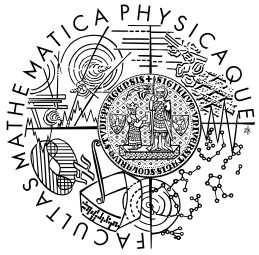


FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

BACHELOR THESIS

Ján Kovačovský

Non-additive intermolecular interactions

Department of Chemical Physics and Optics

Supervisor of the bachelor thesis: Mgr. Jiří Klimeš, Ph.D.
Study programme: Physics (B1701)
Study branch: General Physics

Prague 2023

I declare that I carried out this bachelor thesis independently, and only with the cited sources, literature and other professional sources. It has not been used to obtain another or the same degree.

I understand that my work relates to the rights and obligations under the Act No. 121/2000 Sb., the Copyright Act, as amended, in particular the fact that the Charles University has the right to conclude a license agreement on the use of this work as a school work pursuant to Section 60 subsection 1 of the Copyright Act.

In ..... date ....

Author's signature

I would like to thank my supervisor for all of his advices and willingness to help me with problems very quickly, even on weekends. I would also like to thank Filip Nicek for his support during my studies.

Title: Non-additive intermolecular interactions

Author: Ján Kovačovský

Department: Department of Chemical Physics and Optics

Supervisor: Mgr. Jiří Klimeš, Ph.D., department

Abstract: Non-additive intermolecular interactions, involving three or more molecules, present a significant challenge in accurately predicting molecular behaviour and properties of materials. These interactions, characterized by their high-dimensional and unpredictable nature with molecular position, cannot be reliably estimated by simply summing pairwise interactions between molecules. Incorporating three-body non-additive interactions into first-principles quantum mechanical approaches can greatly enhance agreement with experimental data, often achieving remarkable reductions in deviations.

Keywords: intermolecular interactions ab initio quantum mechanics

Contents

Introduction

Non-additive intermolecular interactions are interactions between three or more molecules that cannot be predicted by simply adding the pairwise interactions between the molecules. These interactions are high-dimensional and vary unpredictably with molecular position. Including three-body non-additive interactions in first-principles approaches can dramatically decrease the deviation from experiments, sometimes by an order of magnitude. Evaluating these interactions accurately using quantum mechanics is challenging, and becomes infeasible for large clusters. An alternative approach is to use intermolecular potentials to describe the interactions. Several types of intermolecular potentials exist, ranging from simple functions with general coefficients, commonly used to study biomolecules, to more complex models that fit potentials for specific molecules using high-quality reference data. While the former are fast, they are typically less accurate. The latter can achieve high accuracy if the model is reliable and are often designed to describe non-additive interactions, which are frequently omitted in simpler schemes. Non-additive terms describe the change in interaction energy of a molecular dimer due to the presence of a third molecule, and are crucial for both electrostatic and van der Waals interactions.

Chapter 1

Intermolecular interactions

In this work, we are interested in the stationary properties of molecules, such as equilibria geometry or vibrational frequencies. The studied system is then described by the time-independent Schrödinger equation

$$\hat{H}\Psi = E\Psi. \tag{1.1}$$

The Schrödinger equation (1.1) is an eigenvalue problem that relates the Hamiltonian operator $\hat{H}$ to its wave function Ψ and energy.

The Hamiltonian operator of a system can be expressed as a sum of the kinetic and potential energies of all its constituents. In the case of a molecular system, $\hat{H}$ can be further decomposed into a sum of nuclear and electronic terms

$$\begin{aligned}\hat{H} = &\underbrace{\sum_i \left(-\frac{1}{2m_e}\nabla_i^2\right)}_{\substack{\text{kinetic energy}\\ \text{of electrons}}} + \underbrace{\sum_k \left(-\frac{1}{2M_k}\nabla_k^2\right)}_{\substack{\text{kinetic energy}\\ \text{of nuclei}}} \\ &+ \underbrace{\sum_{i,k}\left(-\frac{Z_k}{r_{ik}}\right)}_{\substack{\text{electron-nuclear}\\ \text{attraction}}} + \underbrace{\frac{1}{2}\sum_{i\neq j}\frac{1}{r_{ij}}}_{\substack{\text{electron-electron}\\ \text{repulsion}}} + \underbrace{\frac{1}{2}\sum_{k\neq l}\frac{Z_k Z_l}{r_{kl}}}_{\substack{\text{nuclear-nuclear}\\ \text{repulsion}}},\end{aligned} \tag{1.2}$$

where r_{ij} denotes the distance between a pair of particles and Z_i their charge. The resulting Hamiltonian is expressed in atomic units.

The wave function Ψ encompasses all properties of the system, including the positions and movements of all particles, such as electrons and nuclei in a molecule.

To accurately describe the system, the wave function must satisfy certain requirements. Namely, it must be a solution to the Schrödinger equation and also respect the Pauli exclusion principle, no two fermions can occupy the same quantum state. In other words, a many-electron wave function must be antisymmetric with respect to the interchange of the spatial or spin coordinate among any two electrons. This means that if two electrons are interchanged, the wave function changes sign. The state space required to represent a system encompasses all possible superpositions of particles. This makes the quantum many-body problem computationally intractable, and therefore, it is necessary to use appropriate approximations.

Due to their relatively lower masses than the nuclei, electrons move at considerably higher speeds and can adiabatically (instantaneously) adapt to the movement of the nuclei, that is, on a typical timescale of the nuclear motion, the

electrons rapidly relax to the ground-state configuration. This concept is known as the *Born-Oppenheimer approximation* and enables us to study the electronic and nuclear parts of the system separately.

Within the Born-Oppenheimer approximation, the electronic subsystem is studied with fixed nuclei, i.e. the positions of nuclei are treated as constant parameters in the electronic Hamiltonian

$$\hat{H}_{\text{electronic}} = -\frac{1}{2}\sum_i \nabla_i^2 - \sum_{i,k}\frac{Z_k}{r_{ik}} + \frac{1}{2}\sum_{i\neq j}\frac{1}{r_{ij}} + \frac{1}{2}\sum_{k\neq l}\frac{Z_k Z_l}{r_{kl}}. \tag{1.3}$$

For a given configuration, the nuclear-nuclear interaction remains constant and thus does not alter the solution of the eigenfunction.

The total wave function can then be expressed as a product of nuclear and electronic terms

$$\Psi(\mathbf{r},\mathbf{R}) = \Theta_{\text{nuclear}}(\mathbf{R})\,\Phi_{\text{electronic}}(\mathbf{r};\mathbf{R}) \tag{1.4}$$

where $\Phi(r;R)$ is the solution to the electronic Schrödinger equation

$$\hat{H}_{\text{electronic}}\Phi = E_{\text{electronic}}\Phi, \tag{1.5}$$

and $\Theta(R)$ to the corresponding nuclear.

The electronic wave function Φ describes the motion of electrons and their rapid adaptation to any change in the nuclear position. Therefore, the eigenvalues $E_{\text{electronic}}$, called the adiabatic contribution of the electrons to the energy of the system, depend on the nuclear coordinates $\mathbf{R}$. This adiabatic energy acts as an effective potential energy for nuclear motion. Minima of this potential correspond to the equilibrium structures, whereas the saddle points describe transition states. The potential energy surface (PES) captures the relationship between the energy of a system and its geometry.

1.1 Ab initio methods

Ab initio electronic structure methods aim to calculate the many-electron function as a solution to the non-relativistic electronic Schrödinger equation (in the Born-Oppenheimer approximation). The many-electron function is generally a linear combination of many simpler electron functions.

To evaluate these approximations, the exact, known solutions can be used as a basis for the state space. This allows any approximate solution to be expressed as a linear combination of an exact solution, $\phi = \sum_i c_i\psi_i$, with $\sum_i c_i = 1$. The expectation value of Hamiltonian then is $\langle\phi|\hat{H}|\phi\rangle = \sum_i c_i^2 E_i$. Since the ground state energy E_0 is by definition the lowest eigenvalue of $\hat{H}$, the expected value must be greater or equal to it, and equality is only achieved when the trial wave function is equal to the ground state.

The variational principle states that the exact ground state wave function has the lowest possible energy and thus, any approximate solution yields higher energy, i.e. $\langle\Psi_0|\hat{H}|\Psi_0\rangle \leq \langle\Psi|\hat{H}|\Psi\rangle$. This principle can be used to find the ground state of a quantum system by minimizing the energy functional

$$\mathcal{E}[\Psi] = \frac{\langle\Psi|\hat{H}|\Psi\rangle}{\langle\Psi|\Psi\rangle}. \tag{1.6}$$

1.1.1 The Hartree-Fock model

The electronic Schrödinger equation can be solved exactly only for the hydrogen atom and similar one-electron systems. The idea behind the Hartree-Fock method is to write the many-body wave function as a product of single-particle spin orbitals $\chi(\mathbf{x})$, where $\mathbf{x}$ describes both spin and spatial distribution of the particle.

Electrons interact via electron-electron repulsion, meaning the many-body wave function must depend simultaneously on the coordinates of all electrons. Such a many-body wave function can be expressed in terms of a Slater determinant

$$\Phi(\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n) = \frac{1}{\sqrt{n!}} \begin{vmatrix} \chi_1(\mathbf{x_1}) & \chi_2(\mathbf{x_1}) & \cdots & \chi_n(\mathbf{x_1}) \\ \chi_1(\mathbf{x_2}) & \chi_2(\mathbf{x_2}) & \cdots & \chi_n(\mathbf{x_2}) \\ \vdots & \vdots & \ddots & \vdots \\ \chi_1(\mathbf{x_n}) & \chi_2(\mathbf{x_n}) & \cdots & \chi_n(\mathbf{x_n}) \end{vmatrix}, \tag{1.7}$$

which ensures that the Pauli exclusion principle is followed. The Slater determinant has n electrons occupying n spin orbitals $\chi_i, \chi_j, \ldots, \chi_k$ without specifying which electron is in which orbital.

This trial wave function can then be optimized via the variational principle 1.6 with a normalization condition $\langle\Phi|\Phi\rangle = 1$. The electronic Hamiltonian 1.3 can be written in terms of an isolated particle and two-electron interactions $\langle ij|ij\rangle = \sum_j \langle\phi_i\phi_j| \frac{1}{r_{ij}} |\phi_i\phi_j\rangle$, which can be interpreted as an interaction between the particle i and an effective field generated by an average position of other particles. The exchange integral, $\langle ij|ij\rangle$, emerges as a result of spin statistics and has no classical analogy.

