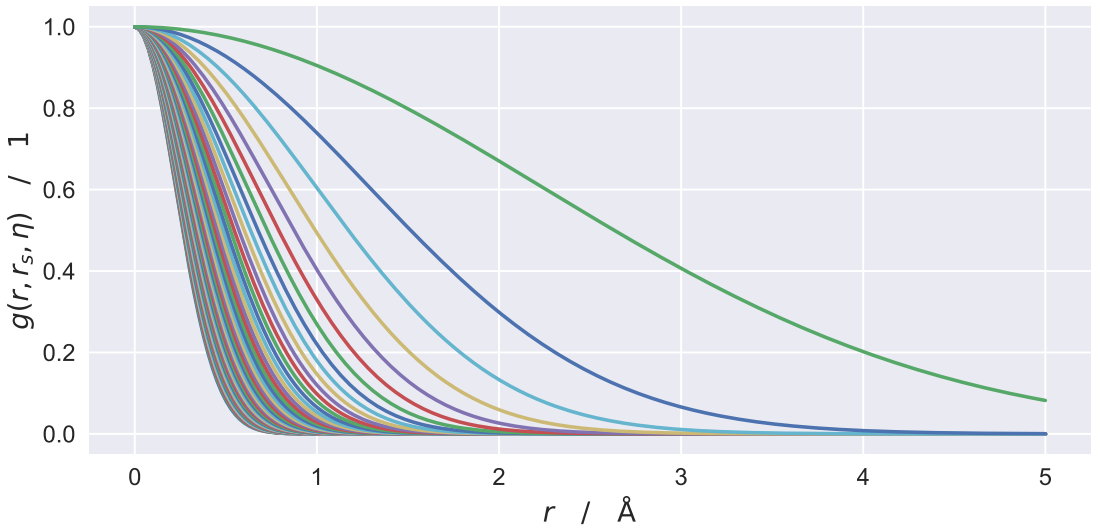


Figure B.1.: Set of radial symmetry functions of the "Origin-50" model. They are all centered at 0. The widths are 50 equidistant points in the interval $[10, 0.1]$.

B.1.2. Manually Tweaked

The models in this section are not built after any algorithm, but rather by intuition. This means the position and the width of the models is hand-crafted.

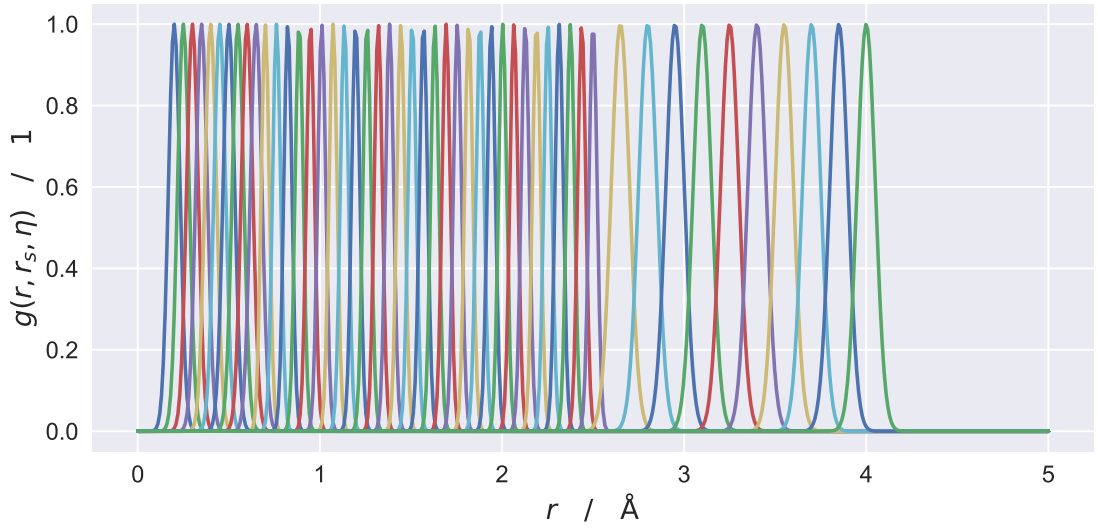


Figure B.2.: Set of radial symmetry functions of the "Man-50" model. The Gaussians have an increased concentration in the range 0.7-2.5 Å, in which all the bounding distances of ethene lie. The set consists of three subsets. The first group's is centers are 10 equidistant points from $[0.2, 0.67]$, and the width of 500 The Gaussians from the second group have centers that are 30 points from $[0.7, 2.5]$, at a width of 1000 and the ones from the last group have 10 points from $[2.7, 4]$ and a width of 200.

B.1.3. Uniformly Distributed

The models in this section consist of Gaussians, which are spaced equidistantly over a given interval. Their widths are monotonically increasing the further away from 0 the center of the Gaussian is; the values of the widths are equidistant points in a specified interval too.

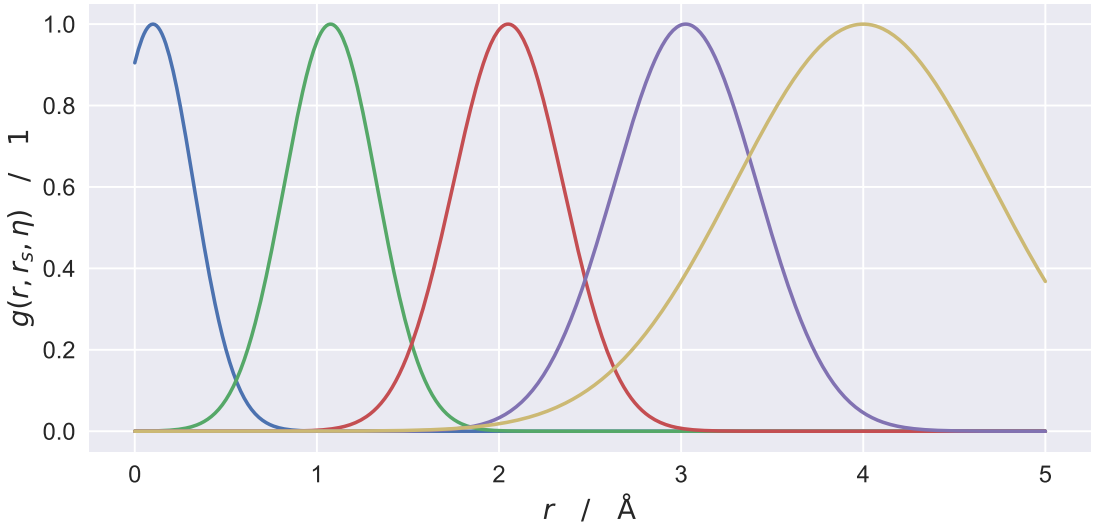


Figure B.3.: Set of radial symmetry functions of the "Unif-5" model. The centers/widths are 10 equidistant points from $[0.1, 4.0]/[10.0, 1.0]$.

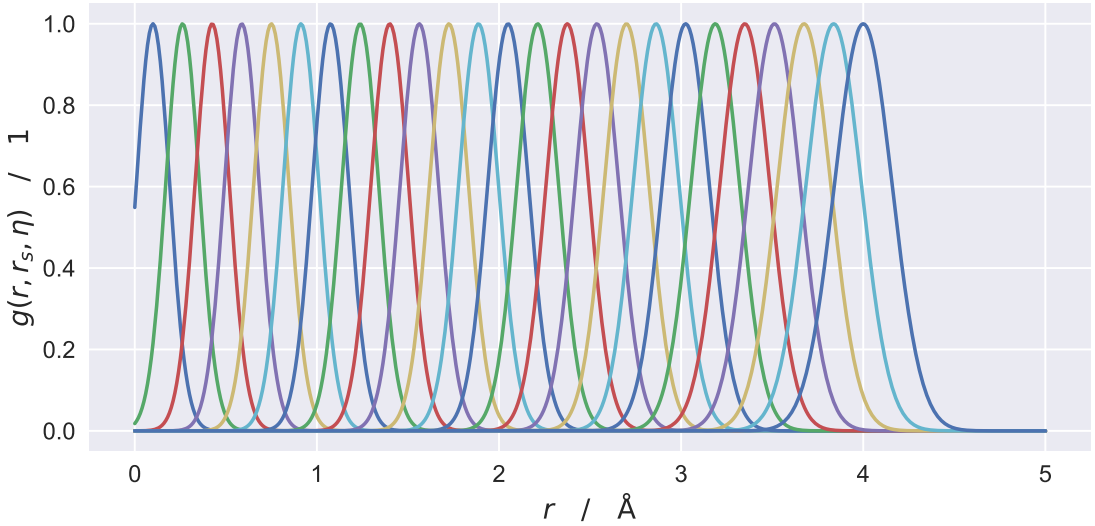


Figure B.4.: Set of radial symmetry functions of the "Unif-25" model. The centers/widths are 25 equidistant points from $[0.1, 4]/[60, 20]$.