The goal is to minimize

$$\mathcal{E}_{HF}[\phi_i] = \sum_i \langle i|\hat{h}|i\rangle + \frac{1}{2}\sum_{i,j}[\langle ij|ij\rangle - \langle ij|ji\rangle] - \sum_{ij}\varepsilon_{ij}(\langle\phi_i|\phi_j\rangle - \delta_{ij}), \tag{1.8}$$

where ε_{ij} are the Lagrange multipliers, constraining the optimization for normalized wave functions. It has the same effect as dividing the expectation values by the norm of the wave function, see equation 1.6. The use of Lagrange multipliers is usually preferred due to computational simplicity.

We can define the Fock operator for the energy terms in equation 1.8 as

$$\langle\phi_i|\hat{F}|\phi_k\rangle = \langle i|\hat{h}|k\rangle + \frac{1}{2}\sum_j \langle ij|kj\rangle - \langle ij|jk\rangle \tag{1.9}$$

and use it to rewrite the functional in eq. 1.8 as $\hat{F}\phi_i = \sum_j \varepsilon_{ij}\phi_j$. Since the Slater determinant is invariant to unitary transformation of orbitals, there always exists a canonical solution, such that

$$\hat{F}\phi_i = \varepsilon_i\phi_i \tag{1.10}$$

The Hartree-Fock equations 1.8 are solved iteratively until a predefined condition is met. With a complete basis set, the best possible solution for the Hartree-Fock energy can be obtained. This would be the case if the equations were to be solved iteratively but analytically. However, we are still restricted by some computational limits and thus need to use some well-behaved pre-defined functions as a basis. We can express the unknown orbitals in this basis $\psi_i = \sum_{\mu=1} c_{\mu i}\varphi_\mu$.

Substituting this expansion into eq. 1.10 and projecting the result onto the basis vector ϕ_μ, we get the Roothaan equations

$$\sum_\nu F_{\mu\nu} c_{\nu i} = \varepsilon_i \sum_\nu S_{\mu\nu} c_{\nu i}. \tag{1.11}$$

The closed-shell HF equations can be written in a more compact way by introducing the diagonal matrix $\boldsymbol{\varepsilon} = \mathrm{diag}\{\varepsilon_i\}$ and $\boldsymbol{C}$ matrix, which describes the spacial orbitals ϕ_i

$$\boldsymbol{FC} = \boldsymbol{SC\varepsilon}. \tag{1.12}$$

$\boldsymbol{S}$ is called the overlap matrix and describes the non-orthogonality of the chosen basis set. The matrix equation 1.12 is a generalized eigenvalue problem ($\boldsymbol{F}$ depends on the orbital coefficients), and thus needs to be solved iteratively.

1.1.2 Post-Hartree-Fock methods

The Hartree-Fock method is unsuitable for systems with multiple electronic configurations contributing significantly to the wave function. This is due to the method's assumption of a single determinant wave function, which fails to accurately represent such systems.

Ab-initio electronic structure calculations are susceptible to various sources of error. One such source arises from the inadequacy of the basis set. Another source stems from inaccuracies in modelling electron correlation. To address this, post-Hartree-Fock methods are employed to recover the correlation energy. The correlation energy denotes the disparity between the system's exact energy and the energy obtained through the Hartree-Fock approximation

$$E_{\text{correction}} = E_{\text{exact}} - E_{\text{HF}}. \tag{1.13}$$

Perturbation Theory

Perturbation theory is a mathematical approach used in quantum mechanics to approximate the solution of a complex problem by starting with the exact solution of a simpler, related problem. We introduce an ordering parameter λ and a small perturbation $\hat{V}$ into the reference Hamiltonian $\hat{H}_0$

$$\hat{H} = \hat{H}_0 + \lambda \hat{V}, \tag{1.14}$$

with λ being later set to one. For simplicity, suppose the perturbation to be time-independent and the wave function non-degenerate, thus we can consider the known eigenfunctions of $\hat{H}_0$ to form a complete orthonormal basis.

We can expand the exact eigenvalues and eigenfunction as a power series in λ

$$E_i(\lambda) = E_i^{(0)} + \lambda E_i^{(1)} + \lambda^2 E_i^{(1)} + \dots \tag{1.15}$$

$$\Phi_i(\lambda) = \Phi_i^{(0)} + \lambda \Phi_i^{(1)} + \lambda^2 \Phi_i^{(2)} + \dots \tag{1.16}$$

Substituting equations 1.15 and 1.16 into the stationary Schrödinger equation 1.1 with the perturbed Hamiltonian 1.14, and equating the coefficient λ^n yields

the following set of equations

$$(n=0): \quad \left(\hat{H}_0 - E_i^{(0)}\right)\Phi_i^{(0)} = 0 \tag{1.17}$$

$$(n=1): \quad \left(\hat{H}_0 - E_i^{(0)}\right)\Phi_i^{(1)} = \left(E_i^{(1)} - \hat{V}\right)\Phi_i^{(0)} \tag{1.18}$$

$$(n=2): \quad \left(\hat{H}_0 - E_i^{(0)}\right)\Phi_i^{(2)} = \left(E_i^{(1)} - \hat{V}\right)\Phi_i^{(1)} + E_i^{(2)}\Phi_i^{(0)} \tag{1.19}$$

$$\vdots$$

$$(n=\ell): \quad \left(\hat{H}_0 - E_i^{(0)}\right)\Phi_i^{(\ell)} = \left(E_i^{(1)} - \hat{V}\right)\Phi_i^{(\ell-1)} + E_i^{(2)}\Phi_i^{(\ell-2)} + \cdots + E_i^{(\ell)}\Phi_i^{(0)} \tag{1.20}$$

We can choose the normalization of Φ_i to be orthonormal to the unperturbed states, i.e. $\left\langle \Phi_i^{(0)} \middle| \Phi_i \right\rangle = 1$. From equation 1.16 we then obtain the orthogonality relations

$$\left\langle \Phi_i^{(0)} \middle| \Phi_i^{(n)} \right\rangle = 0 \quad \text{for} \quad n = 1, 2, 3, \ldots \tag{1.21}$$

Projecting the equations 1.17 - 1.20 onto the unperturbed wave functions, we obtain the n-th order energy corrections

$$E_i^{(0)} = \left\langle \Phi_i^{(0)} \middle| \hat{H}_0 \middle| \Phi_i^{(0)} \right\rangle \tag{1.22}$$

$$E_i^{(1)} = \left\langle \Phi_i^{(0)} \middle| \hat{V} \middle| \Phi_i^{(0)} \right\rangle \tag{1.23}$$

$$E_i^{(2)} = \left\langle \Phi_i^{(0)} \middle| \hat{V} \middle| \Phi_i^{(1)} \right\rangle \tag{1.24}$$

$$\vdots$$

Plugging equation 1.23 into 1.18 we obtain

$$\left(E_i^{(0)} - \hat{H}_0\right)\Phi_i^{(1)} = \left(\hat{V} - E_i^{(1)}\right)\Phi_i^{(0)}. \tag{1.25}$$

Using the fact that the zeroth-order wave functions are orthogonal, we can project the relation 1.25 onto $\Phi_j^{(0)}$, plug the result into the expression for the second order energy correction 1.24 to obtain the second-order energy contribution

$$E_i^{(2)} = \left\langle \Phi_i^{(0)} \middle| \hat{V} \middle| \Phi_i^{(1)} \right\rangle = \sum_j \left\langle \Phi_i^{(0)} \middle| \hat{V} \middle| \Phi_j^{(0)} \right\rangle \left\langle \Phi_j^{(0)} \middle| \Phi_i^{(1)} \right\rangle = \sum_{i \neq j} \frac{\left| \left\langle \Phi_i^{(0)} \middle| \hat{V} \middle| \Phi_j^{(0)} \right\rangle \right|^2}{E_i^{(0)} - E_j^{(0)}}. \tag{1.26}$$

Higher-order terms can be calculated analogously through an iterative process.

Møller-Plesset Perturbation Theory

Møller-Plesset perturbation theory (MP) is a post-Hartree-Fock ab initio method that improves on the Hartree-Fock method by adding electron correlation effects via the Rayleigh-Schrödinger perturbation theory. The zeroth-order wave function is an exact eigenfunction of the Fock operator, which serves as the unperturbed operator and electron correlation as perturbation. Since the Hartree-Fock energy is not the eigenvalue of the Fock operator, it is suitable to define the unperturbed Hamiltonian as a shifted Fock operator $\hat{H}_0 = \hat{F} - \langle \Phi_0 | \hat{H} - \hat{F} | \Phi_0 \rangle$. This way the eigenfunctions stay the same, but eigenvalues give the correct energy.

The zeroth-order energy is the expectation of $\hat{H}$ with respect to the unperturbed state Φ_0, i.e. the Hartree-Fock energy. First-order correction to energy

is identically zero (by construction). The first meaningful energy correction is given by the second-order perturbation. In order to obtain this correction for a closed shell molecule, we need to write the formula 1.24 in a basis of doubly excited Slater determinants. The Brillouin theorem states that for an optimized Hartree-Fock wave function ψ_0, the matrix element of the Hamiltonian between the ground state and a single excited determinant must be zero. Therefore, the singly-excited Slater determinants do not contribute.

$$E_0^{(2)} = \sum_{i,j>i} \sum_{a,b>a} \frac{\left|\left\langle \Phi_{ij}^{ab} \right| \hat{H} \left| \Phi_0 \right\rangle\right|^2}{E^{(0)} - E_{ij}^{ab}} = \sum_{i,j>i} \sum_{a,b>a} \frac{\left|\langle ij|ab\rangle - \langle ij|ba\rangle\right|^2}{\varepsilon_i + \varepsilon_j - \varepsilon_a - \varepsilon_b}. \tag{1.27}$$

The second-order Møller-Plesset (MP2) energy is always lower than that of Hartree-Fock due to the fact that virtual orbitals possess higher energy than occupied ones, resulting in a negative contribution. However, the energy is not an expectation value of the Hamiltonian, and as such, MP2 does not adhere to the variational principle.

Explicitly correlated methods, such as F12, are employed to address the issue of slow convergence of the wave function. It is based on the idea of including the interelectronic distance r_{12}, directly into the wave function, to more accurately describe the correlation energy. [1]

1.2 Intermolecular interactions

In quantum mechanics, charged particles are not considered to be point-like or rigid. Instead, atoms and molecules have an internal electronic structure that can change in different environments. This leads to non-additive interatomic forces, such as induction, dispersion, or repulsion. [Kaplan 2], [Stone 3]

The classification of intermolecular interactions is dependent on the distance between the interacting entities. It is important to note that all types of intermolecular interactions share the same underlying physical nature, which is electromagnetic. These interactions can be classified according to three ranges of interatomic separation, as defined by a typical interatomic potential. The first range encompasses short distances, where the potential exhibits a repulsive nature and electronic exchange dominates due to the overlap of molecular electronic shells. The second range includes intermediate distances, characterized by the presence of a van der Waals minimum resulting from a balance between repulsive and attractive forces. The third range encompasses large distances, where electronic exchange becomes negligible and intermolecular forces are predominantly attractive.

The pairwise additivity, as seen in the Lennard-Jones potential, is only the first approximation. The energy of an n-particle system can be expressed in terms of a dimer, trimer, tetramer, etc. contributions

$$E = E_1(n) + E_2(n) + E_3(n) + \cdots + E_n(n). \tag{1.28}$$

In many cases, this series converges rapidly and pairwise interactions are dominant.

1.2.1 Polarization

Polarization forces arise when the electron cloud of one atom is distorted by the electric field of another atom, leading to an attractive force between the two atoms. They can be derived from the Rayleigh-Schrödinger perturbation theory. Take the isolated systems as an unperturbed Hamiltonian $\hat{H}_0 = \hat{H}_A + \hat{H}_B$ and the perturbation as an electrostatic interaction between the particles of both of them

$$\hat{V} = \frac{1}{4\pi\epsilon_0} \frac{\hat{\rho}^A(\mathbf{r}_a)\hat{\rho}^B(\mathbf{r}_b)}{|\mathbf{r}_a - \mathbf{r}_b|} \,\mathrm{d}^3\mathbf{r}_a \,\mathrm{d}^3\mathbf{r}_b \,, \tag{1.29}$$

with a being a charged element of atom A at position $\mathbf{r}_a$ and b in atom B. $\hat{\rho}^X(\mathbf{r}) = \sum_{x\in X} Z_x \delta(\mathbf{r} - \mathbf{r}_x)$ is the charge density operator, which describes the electrostatic cloud created by the molecule.

It is suitable to express the complete electrostatic interaction using the charge density operator

$$\hat{V} = \int \hat{V}_B(\mathbf{r})\hat{\rho}_A(\mathbf{r}) \,\mathrm{d}\Omega \,, \quad \text{where} \quad \hat{V}_B(\mathbf{r}) = \frac{1}{4\pi\epsilon_0} \int \frac{\hat{\rho}_B(\mathbf{r}')}{|\mathbf{r} - \mathbf{r}'|} \,\mathrm{d}\Omega' \,.$$

The unperturbed states are simple product functions $\Phi_i^A \Phi_j^B$. The zeroth-order term is a sum of individual energies, first-order energy correction is just the expectation value of the electrostatic interaction for the ground state. For closed-shell molecules, the second-order of perturbation theory, equation 1.26, describes the polarization interaction between two molecules in their ground states

$$E^{(2)}_{\text{polarization}} = -\sum_{i,j} \frac{\left|\left\langle \Phi_i^A \Phi_j^B \right| \hat{V} \left| \Phi_0^A \Phi_0^B \right\rangle\right|^2}{(E_i^A - E_0^A) + \left(E_j^B - E_0^B\right)} = E^{(2)}_{\text{induction}} + E^{(2)}_{\text{dispersion}} \tag{1.30}$$

The quantum numbers i and j do not simultaneously take the values corresponding to the ground states of the isolated molecules. The summation over i and j may be divided into two parts, induction and dispersion. Components within the summation can be analysed individually wherein the molecule A is in an excited state while molecule B remains in its ground state, the components where the molecule B is in an excited state while molecule A remains in its ground state, and the components where both molecules are concurrently in an excited state.

Induction

If the molecules are apart far enough and the overlap between their wave functions can be ignored, the theory becomes relatively simple. For neutral atoms in the spherical-symmetry ground states, like noble gases, the induction forces are equal to zero if the overlap of the charge distribution of interacting atoms can be neglected. In the presence of the overlap, the induction forces are defined by equation 1.30 as

$$E^{(2)}_{\text{induction}} = -\sum_{j\neq 0} \frac{\left|\left\langle \Phi_0^A \Phi_j^B \right| \hat{V} \left| \Phi_0^A \Phi_0^B \right\rangle\right|^2}{E_j^B - E_0^B} - \sum_{i\neq 0} \frac{\left|\left\langle \Phi_i^A \Phi_0^B \right| \hat{V} \left| \Phi_0^A \Phi_0^B \right\rangle\right|^2}{E_i^A - E_0^A} \tag{1.31}$$

The first term describes the electrostatic interaction of the molecule A in its ground state with the induced electron density distribution of the molecule B and vice versa.

When the distance between molecules is significant, the induction energy can be effectively approximated by employing a multipole series. This approach involves expanding the potential energy $\hat{V}$ in a series of powers of $1/r$. The leading term in this expansion corresponds to the interaction of an induced dipole with the field of the inducing molecule. The distance dependence is given by the square of the corresponding dipole-multipole interaction. For example, the first term for an interaction between a polar and a neutral molecule has the $1/r^6$ dependence.

Dispersion

The dispersion energy is a quantum-mechanical phenomenon that arises from s fluctuations in electronic density. These fluctuations cause an instantaneous redistribution of electron density, leading to the creation of a non-zero mean dipole moment even in cases where the permanent dipole moment is zero, such as in nonpolar molecules and noble gas atoms. This instantaneous dipole moment can induce dipoles and higher-order moments in neighbouring molecules.

$$E^{(2)}_{\text{dispersion}} = -\sum_{i,j\neq 0} \frac{\left|\left\langle \Phi^A_i \Phi^B_j \right| \hat{V} \left| \Phi^A_0 \Phi^B_0 \right\rangle\right|^2}{\left(E^A_i - E^A_0\right) + \left(E^B_j - E^B_0\right)}. \tag{1.32}$$

This expression, which was first studied by Heitler and London [4], shows that the first term in the orientation-averaged dispersion energy is directly proportional to r^{-6}. The dispersion energy is commonly represented using a multipole expansion series

$$E^{(2)}_{\text{dispersion}} = -\sum_{n=6} \frac{c_n}{r^n}. \tag{1.33}$$

Dispersion forces are typically the dominant type of van der Waals forces between atoms and molecules. For example, in the case of a noble gas dimer such as Ne-Ne, the dispersion force is the only contributor to the interaction energy, meaning that the total interaction energy is equal to the dispersion energy. They are attractive for molecules in ground states.

Third-order perturbation correction to the dispersion energy was studied by Axilrod and Teller [5]

$$E^{(3)}_{\text{dispersion}}(ijk) = E_0 \frac{1 + 3\cos\gamma_i \cos\gamma_j \cos\gamma_k}{r^3_{ij} r^3_{ik} r^3_{jk}}, \tag{1.34}$$

where r_{ij} is the distance between atoms i and j, while γ_{ij} represents the angle formed by the vectors $\mathbf{r}_{ij}$ and $\mathbf{r}_{ik}$. The sign of the three-body interaction is given by the geometrical factor. For an equilateral triangle configuration leads this correction to repulsion, while for a linear molecule to attraction.

For an accurate description of short-range forces, it is necessary to damp the dispersion energy at internuclear distances where significant charge overlap occurs. This is required to account for the effects of charge penetration and exchange repulsion, which become increasingly important at shorter distances. To achieve this, damping functions such as

$$f_n(r;\ell,k) = \begin{cases} \exp\left\{-\ell\left(\frac{k}{r} - 1\right)^2\right\}, & r < k, \\ 1, & r \geq k \end{cases} \tag{1.35}$$

are employed. These damping functions tend to 1 as $r \to \infty$, and to zero as $r \to 0$. Each damping function $f_n(r)$ must suppress the corresponding r^{-n} singularity.

1.2.2 Repulsion

The repulsive forces present in a system have two primary sources. The first originates from the interaction between electrons, while the second arises due to the Pauli exclusion principle.

Interaction between molecules at long range can be suitably expressed using a multipole expansion in $1/r$, where r represents the distance between the molecules. This approach offers a clear and comprehensible framework for analyzing and manipulating long-range interactions.

However, at short range, the description of the interaction becomes considerably more intricate. This complexity arises due to the interplay of overlapping electron densities and the effects of electron exchange. The exchange energy is a consequence of the Pauli exclusion principle, which states that no two fermions can occupy the same quantum state. As we have seen before, the many-body wave function must be antisymmetric with respect to permutations of electrons. Consider two atoms A and B, being in the ground state with wave functions Φ_0^A and Φ_0^B. The wave functions are eigenfunctions of the isolated Hamiltonian for the corresponding atom, hence they both satisfy the antisymmetric principle. However, for the composed system, the total wave function $\Phi_0^A\Phi_0^B$ must be antisymmetric with respect to all permutations of electrons, that is, also between atoms. This condition can be fulfilled using the antisymmetrizing operator $\hat{A} = \frac{1}{N!}\sum_{P\in S_N}(-1)^\pi \hat{P}$

$$\Phi_0^{AB} = \hat{A}\Phi_0^A\Phi_0^B = N_{AB}\sum_{P\in S_N}(-1)^\pi \hat{P}\Phi_0^A\Phi_0^B, \tag{1.36}$$

where $\hat{P}$ is an operator permuting electrons between molecules, π is parity of the permutation and N_{AB} is a normalization factor.

The interaction energy in the first-order of perturbation theory is then

$$\begin{aligned} E^{(1)}_{\text{interaction}} &= \left\langle \Phi_0^{AB}\right| \hat{V} \left|\Phi_0^{AB}\right\rangle \\ &= N_{AB}^A \left[\left\langle \Phi_0^A\Phi_0^B\right| \hat{V} \left|\Phi_0^A\Phi_0^B\right\rangle + \left\langle \Phi_0^A\ \Phi_0^B\right| \hat{V}\Big| \sum_{P\neq\mathbb{1}} (-1)^\pi\ \hat{P}\Phi_0^A\ \Phi_0^B\right\rangle\right], \end{aligned} \tag{1.37}$$

The first term in brackets represents the electrostatic interaction, while the second term corresponds to the exchange energy. Böhm and Ahlrichs [6] found that the exchange energy for two many-electron atoms with closed shells can be approximated using a simple exponential function of the form $A\exp(-\alpha r)$. This implies that as the distance between the two atoms increases, the exchange energy approaches zero exponentially. However, at short distances, the exchange force is the dominant interaction. It is positive for interacting systems with closed electronic shells, such as noble gas atoms.

In the context of second-order perturbation theory, it is not possible to separate the contributions of exchange and polarization energies. Standard perturbation theory becomes incorrect, since the zeroth-order functions are not eigenfunctions of the total unperturbed Hamiltonian $\hat{H}_0 = \hat{H}_A + \hat{H}_B$, only the first-order

correction is valid. In the case of closed-shell systems, the impact of exchange contributions to the dispersion energy within the second-order perturbation theory is rather negligible.