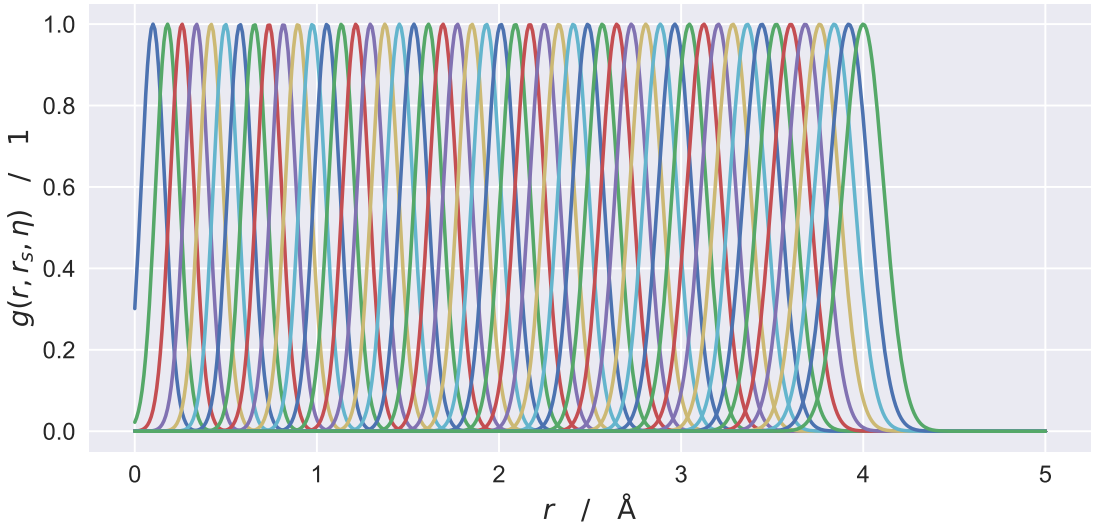


Figure B.5.: Set of radial symmetry functions of the "Unif-50" model. The centers/widths are 50 points from $[0.1, 4]/[120, 40]$.

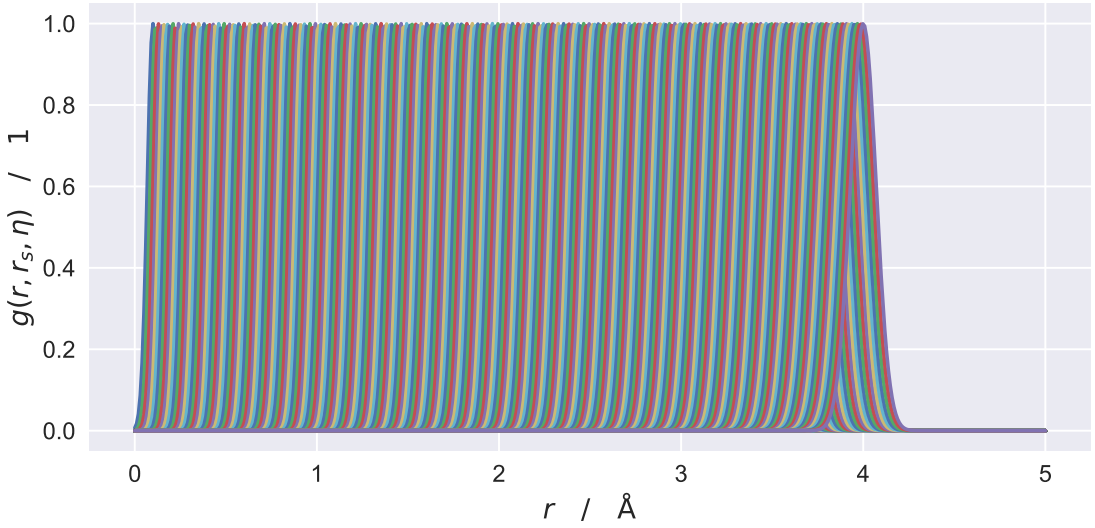


Figure B.6.: Set of radial symmetry functions of the "Unif-250" model. The centers/widths are 250 points from $[0.1, 4]/[500, 100]$.