In the context of interactions between neutral, spherical atoms, the induction component does not contribute to the total energy of the system. The primary non-additive effects are attributed to many-body repulsion and dispersion terms.

1.3 Model potentials

Molecular dynamics and Monte Carlo calculations require a description of the potential energy at arbitrary configurations. Ideally, such a potential surface would respect all the physical constraints, such as differentiability. These potential surfaces can be deduced from experimental data or electronic structure calculations. Nonetheless, procuring precise data is resource-intensive and often restricted to a finite set of geometries. In applications where the potential energy must be evaluated at numerous configurations, an efficient interpolation method is essential.

One approach is to assume a physically-motivated functional form with a small number of adjustable parameters. A well-chosen function may allow for effective fits to small amounts of data, however, finding such a function requires some intuition for all but the simplest systems. Such functions could also give the model unrealistic properties. The accuracy of interpolation may not improve by simply increasing the amount of data unless additional parameters are added to increase the flexibility of the model.

Semi-empirical model potentials, such as the Lennard-Jones potential, are widely used in computational chemistry due to their simplicity and computational efficiency. These potentials are approximate representations of the interactions between atoms or molecules and provide a reasonable compromise between accuracy and computational cost.

The Lennard-Jones potential is an interatomic pair potential describing electronically neutral atoms

$$V_{\text{Lennard-Jones}}(r) = 4\varepsilon \left[\left(\frac{\sigma}{r} \right)^{12} - \left(\frac{\sigma}{r} \right)^{6} \right], \tag{1.38}$$

where r is the distance between the interacting particles, $-\varepsilon$ is the minimum of this potential and σ describes the position of the minimum $r_{\min} = 2^{1/6}\sigma$.

The attractive term in the Lennard-Jones potential corresponds to the dipole-dipole dispersion interaction, while the r^{12} term is chosen for computational simplicity in molecular dynamics simulations. The repulsive term can be more accurately represented by an exponential function, which has a better theoretical justification

$$V_{\text{Buckingham}}(r) = Ae^{\alpha r} - \frac{\gamma}{r^6}. \tag{1.39}$$

The idea behind selecting the Buckingham-type potential to describe the two-body interaction is well-founded for atoms with closed electron shells. In such cases, these atoms lack multipole moments, leading to purely overlapped electrostatic and induction interactions, akin to the exchange case. Consequently, these interactions exhibit a short-range nature, permitting their approximation by the

same exponential function. Conversely, at considerable distances, where electron function overlap between interacting atoms becomes negligible, the interaction energy is primarily governed by dispersion forces.

The mentioned model potentials can capture only the pair interaction, however if we want to study more complicated structures, we need to include the spatial resolution of the interacting atoms. In the third-order perturbation approximation, this can be represented by the Axilrod-Teller dispersion term 1.34 employing the inscribed angles.

The exchange part of the three-body potential can be expressed in terms of symmetry adapted coordinates Q_1, Q_2 and Q_3, which are the following linear combinations of the internal distances,

$$Q_1 = \sqrt{\frac{1}{3}}\left(r_1 + r_2 + r_3\right), \quad Q_2 = \sqrt{\frac{1}{2}}\left(r_2 - r_3\right) \quad \text{and} \quad Q_3 = \sqrt{\frac{1}{6}}\left(2r_1 - r_2 - r_3\right). \tag{1.40}$$

The potential must be symmetric to the exchange of the same type of atoms. This is achieved by using only the totally symmetric combinations, i.e. Q_1, $Q_2^2 + Q_3^2$ and $Q_3^3 - 3Q_3Q_2^2$. [7]

The analytic form of the three-body potential is taken as a sum of an exchange term, expressed as a polynomial in Q coordinates multiplied by an exponentially decaying function of Q_1

$$\begin{aligned} V^{(3)}_{\text{exchange}} = \{&c_0 + c_1Q_1 + c_2Q_1^2 + \left(c_3 + c_4Q_1 + c_5Q_1^2\right)\left(Q_2^2 + Q_3^2\right) \\ &+ \left(c_6 + c_7Q_1 + c_8Q_1^2\right)\left(Q_3^3 - 3Q_3Q_2^2\right) \\ &+ \left(c_9 + c_{10}Q_1 + c_{11}Q_1^2\right)\left(Q_2^2 + Q_3^2\right)^2 \\ &+ \left(c_{12} + c_{13}Q_1 + c_{14}Q_1^2\right)\left(Q_2^2 + Q_3^2\right) \\ &\times \left(Q_3^3 - 3Q_3Q_2^2\right)\} \exp\{(-\alpha Q_1)\}. \end{aligned} \tag{1.41}$$

The three-body model potential is then given as a sum of exchange and dispersion terms

$$E^{(3)} = V^{(3)}_{\text{exchange}} + F(r_1, r_2, r_3)V^{(3)}_{\text{dispersion}}, \tag{1.42}$$

where $F(r_1, r_2, r_3)$ is a product of three damping functions 1.35.

The polynomial representation quickly gets out of hand, posing a challenging task in interpreting the coefficients. To account for higher-order contributions or other physical properties, a more versatile functional approximator, such as a neural network, may be employed.

The parameters of these model potentials are not representative of real many-body interaction parameters. Instead, their values incorporate higher-order contributions. Consequently, these parameters cannot be directly related to specific physical properties.

Optimization

In order the model potentials describe the physical properties of the system with good precision or be suitable to be used in molecular dynamics, they need to be at least twice differentiable. Therefore, we can use optimization algorithms which

use gradients or hessians. This is not strictly true, when optimizing the models, we are differentiating the model with respect to its parameters. However, when computing forces we are differentiating the model with respect to the inputs, or more generally with respect to the Cartesian coordinates of atoms.

The analytic functional representation of the model potential is usually determined by fitting to ab initio reference points on the potential energy surface. The goodness of the fit is given by the *loss function*[1] $\mathcal{L}$, which characterizes the deviation of the model potential from the ab initio data. The loss function is typically chosen to be the norm in an appropriate function space. For example, the *mean squared error*

$$\mathcal{L}_{\text{MSE}}\left[\hat{f}_{\text{reference}}(\boldsymbol{x}), f_{\text{model}}(\boldsymbol{x};\boldsymbol{w})\right] = \frac{1}{n}\sum_{i=1}^{n}\left(\hat{f}_{\text{reference}}(\boldsymbol{x}_i) - f_{\text{model}}(\boldsymbol{x}_i;\boldsymbol{w})\right)^2. \quad (1.43)$$

The loss function can be thought of as a functional, depending on the input data, model function and model parameters. For simplicity, we will denote only the relevant parts of its argument.

For systems consisting of multiple atoms, the potential energy surface is a multidimensional surface with numerous local minima. Mathematically, this problem can be reduced to minimizing a function of several variables with multiple minima. It is not feasible to solve such a by examining all local minima, as their number increases exponentially with the number of atoms.

Heuristic methods for finding the global minimum typically employ two fundamental strategies: an iterative improvement approach and a "divide-and-conquer" strategy.

Genetic algorithms are inspired by the processes of natural selection from nature or statistical physics. Initially, a predetermined number of parameters within the search space is randomly selected. Each set of parameters is then evaluated for its suitability as a solution according to the objective function. Best-ranked sets are selected for breeding to produce a new population. The algorithm is reiterated until a specified condition is met.

Descent direction search algorithms should reduce the objective function at every step they take, that is, $\mathcal{L}\left[f(\boldsymbol{x};\boldsymbol{w}_{n+1})\right] \leq \mathcal{L}\left[f(\boldsymbol{x};\boldsymbol{w}_n)\right]$. To find a suitable direction, they often utilize the gradient of the objective function.

Gradient-based algorithms cannot typically distinguish local and global minima and so should be iterated over a range of initial conditions, be it random values of model parameters or even slightly different functional representations of the model potential.

The simplest of these algorithms is called the *steepest descent*, which optimizes the objective function according to the following update rule

$$\boldsymbol{w}_{n+1} = \boldsymbol{w}_n - \gamma\nabla_{\boldsymbol{w}}\mathcal{L}(\boldsymbol{w}_n), \quad (1.44)$$

where γ is called the learning rate and determines the step length in the direction of the gradient. Sufficiently small choices of the learning rate allow the loss function to converge to a local minimum. However, in flat regions, the algorithm would take unnecessarily many steps in a similar direction. Conversely, if γ is

[1] Also called the cost function or objective function.

chosen too large, the minimum may be overshot and algorithm become unstable, either oscillating or even moving away from the minimum.

To prevent unwanted behaviour and speed up the optimization process, the learning rate parameter can be set during every step of the optimization according to the current curvature of the region. This method is called the *line search method* [8], however, it is often sufficient to try out a small number of different values, for example, from a geometric progression $10^{-4} \sim 10^2$.

Newton's method chooses the step for the parameters in such a way as to minimize a second-order Taylor expansion of the loss function

$$\mathcal{L}[\hat{f}(\boldsymbol{x}), f(\boldsymbol{x}; \boldsymbol{w}+\boldsymbol{u})] \approx \mathcal{L} + \nabla_{\boldsymbol{w}}\mathcal{L}(\boldsymbol{w})\boldsymbol{u} + \frac{1}{2}\boldsymbol{u}^T\boldsymbol{H}(\boldsymbol{w})\boldsymbol{u}, \tag{1.45}$$

with $\boldsymbol{H}$ being the Hessian matrix of second derivatives of the loss function with respect to the model parameters and $\boldsymbol{u}$ is the parameter update. Differentiating this equation with respect to $\boldsymbol{u}$, noting that the optimal set of parameters should have the gradient equal to zero, we obtain $0 = \nabla_{\boldsymbol{w}}\mathcal{L} + \boldsymbol{H}(\boldsymbol{w})\boldsymbol{u}_{\text{optimal}}$. Rearranging the terms gives the update rule for Newton's method

$$\boldsymbol{w}_{n+1} = \boldsymbol{w}_n - \frac{1}{\boldsymbol{H}(\boldsymbol{w}_n)}\nabla_{\boldsymbol{w}}\mathcal{L}(\boldsymbol{w}_n). \tag{1.46}$$

Note that the update rule has no free parameter, the length of the step is given by the Hessian. Newton's method employs second-order information about the objective function, i.e. the curvature. It can be geometrically interpreted as the process of approximating the graph of the function by fitting a parabola with the same slope and curvature as the graph at that point. In other words, it finds the minimum of a parabola in just one step.

For the Hessian to be invertible, the loss function must be convex. This, however, is often not the case for regions far from the optimum. One way to ensure convergence is to approximate the Hessian with a suitable positive-definite matrix. Quasi-Newton methods such as the Levenberg-Marquardt algorithm approximate the Hessian with Jacobian, $\boldsymbol{H} \approx \boldsymbol{J}^T\boldsymbol{J}$. To make sure the approximated Hessian matrix is invertible, the Levenberg-Marquardt algorithm introduces a damping parameter μ

$$\boldsymbol{H} \approx \boldsymbol{J}^T\boldsymbol{J} + \mu\boldsymbol{I}, \tag{1.47}$$

with μ being large enough to make the matrix positive-definite.

Plugging the approximated Hessian into Newton's update rule (eq. 1.46) we get

$$\boldsymbol{w}_{n+1} = \boldsymbol{w}_n - \frac{1}{\boldsymbol{J}^T\boldsymbol{J} + \mu\boldsymbol{I}}\nabla_{\boldsymbol{w}}\mathcal{L}(\boldsymbol{w}_n). \tag{1.48}$$

Levenberg-Marquardt algorithm is a combination of the steepest descent and Newton's algorithm. When the damping parameter is small, the equation 1.48 approaches the update rule for Newton's algorithm. For large values of μ, the Levenberg-Marquardt approaches the steepest descent, and the inverse of the damping parameter can be interpreted as the learning rate $\gamma = \frac{1}{\mu}$. During every step of the optimization is the damping parameter μ selected using a line-search method. [9]

Above in figure 1.1, we show a comparison between gradient descent and Levenberg-Marquardt algorithms. The objective function was chosen as the

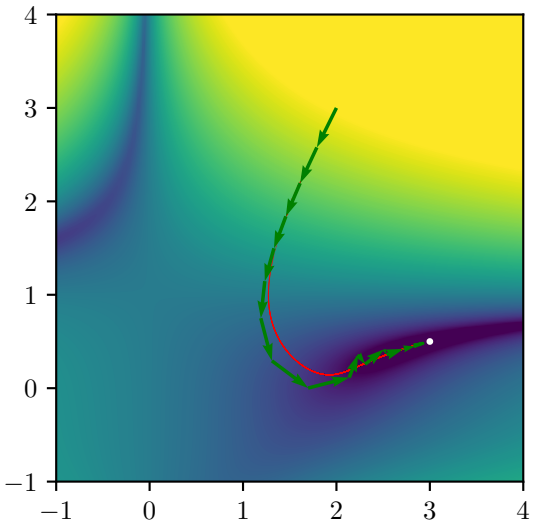


Figure 1.1: LM green, GD red

Beale's function for its shallow minima and steep maxima. The step-size for gradient descent was chosen very small so as to compensate for the very steep surface at the beginning. On the other hand, when the GD reached close to the optimal value, the region was very shallow and the optimization method spent a considerable number of steps in the same direction. LM took advantage of the surface curvature, and it took him lower tens of steps, whereas for GD it took him about 10^5 steps to reach the optimal value.

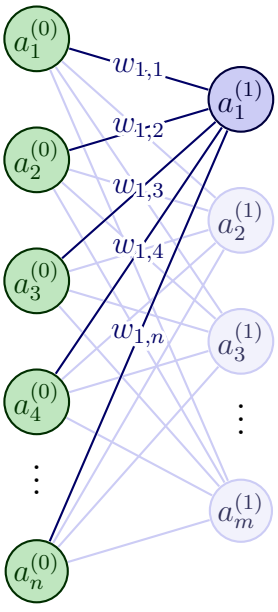
Gradient-based optimization algorithms converge typically faster than genetic algorithms, however, they have one fatal flaw, they fail to optimize when the gradient is zero, i.e. the update rule keeps the parameters unchanged. As we have seen in the previous section, many parts of the model potentials would return zero gradients, be it the cut-off function or the damping function. It is often sufficient to fine-tune these hyperparameters using a simple grid search.

Chapter 2

Neural Networks

A neural network is a type of machine learning algorithm that can be used to fit non-linear functions. It consists of layers of interconnected nodes, where each node performs a simple computation on its inputs and passes the result to the next layer. The connections between the nodes have associated weights, which are adjusted during training to minimize the error between the predicted and actual outputs.

In terms of non-linear function fitting, a neural network can be thought of as a universal function approximator [10]. Given a set of input-output pairs, the neural network can learn to approximate the underlying function that maps the inputs to the outputs. This is achieved by adjusting the weights of the connections between the nodes in such a way that the error between the predicted and actual outputs is minimized.



$$\begin{pmatrix} a_1^{(1)} \\ a_2^{(1)} \\ \vdots \\ a_m^{(1)} \end{pmatrix} = \sigma \left[\begin{pmatrix} w_{1,0} & w_{1,1} & \dots & w_{1,n} \\ w_{2,0} & w_{2,1} & \dots & w_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ w_{m,0} & w_{m,1} & \dots & w_{m,n} \end{pmatrix} \begin{pmatrix} a_1^{(0)} \\ a_2^{(0)} \\ \vdots \\ a_n^{(0)} \end{pmatrix} + \begin{pmatrix} b_1^{(0)} \\ b_2^{(0)} \\ \vdots \\ b_m^{(0)} \end{pmatrix} \right]$$

$$a^{(1)} = \sigma\left(\mathbf{W}^{(0)} a^{(0)} + \mathbf{b}^{(0)}\right), \quad \mathbf{W}^{(0)} \in \mathbb{R}^{m \times n}$$

Above is shown a typical fully-connected layer, that is, a layer in which all neurons are connected with all neurons from the previous layer. The input into a neuron is linearly scaled with its internal parameters $\boldsymbol{w}$, and then an activation function σ is applied. Mathematically, the computation of the whole layer can be compressed into a simple matrix multiplication.

2.1 Activation functions

The physical nature of the studied problem restrains the sample space of possible activation functions to those which are at least continuously differentiable. However, infinitely differentiable activation functions allow us to obtain smooth potential energy surfaces, force fields, and second derivatives which are required for training with forces as well as the calculation of vibrational modes.

Examples of such functions are the *Softplus* activation function, hyperbolic tangent *tanh* or Gaussian Error Linear Units *GELU*.

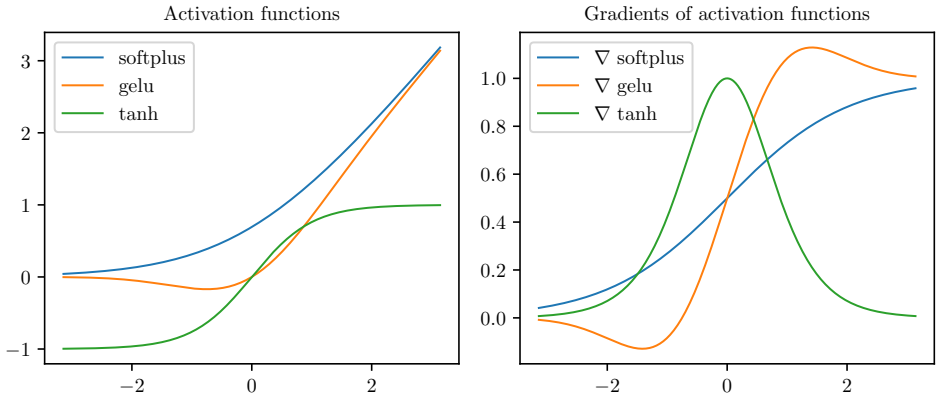


Figure 2.1: Softplus, tanh and GELU activation functions and their gradients.

Despite being the preferred option, the Softplus function is not without its limitations. As with any unbounded activation function, suffers from exploding gradient while its gradient, the sigmoid, suffers from the vanishing gradient problem. Although the issues mentioned are not significant for the shallow neural networks utilized in this work, they should be taken into consideration when choosing an activation function.

The issue of exploding gradients arises due to the nature of unbounded functions, such as Softplus, where large inputs result in even larger outputs. This can pose problems with learning even for relatively shallow neural networks. This issue can be partially dealt with by using advanced techniques such as *batchnorm layer* directly after this activation function of *gradient clipping*.

Saturating activation functions, such as tanh, can suffer from the vanishing gradient problem, where small gradients for large inputs hinder further training of the neural network. Since the gradients control how much the network learns during training, if the gradients are very small or zero, then little to no training can take place, leading to poor predictive performance.

The selection of activation functions must be carefully considered to accurately reflect the physical properties of the system being studied. For instance, in the case of a simple Coulomb potential, $\propto 1/r$, is convex on any sub-interval. This raises the question of whether our model should also be represented by a convex function.

In the plot below we compare training on a Coulomb potential for a shallow neural network using tanh and Softplus as activation functions for hidden layers. The Softplus function has been shown to achieve lower loss for energy predictions, and its convexity allows for the prediction of forces over a larger out-of-sample

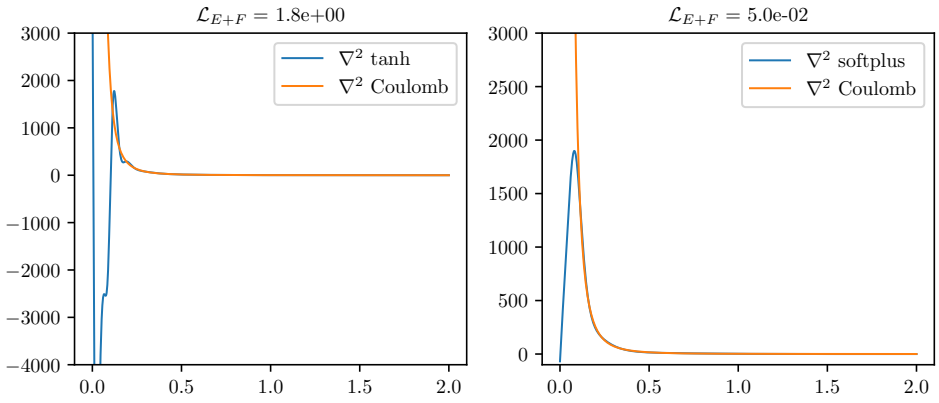


Figure 2.2: Comparison of convexity between the model and the underlying physical system.

interval. While it may appear to be the ideal candidate, the difference in energy loss compared to tanh is relatively small. It is the Hessian that highlights the significant difference between them.

The scale of the output variable must match the range of the activation function in the output layer. Otherwise, the network is impossible to converge in this region, similar to fitting nonzero energy outside the cutoff radius r_C. Therefore, in the last layer, we use a linear activation function and set the bias to represent the one-body contribution to the energy for the studied element.

2.2 Optimization

Just as with any nonlinear continuous function, the parameters of a neural network can be optimized by a gradient-based algorithm. Due to the high flexibility of neural networks, it is recommended to use a network with a minimal number of weighted connections between nodes to reduce the likelihood of overfitting, which can occur when insufficient data is available for optimal weight adjustment. The training time required for large neural networks can be considerable, as the number of calculations performed during each weight update using the Levenberg-Marquardt algorithm is proportional to the square of the number of connection weights.