B.2. Angular Models

Similarly to Section B.1.3, the models shown in this section are collections of (periodic) Gaussians, which are equidistantly spaced. In the context of our angular models, this means that for a model with N Gaussians periodic with a period P the centers will be $\left\{\frac{P}{N}i : i \in \{0, \ldots, N-1\}\right\}$, yielding a periodic distribution, symmetric around 0. Due to the isotropy of the angle-space all Gaussians will have the same width. They are chosen so that the periodic Gaussians overlap exactly at their points of inflection, i.e. at $\frac{2N^2}{P^2}$. The values for the centers and widths will thus not be given explicitly.

B.2.1. Azimuthal Models

These models are used to sample the azimuthal angle. They have a periodicity of 2π.

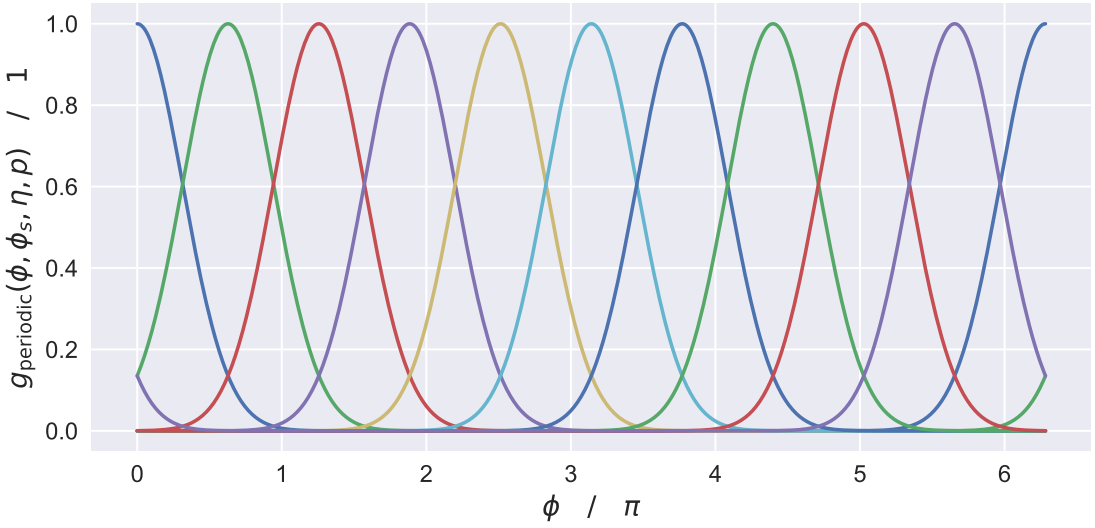


Figure B.7.: Set of angular symmetry functions of the azimuthal part of the "PG-10" model. It contains 10 periodic Gaussians.