Typical choices for the optimization algorithm for shallow neural networks or in general, models with a moderate number of parameters, are the Levenberg-Marquardt algorithm or the extended Kalman filter. Both of these methods are quasi-Netwon, meaning they scale quadratically with the number of parameters and number of samples in the batch. Blank and Brown [11] proposed to screen the sample space during the optimization process and avoid data containing redundant information. Only the data points with a significant contribution to the loss function are considered, effectively reducing the matrix size (see eq. 1.48). Another possibility is to monitor the gradients, which are being accumulated into the parameters during the backpropagation, and freeze the weights with a small contribution to the overall variance.

From the ab initio methods, we can obtain not only the potential energy surface but also the corresponding forces and since the model function is differ-

entiable with respect to its inputs, we can train on both sets of data.

The individual forces can be obtained from the model output by differentiating the output with respect to the atomic coordinates

$$\hat{F} = -\nabla_x \hat{E} = -\sum_i \nabla_x G_i \nabla_{G_i} \hat{E}, \tag{2.1}$$

where G are the input functions. Gradients of the input functions can be calculated before the training process and since during training this part of the computational graph stays unchanged, we only need to differentiate the output with respect to the input function.

The simultaneous training on both energies and forces has been shown to improve the quality of the fit and also prevent the function from overfitting. We can think of this as training on a dataset with implicit regularization. Below we compare training on energies and forces with training on energies with L^2 regularization technique for 50 random shallow neural networks.

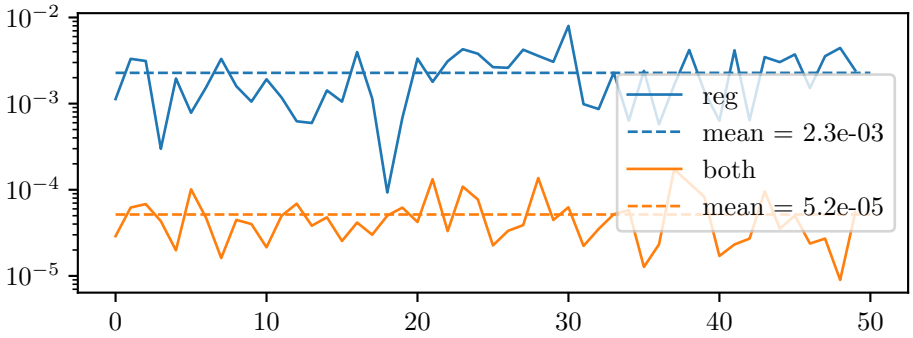


Figure 2.3: Comparison on using force as an implicit regularization technique versus the standard L^2.

When training using such an augmented dataset, it is important to bear in mind, that there are significantly many more force components than energies. This is due to the fact that we are usually interested in approximating the total energy of a system, whereas the forces are counted per atom. Therefore, if we are interested primarily in energy estimation, we need to penalize the force component in the loss function

$$\mathcal{L}_{\text{total}} = \mathcal{L}_{\text{energy}} + \frac{\rho}{n}\mathcal{L}_{\text{force}}, \tag{2.2}$$

where ρ is a scaling factor that dictates the relative significance of energy and force errors. Pukrittayakamee [12] proposed the value of ρ as λ/η^2, where λ is a hyperparameter typically set to 10^4 and η is the ratio of the maximum absolute value of force to the maximum absolute value of the potential energy.

The relative importance of force in the loss function can be small, however, when optimizing a function with steep sections, the optimization process can be led astray, especially in the first epochs of training. Even though the models are asymptotically expected to reach the same optimum, it is better to introduce the force incrementally. Below, we compare training on a fixed number of epochs between constant ρ and linearly increasing importance.

Integrating a differential equation formally solves the problem up to a constant. A model that learns only on forces can integrate the loss and match the energy up to a constant.

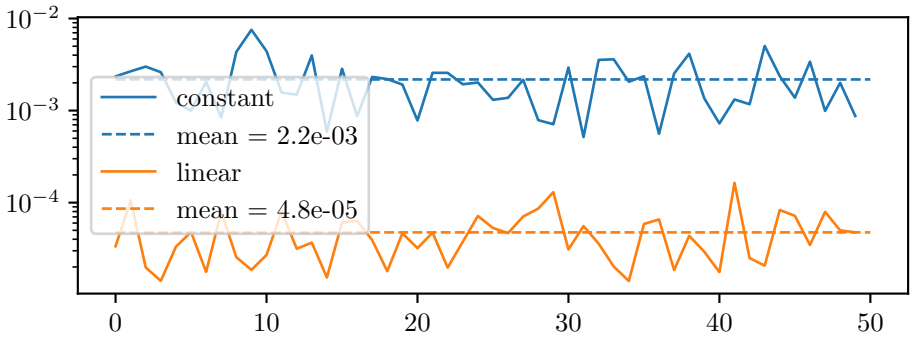


Figure 2.4: Comparison of constant and linear force importance ρ.

Overfitting can be mitigated via two approaches, either by using less complex models with a reduced number of parameters or by increasing the size of the dataset to decrease the probability of noise exhibiting a pattern.

Another form of regularization used to avoid overfitting is called *Early Stopping*. It involves dividing the dataset into distinct training and testing subsets, where the former is utilized for parameter optimization, and the latter serves as an out-of-sample validation to assess overfitting. Early stopping terminates the training process when the validation error begins to increase, ensuring avoidance of fitting sample-specific features in the data. [13]

2.3 Fingerprints

If Cartesian coordinates were to be fed into a neural network, the model would not only need to learn the translational and rotational symmetries but also the quantum invariance of identical particles. It is preferable to preprocess the input data in such a way as to incorporate these symmetries into the inputs and the model itself, and this way can the model focus on learning the desired yet unknown physical properties.

As we have seen in the section about empirical potentials, two and three-body terms can be effectively represented using the relative interatomic distances and inscribed angles. The three-dimensional structure of an atomic cluster can thus be represented in terms of these values.

2.3.1 Cut-off functions

A cut-off function $f_c(r; r_c)$ is employed to define only the energetically relevant regions close to the central atom.

The choice of the cut-off radius r_c is important because it can affect the accuracy of the fit, since it determines how many neighbouring atoms are considered when describing the environment of a central atom. If the cut-off radius is too small, important features may be missed, while a large cut-off radius increases the computational cost of the descriptor as more neighbouring atoms are included in the calculation. A good compromise between accuracy and computational cost is always system-dependent, and different values should be tested to achieve satisfying results.

The cut-off function must be differentiable and decay smoothly to zero in value, slope, and if possible, even higher derivatives at the cut-off radius. This is required to avoid discontinuities in the descriptor values and its derivatives, and consequently in the energy and its gradients, if atoms leave or enter the cut-off spheres in molecular dynamics simulations. [14]

Common choices for the cut-off functions are the cosine cut-off function

$$f_c(r_{ij}) = \begin{cases} \frac{1}{2}\left[\cos\left(\pi \frac{r_{ij}}{r_c}\right) + 1\right], & r_{ij} \leq r_c \\ 0, & r_{ij} > r_c \end{cases} \tag{2.3}$$

A well-known problem of this cut-off function is that its second derivative has a discontinuity at the cut-off radius. If the cut-off chosen is sufficiently large, the effect of this discontinuity becomes as small as needed.

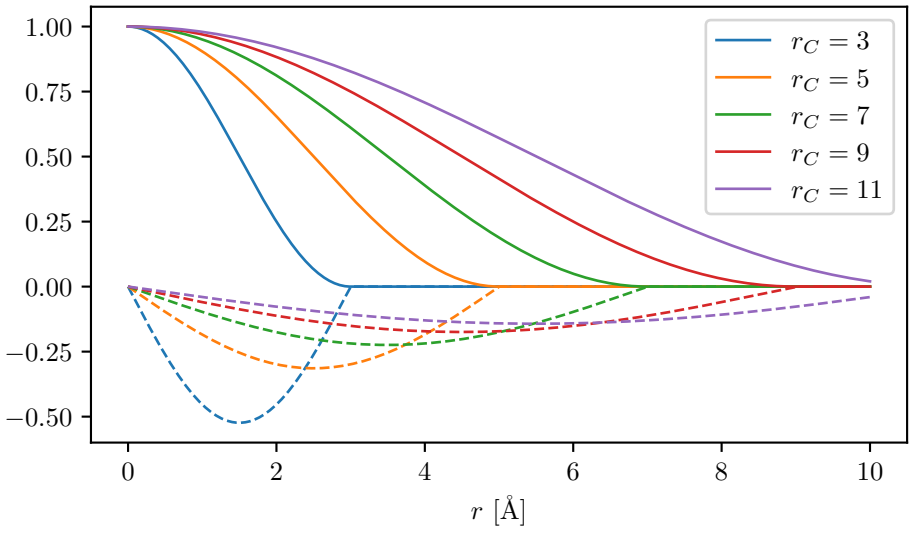


Figure 2.5: Cosine cut-off function for a selection of r_C.

Another possible cut-off function that has continuous first and second derivatives is the tanh cut-off function, defined as

$$f_c(r_{ij}) = \begin{cases} \tanh^3\left(1 - \frac{r_{ij}}{r_c}\right) & r_{ij} \leq r_c \\ 0 & r_{ij} > r_c \end{cases} \tag{2.4}$$

The advantage of this cut-off function is that its value as well as first and second derivatives go to zero at the cut-off radius r_c.

Another possible choice is the exponential cut-off function

$$f_c(r_{ij}) = \begin{cases} \exp\left(1 - \frac{1}{1-\left(\frac{r_{ij}}{r_c}\right)^2}\right) & r_{ij} \leq r_c \\ 0 & r_{ij} > r_c \end{cases} \tag{2.5}$$

with continuous derivatives up to infinite order at the cut-off radius.

In all these cut-off functions it is also possible to replace the term $\frac{r_{ij}}{r_C}$ by $x = \frac{r_{ij} - r_{c_i}}{r_c - r_{c_i}}$ with the inner cut-off $r_{c_i} < r_c$ such that

$$f_c(r_{ij}) = \begin{cases} 1 & r_{ij} < r_{c_i} \\ f_c(x) & r_{c_i} \leq r_{ij} \leq r_c \\ 0 & r_{ij} > r_C \end{cases} \tag{2.6}$$

The advantage of including an inner cut-off is the possibility of focusing the numerical range of function values to the chemically meaningful range of interatomic distances r_{ij}.

2.3.2 Radial symmetry functions

Radial symmetry functions are constructed as sums of two-body terms multiplied by one or more cut-off functions to ensure that the total symmetry function decays to zero in value and slope at the cut-off radius.

The G^1 symmetry function is just a sum of the cut-off functions with respect to all neighbouring atoms j

$$G_i^1 = \sum_j f_C(r_{ij}). \tag{2.7}$$

The radial atomic environment can also be described in terms of Gaussian functions for a central atom i

$$G_i^2 = \sum_j \exp\left\{-\eta \left(r_{ij} - r_S\right)^2\right\} \cdot f_C(r_{ij}). \tag{2.8}$$

For $\eta = 0$ we get the G_i^1. Recommended choices for parameters r_S and η are

- $r_S =$ `linspace(0, rC, n)` and $\eta = \left(\frac{2\pi}{r_S}\right)^2$
- $r_S = \frac{n}{n^{m/n}}$ and $\eta = \left(\frac{n}{n^{(n-m)/n}} - \frac{n}{n^{(n-m-1)/n}}\right)^2$

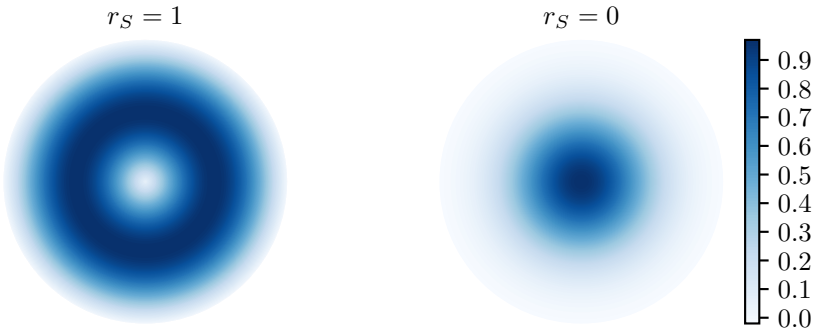


Figure 2.6: Polar plot of the G^2 radial symmetry function.

2.3.3 Angular symmetry functions

As we have seen with the three-body correction term in 1.34, the spatial orientation of the neighbouring atoms can be captured using the cosine of an inscribed angle. Angular symmetry functions are a sum of such cosine terms over all pairs of neighbouring atoms.

In the function G^4, the multiplication of three Gaussian functions ensures that only triplets of atoms, for which all three interatomic distances are less than the specified cut-off radius, are included in the summation

$$G_i^4 = 2^{1-\zeta} \sum_{j,k \neq i} (1 + \lambda \cos\theta_{ijk})^\zeta \cdot \exp\left\{-\eta \left(r_{ij}^2 + r_{ik}^2 + r_{jk}^2\right)\right\} \cdot f_C(r_{ij}) \cdot f_C(r_{ik}) \cdot f_C(r_{jk}). \tag{2.9}$$

The angular resolution of the function is determined by the parameter ζ. Higher values of ζ result in a narrower range of non-zero symmetry function values. As such, a set of angular functions with varying ζ values can be utilized to obtain the distribution of angles centred at each reference atom. This is analogous to the control of radial resolution in radial functions G^2 through the parameter η. Additionally, by appropriately selecting the values of η and r_C, which control the radial component, the angular distribution can be determined at various distances from the central atom.

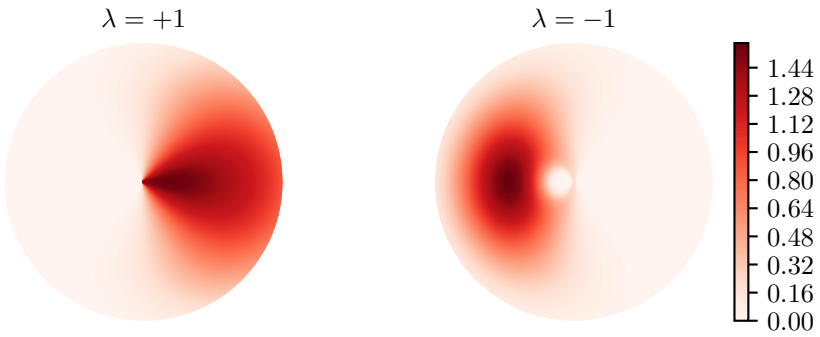


Figure 2.7: Angular symmetry function G^4 around a central atom positioned at origin and another atom at $x = 1$.

The range of values for individual symmetry functions used to characterize atomic environments can vary significantly. For numerical reasons, it is advantageous to precondition the range of values for each symmetry function as $G_i^k \leftarrow G_i^k - \frac{1}{N}\sum_{j=1}^{N} G_j^k$. A fundamental operation is to shift the mean value of each symmetry function to zero. Centring the range of values around 0 is often beneficial, as this corresponds to the centre of the nonlinear regions for most activation functions. This can also be achieved by shifting the centre of mass of all symmetry function values to zero.[15]

The relative importance of all symmetry functions can be balanced by rescaling the range of values for each symmetry function to a predefined interval $[S_{\text{min}}, S_{\text{max}}]$, for example $[-1, 1]$, by applying

$$G_i^{\text{scaled}} = \frac{G_i - G_i^{\text{min}}}{G_i^{\text{max}} - G_i^{\text{min}}} \cdot (S_{\text{max}} - S_{\text{min}}) + S_{\text{min}}, \tag{2.10}$$

where G_i^{min} is the smallest value of the function G_i occurring in any atomic environment and G_i^{max} its largest value in the dataset.

Similar to training on forces, if the symmetry functions are scaled, their derivatives have to be rescaled must also be rescaled to maintain the consistency of the model's gradient. For the min-max normalization, eq. 2.10, the derivatives need to be divided by $G_{\text{max}} - G_{\text{min}}$. However, when using PyTorch, these transformations are a part of the computational graph and so are automatically accounted for.

To accurately represent the atomic environment, a typical approach involves the use of 50 symmetry functions, with hyperparameters selected to optimally

capture the environment. The initial step in this process is to determine a comprehensive and manageable pool of candidate symmetry functions. [16]

For the additive part, two distinct sets of radial symmetry functions, G^2, are generated. The first set is centred on the reference atom i, with $r_S = 0$, and the width varies as

$$\eta_m = \left(\frac{n^{m/n}}{r_C}\right)^2, \tag{2.11}$$

where n represents the number of intervals in which the space is divided and $m = \{0, 1, \ldots, n\}$. The second set is centred along the path between the central atom and its neighbours, at increasing distances $r_{S,m} = \frac{r_C}{n^{m/n}}$, while the Gaussian widths are chosen as

$$\eta_{s,m} = \left(r_{s,n-m} - r_{s,n-m-1}\right)^2 \tag{2.12}$$

to produce narrow Gaussians close to the central atom and wider ones as the distance increases. This creates a finer grid near the central atom, where small variations in position have a larger effect on the potential.

The angular symmetry functions are chosen analogously with values for η selected according to equation 2.11, $\lambda = \pm 1$ and a few values of ζ chosen on a logarithmic scale.

From such a set of symmetry functions, a smaller subset can be chosen using the *farthest point sampling* technique. In this scheme, points are selected successively to maximize their Euclidean distance. After the first fingerprint is chosen arbitrarily, each additional fingerprint is determined according to

$$k = \text{argmax}\Big(\min_j |X_k - X_j|\Big), \tag{2.13}$$

where j represents all previously chosen features. This process is repeated until all features have been selected. The distance between two fingerprints is defined as $|f - g| = \sum_x |f(x) - g(x)|$.

An alternative method to simplify the input space involves identifying and removing symmetry functions that have minimal impact on training performance, i.e. discarding all fingerprints with a range below a predefined threshold.

Chapter 3

Results

3.1 Computational details

3.1.1 Molpro

The reference ab initio data were constructed using Molpro [17].

Firstly, we compared the convergence behaviour for Hartree-Fock interaction energy, MP2 and MP2 with F12 correction, using several basis sets, namely AVDZ, AVTZ, AVQZ and AV5Z. We found the best convergence for the MP2-f12 explicitly correlated method using the Dunning correlation consistent (cc) basis sets with added diffuse functions (aug), denoted aug-cc-pV5Z.

The energy threshold was set to `energy = 1.d-16` threshold for forces and `gradient = 1.d-10`. The neglection of two-electron integrals was set to `twoint = 1.d-19`.

Interaction energies for both dimers and trimers were computed using the counterpoise correction.

3.1.2 PyTorch

The models used were constructed using the PyTorch machine-learning library. In particular, we have taken great use of their automatic differentiation module `torch.autograd` and the computational graph allowing us to treat all the trained models the same, regardless of whether it was a simple empirical potential, a fully-connected neural network or even something more complicated.

These models were then trained using the Levenberg-Marquardt algorithm, the implementation of which is in the appendix A.1.

3.2 Dimer

For argon dimer we compared three model potentials, namely, empirical, Buckingham type potential with the functional representation

$$V_{\text{empirical}} = Ae^{-\alpha r-\beta r^2} - \frac{\gamma}{r^6} \tag{3.1}$$

and as we were not sure about the contribution of the repulsion term, we chose the parameter A to be represented by a neural network

$$V_{\text{combined}} = A_{\text{NN}} e^{-\alpha r - \beta r^2} - \frac{\gamma}{r^6}. \tag{3.2}$$

Lastly, we consider a full neural network potential with no "physically meaningful" terms.

Below, in figure 3.1, we show the fit of these potentials on an argon dimer reference data.

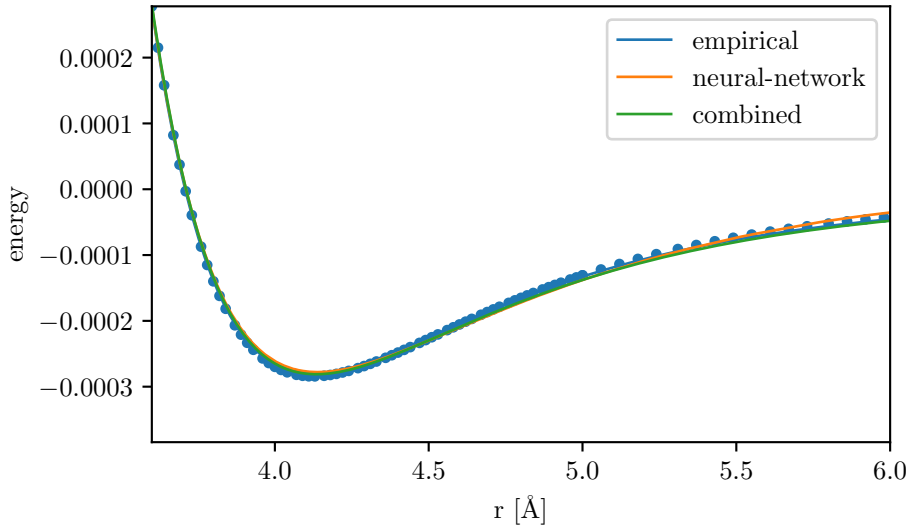


Figure 3.1: Argon dimer pair interaction potential.