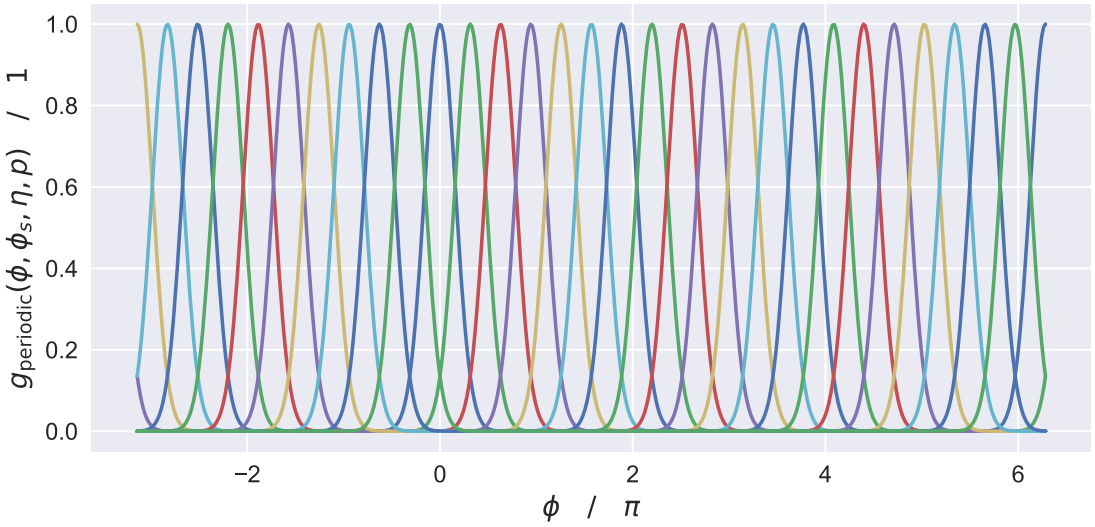


Figure B.8.: Set of angular symmetry functions of the azimuthal part of the "PG-20" model. It contains 20 periodic Gaussians.

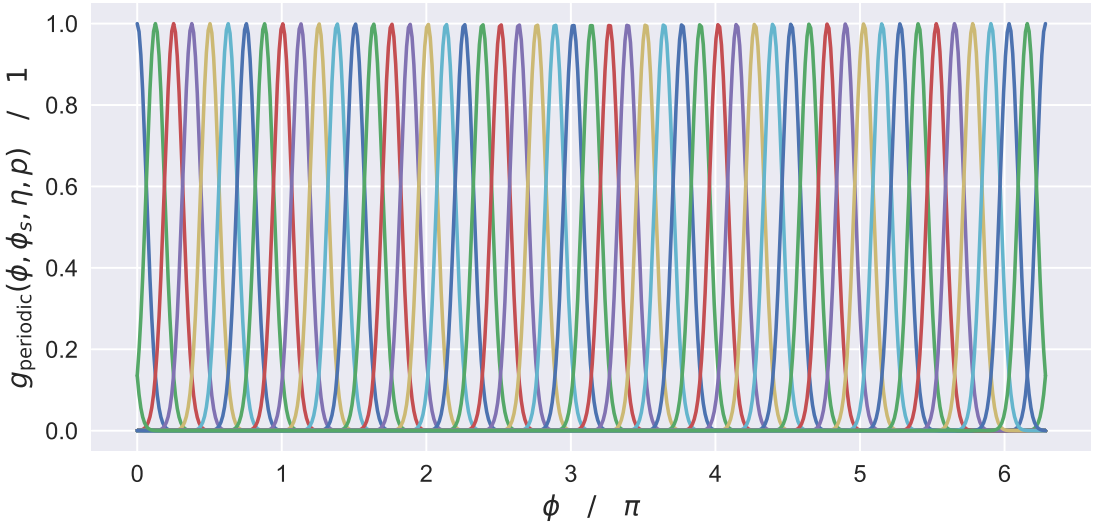


Figure B.9.: Set of angular symmetry functions of the azimuthal part of the "PG-20" model. It contains 20 periodic Gaussians.

B.2.2. Polar Models

These models are used to sample the polar angle. They have a periodicity of π.

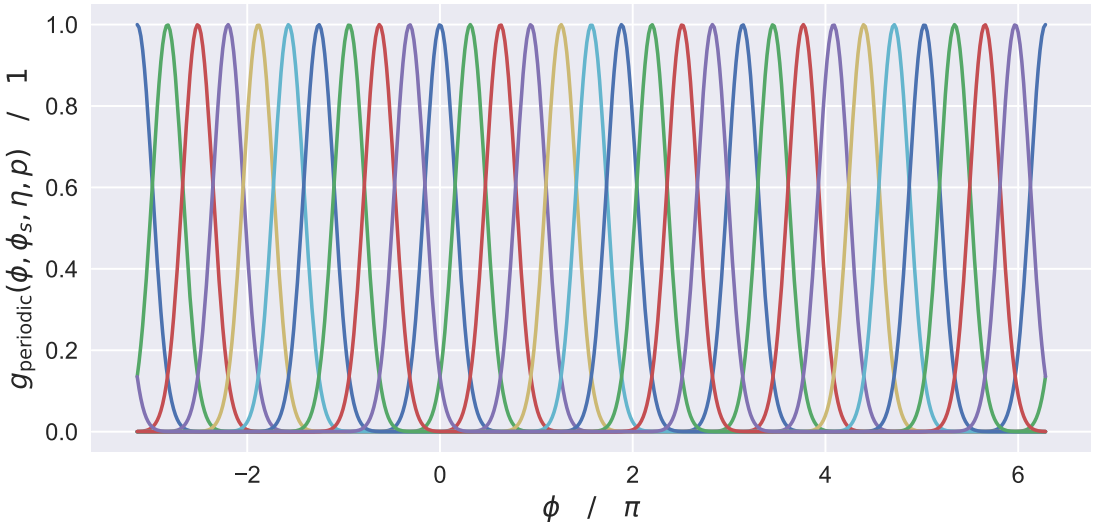


Figure B.10.: Set of angular symmetry functions of the polar part of the "PG-10" model. Note the similarity to B.7. However, due to the lower periodicity of only π, it is closer to B.8.

Bibliography

[1] F. Jensen, *Introduction to Computational Chemistry*. John Wiley & Sons, 2006.

[2] J. H. V. Lenthe, R. Zwaans, H. J. J. V. Dam, and M. Guest, "Starting SCF calculations by superposition of atomic densities," *Journal of Computational Chemistry*, vol. 27, no. 8, pp. 926–932, 2006.

[3] E. Jones, T. Oliphant, P. Peterson, *et al.*, "SciPy: Open source scientific tools for Python," 2001–. [Online; accessed 27.02.2018].

[4] T. E. Oliphant, *A guide to NumPy*, vol. 1. Trelgol Publishing USA, 2006.

[5] T. E. Oliphant, "Python for scientific computing," *Computing in Science Engineering*, vol. 9, pp. 10–20, May 2007.

[6] K. J. Millman and M. Aivazis, "Python for scientists and engineers," *Computing in Science Engineering*, vol. 13, pp. 9–12, Mar. 2011.

[7] J. D. Hunter, "Matplotlib: A 2D graphics environment," *Computing in Science Engineering*, vol. 9, pp. 90–95, May 2007.

[8] M. D. Hanwell, D. E. Curtis, D. C. Lonie, T. Vandermeersch, E. Zurek, and G. R. Hutchison, "Avogadro: an advanced semantic chemical editor, visualization, and analysis platform," *Journal of Cheminformatics*, vol. 4, p. 17, Aug. 2012.

[9] "Avogadro: an open-source molecular builder and visualization tool. version 1.1.1." https://avogadro.cc/. [Online; accessed 24.11.2018].

[10] Y. Shao, Z. Gan, E. Epifanovsky, A. T. Gilbert, M. Wormit, J. Kussmann, A. W. Lange, A. Behn, J. Deng, X. Feng, D. Ghosh, M. Goldey, P. R. Horn, L. D. Jacobson, I. Kaliman, R. Z. Khaliullin, T. Ku, A. Landau, J. Liu, E. I. Proynov, Y. M. Rhee, R. M. Richard, M. A. Rohrdanz, R. P. Steele, E. J. Sundstrom, H. L. W. III, P. M. Zimmerman, D. Zuev, B. Albrecht, E. Alguire, B. Austin, G. J. O. Beran, Y. A. Bernard, E. Berquist, K. Brandhorst, K. B. Bravaya, S. T. Brown, D. Casanova, C.-M. Chang, Y. Chen, S. H. Chien, K. D. Closser, D. L. Crittenden, M. Diedenhofen, R. A. D. Jr., H. Do, A. D. Dutoi, R. G. Edgar, S. Fatehi, L. Fusti-Molnar, A. Ghysels, A. Golubeva-Zadorozhnaya, J. Gomes, M. W. Hanson-Heine, P. H. Harbach, A. W. Hauser, E. G. Hohenstein, Z. C. Holden, T.-C. Jagau, H. Ji, B. Kaduk, K. Khistyaev, J. Kim, J. Kim, R. A. King, P. Klunzinger, D. Kosenkov, T. Kowalczyk, C. M. Krauter, K. U. Lao, A. D. Laurent, K. V. Lawler, S. V. Levchenko, C. Y. Lin, F. Liu, E. Livshits, R. C. Lochan, A. Luenser, P. Manohar, S. F. Manzer, S.-P. Mao, N. Mardirossian, A. V. Marenich, S. A. Maurer, N. J. Mayhall, E. Neuscamman, C. M. Oana, R. Olivares-Amaya, D. P. O'Neill, J. A. Parkhill, T. M. Perrine,

R. Peverati, A. Prociuk, D. R. Rehn, E. Rosta, N. J. Russ, S. M. Sharada, S. Sharma, D. W. Small, A. Sodt, T. Stein, D. Stück, Y.-C. Su, A. J. Thom, T. Tsuchimochi, V. Vanovschi, L. Vogt, O. Vydrov, T. Wang, M. A. Watson, J. Wenzel, A. White, C. F. Williams, J. Yang, S. Yeganeh, S. R. Yost, Z.-Q. You, I. Y. Zhang, X. Zhang, Y. Zhao, B. R. Brooks, G. K. Chan, D. M. Chipman, C. J. Cramer, W. A. G. III, M. S. Gordon, W. J. Hehre, A. Klamt, H. F. S. III, M. W. Schmidt, C. D. Sherrill, D. G. Truhlar, A. Warshel, X. Xu, A. Aspuru-Guzik, R. Baer, A. T. Bell, N. A. Besley, J.-D. Chai, A. Dreuw, B. D. Dunietz, T. R. Furlani, S. R. Gwaltney, C.-P. Hsu, Y. Jung, J. Kong, D. S. Lambrecht, W. Liang, C. Ochsenfeld, V. A. Rassolov, L. V. Slipchenko, J. E. Subotnik, T. V. Voorhis, J. M. Herbert, A. I. Krylov, P. M. Gill, and M. Head-Gordon, "Advances in molecular quantum chemistry contained in the q-chem 4 program package," *Molecular Physics*, vol. 113, no. 2, pp. 184–215, 2015.

[11] Q. Sun, T. C. Berkelbach, N. S. Blunt, G. H. Booth, S. Guo, Z. Li, J. Liu, J. D. McClain, E. R. Sayfutyarova, S. Sharma, S. Wouters, and G. K.-L. Chan, "PySCF: the Python-based simulations of chemistry framework," *Wiley Interdisciplinary Reviews: Computational Molecular Science*, vol. 8, no. 1, p. e1340.

[12] F. Chollet *et al.*, "Keras," 2015.

[13] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, S. Ghemawat, I. Goodfellow, A. Harp, G. Irving, M. Isard, Y. Jia, R. Jozefowicz, L. Kaiser, M. Kudlur, J. Levenberg, D. Mané, R. Monga, S. Moore, D. Murray, C. Olah, M. Schuster, J. Shlens, B. Steiner, I. Sutskever, K. Talwar, P. Tucker, V. Vanhoucke, V. Vasudevan, F. Viégas, O. Vinyals, P. Warden, M. Wattenberg, M. Wicke, Y. Yu, and X. Zheng, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015. Software available from tensorflow.org.

[14] A. Szabo and N. S. Ostlund, *Modern Quantum Chemistry: Introduction to Advanced Electronic Structure Theory*. Mineola: Dover Publications, Inc., first ed., 1996.

[15] T. Koopmans, "Über die Zuordnung von Wellenfunktionen und Eigenwerten zu den Einzelnen Elektronen Eines Atoms," *Physica*, vol. 1, no. 1, pp. 104 – 113, 1934.

[16] R. S. Mulliken, "Electronic population analysis on LCao–mo molecular wave functions. i," *The Journal of Chemical Physics*, vol. 23, no. 10, pp. 1833–1840, 1955.

[17] M. Wolfsberg and L. Helmholz, "The Spectra and Electronic Structure of the Tetrahedral Ions MnO4-, CrO4–, and ClO4-," *The Journal of Chemical Physics*, vol. 20, no. 5, pp. 837–843, 1952.

[18] R. Hoffmann, "An extended hückel theory. i. hydrocarbons," *The Journal of Chemical Physics*, vol. 39, no. 6, pp. 1397–1412, 1963.

[19] P. Pulay, "Convergence acceleration of iterative sequences. the case of scf iteration," *Chemical Physics Letters*, vol. 73, no. 2, pp. 393–398, 1980.

[20] P. Pulay, "Improved SCf convergence acceleration," *Journal of Computational Chemistry*, vol. 3, no. 4, pp. 556–560, 1982.

[21] D. Feller, "The role of databases in support of computational chemistry calculations," *Journal of Computational Chemistry*, vol. 17, no. 13, pp. 1571–1586, 1996.

[22] K. L. Schuchardt, B. T. Didier, T. Elsethagen, L. Sun, V. Gurumoorthi, J. Chase, J. Li, and T. L. Windus, "Basis set exchange: a community database for computational sciences," *Journal of chemical information and modeling*, vol. 47, no. 3, pp. 1045–1052, 2007.

[23] R. M. Balabin, "Enthalpy difference between conformations of normal alkanes: Intramolecular basis set superposition error (bsse) in the case of n-butane and n-hexane," *The Journal of Chemical Physics*, vol. 129, no. 16, p. 164101, 2008.

[24] J. Heaton, *Artificial Intelligence for Humans Volume 3: Deep Learning and Neural Networks*. Heaton Research, Inc., 2015.

[25] F. Rosenblatt, "The perceptron: a probabilistic model for information storage and organization in the brain.," *Psychological review*, vol. 65, no. 6, p. 386, 1958.

[26] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, no. 5, pp. 359–366, 1989.

[27] D. Clevert, T. Unterthiner, and S. Hochreiter, "Fast and accurate deep network learning by exponential linear units (elus)," *CoRR*, vol. abs/1511.07289, 2015.

[28] V. Nair and G. E. Hinton, "Rectified linear units improve restricted boltzmann machines," in *Proceedings of the 27th International Conference on International Conference on Machine Learning*, ICML'10, (USA), pp. 807–814, Omnipress, 2010.

[29] G. Klambauer, T. Unterthiner, A. Mayr, and S. Hochreiter, "Self-normalizing neural networks," *CoRR*, vol. abs/1706.02515, 2017.

[30] H. J. Kelley, "Gradient theory of optimal flight paths," *Ars Journal*, vol. 30, no. 10, pp. 947–954, 1960.

[31] A. E. Bryson, "A gradient method for optimizing multi-stage allocation processes," in *Proc. Harvard Univ. Symposium on digital computers and their applications*, vol. 72, 1961.

[32] D. E. Rummelhart, G. E. Hinton, and R. J. Willams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 9, pp. 533–536, 1986.

[33] S. Dreyfus, "The numerical solution of variational problems," *Journal of Mathematical Analysis and Applications*, vol. 5, no. 1, pp. 30 – 45, 1962.

[34] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *CoRR*, vol. abs/1412.6980, 2014.

[35] A. Krogh and J. A. Hertz, "A simple weight decay can improve generalization," in *Advances in neural information processing systems*, pp. 950–957, 1992.

[36] J. Behler and M. Parrinello, "Generalized neural-network representation of high-dimensional potential-energy surfaces," *Phys. Rev. Lett.*, vol. 98, p. 146401, Apr. 2007.

[37] A. P. Bartók, R. Kondor, and G. Csányi, "On representing chemical environments," *Phys. Rev. B*, vol. 87, p. 184115, May 2013.

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

[39] W. J. Hehre, R. F. Stewart, and J. A. Pople, "Self-Consistent Molecular-Orbital Methods. I. Use of Gaussian Expansions of Slater-Type Atomic Orbitals," *The Journal of Chemical Physics*, vol. 51, no. 6, p. 2657, 1969.

[40] R. McWeeny, "Some Recent Advances in Density Matrix Theory," *Rev. Mod. Phys.*, vol. 32, pp. 335–369, Apr. 1960.

[41] J. M. Martin and P. R. Taylor, "The geometry, vibrational frequencies, and total atomization energy of ethylene. a calibration study," *Chemical Physics Letters*, vol. 248, no. 5, pp. 336 – 344, 1996.

[42] J. Behler, "Representing potential energy surfaces by high-dimensional neural network potentials," *Journal of Physics: Condensed Matter*, vol. 26, no. 18, p. 183001, 2014.

[43] R. Krishnan, J. S. Binkley, R. Seeger, and J. A. Pople, "Self-consistent molecular orbital methods. XX. A basis set for correlated wave functions," *The Journal of Chemical Physics*, vol. 72, no. 1, pp. 650–654, 1980. (H,Li-Ne).

[44] A. Hurley, *Introduction to the Electron Theory of Small Molecules.* Academic Press, 1976.