All models we capable of minimizing the loss below a predefined threshold of 10^{-10}.

3.3 Trimer

For three-body interactions, we compared the empirical potential defined in equation 1.42 with a pure neural network potential for argon and helium. Around 50 data points were obtained from the costly ab initio software for C_{2v} geometry, that is, isosceles triangles with interatomic distances r_1 and $r_2 = r_3$.

The empirical potential was able to capture the reference data rather well, however, the neural network potential completely failed. It was unable to capture even the simplest physical features, e.g. qualitatively correct behaviour on the diagonal $r_1 = r_2 = r_3$.

To ensure the implementation of the neural network potential is correct, we fit the model on an artificial dataset obtained from the Lennard-Jones potential 1.38 for the C_{2v} geometries. To obtain a relatively small error, we had to train the model on a considerably greater number of reference data points, that is 10^3. The results of this fit, for equilateral triangles $r_1 = r_2 = r_3$ are shown in figure 3.4.

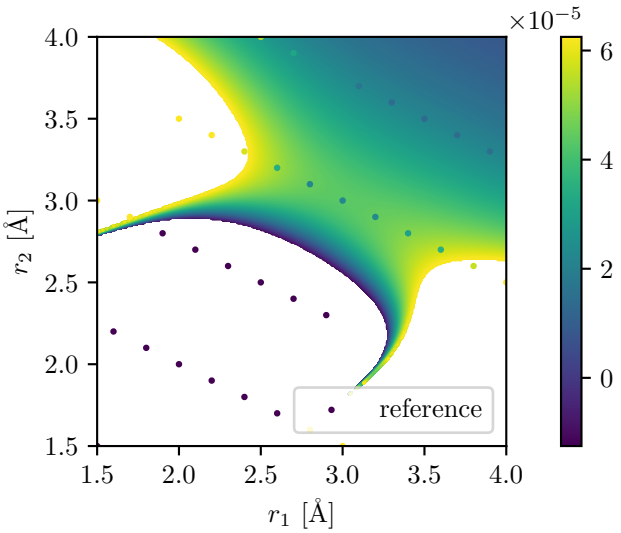


Figure 3.2: Potential energy surface for the C_{2s} configurations of helium trimer.

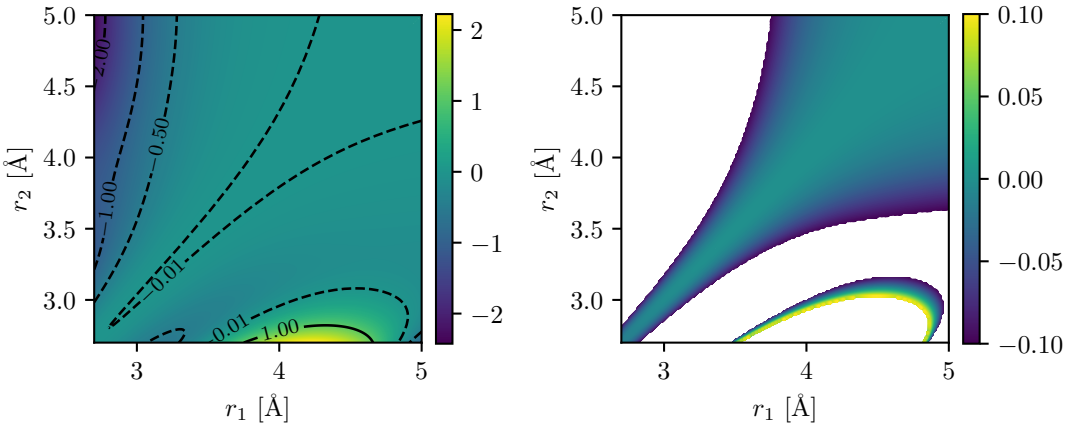


Figure 3.3: Potential energy surface for the C_{2s} configurations of argon trimer.

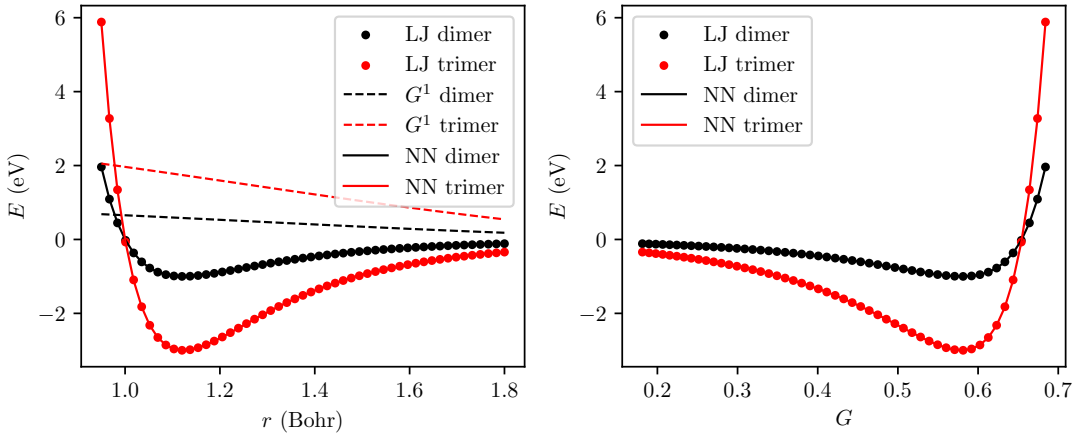


Figure 3.4: Neural network fit of the Lennard-Jones potential.

Conclusion

We have analysed the most basic of systems, dimer, and trimer of two noble gasses. From the quantum chemistry software, we obtained reference data on the potential energy surface and corresponding forces. We optimized several model potentials on these potential energy surfaces and compared the obtained precision.

The semi-empirical model potentials were able to capture all physical features of the dataset and minimize the loss below a predefined threshold. This was to be expected, since their functional form entails these features, only the parameters need to be set for the specific medium.

We found models comprised of the neural networks to be unable to learn on such small datasets. In case of the argon dimer, the number of reference data points was smaller than the number of parameters of the neural network. Later we fit the neural network on an artificial dataset obtained for the C_{2v} geometries from the analytical functional form of the Lennard-Jones potential and showed, that this method is indeed capable of capturing important physical features, however, necessitates a significantly larger dataset.

Bibliography

[1] Liguo Kong, Florian A. Bischoff, and Edward F. Valeev. Explicitly correlated r12/f12 methods for electronic structure. 112(1):75–107. Publisher: American Chemical Society.

[2] Ilya G. Kaplan. *Intermolecular Interactions: Physical Picture, Computational Methods and Model Potentials*. Wiley, 1 edition.

[3] A. J. Stone. *The theory of intermolecular forces*. Oxford University Press, second edition edition.

[4] W. Heitler and F. London. Wechselwirkung neutraler atome und homöopolare bindung nach der quantenmechanik. 44(6):455–472.

[5] B. M. Axilrod and E. Teller. Interaction of the van der waals type between three atoms. 11(6):299–300.

[6] Hans-Joachim Böhm and Reinhart Ahlrichs. A study of short-range repulsions. 77(4):2028–2034.

[7] Michael J. Cohen and John N. Murrell. An analytic function for the three-body potential of he3. 260(3):371–376.

[8] Thomas F. Edgar, David Mautner Himmelblau, and Leon S. Lasdon. *Optimization of chemical processes*. McGraw-Hill chemical engineering series. McGraw-Hill, 2nd ed edition.

[9] Hao Yu. Levenberg–marquardt training. page 16.

[10] Kurt Hornik, Maxwell Stinchcombe, and Halbert White. Multilayer feedforward networks are universal approximators. 2(5):359–366.

[11] Thomas B. Blank and Steven D. Brown. Adaptive, global, extended kalman filters for training feedforward neural networks. 8(6):391–407. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/cem.1180080605.

[12] A. Pukrittayakamee, M. Malshe, M. Hagan, L. M. Raff, R. Narulkar, S. Bukkapatnum, and R. Komanduri. Simultaneous fitting of a potential-energy surface and its corresponding force fields using feedforward neural networks. 130(13):134101.

[13] Pankaj Mehta, Marin Bukov, Ching-Hao Wang, Alexandre G. R. Day, Clint Richardson, Charles K. Fisher, and David J. Schwab. A high-bias, low-variance introduction to machine learning for physicists. 810:1–124.

[14] Jörg Behler. Atom-centered symmetry functions for constructing high-dimensional neural network potentials. 134(7):074106.

[15] Jörg Behler. First principles neural network potentials for reactive simulations of large molecular and condensed systems. 56(42):12828–12840. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/anie.201703114.

[16] Giulio Imbalzano, Andrea Anelli, Daniele Giofré, Sinja Klees, Jörg Behler, and Michele Ceriotti. Automatic selection of atomic fingerprints and reference configurations for machine-learning potentials. 148(24):241730.

[17] Hans-Joachim Werner, Peter J. Knowles, Gerald Knizia, Frederick R. Manby, and Martin Schütz. Molpro: a general-purpose quantum chemistry program package. 2(2):242–253. _eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/wcms.82.

List of Figures

Appendix A

Attachments

A.1 Implementation of Levenberg-Marquardt algorithm

```
import torch

class LevenbergMarquardt(torch.optim.Optimizer):
    """
    PyTorch implementation of the Levenberg-Marquardt optimization algorithm.
    This optimizer is designed for a single parameter group.

    Parameters:
        params (iterable): Iterable of parameters to optimize.
        mu (float, optional): Initial value of the damping factor (default=10**3).
        mu_factor (float, optional): Factor by which mu is multiplied during line search (default=5).
        m_max (int, optional): Maximum number of line search iterations (default=10).

    Example usage:
        optimizer = LevenbergMarquardt(model.parameters(), mu=100, mu_factor=10, m_max=5)
        for epoch in range(num_epochs):
            def closure():
                optimizer.zero_grad()
                outputs = model(inputs)
                errors = loss_function(outputs, targets)
                errors.backward()
                return errors
            loss = optimizer.step(closure)
    """

    def __init__(self, params, mu=10**3, mu_factor=5, m_max=10):
        """
        Initializes Levenberg-Marquardt optimizer with the specified parameters.
        """
        self.mu = mu
        self.mu_factor = mu_factor
        self.m_max = m_max

        defaults = dict(mu=self.mu, mu_factor=self.mu_factor, m_max=self.m_max)
        super(LevenbergMarquardt, self).__init__(params, defaults)

        self.numel = sum([
                param.numel() for group in self.param_groups
                for param in group['params'] if param.requires_grad
            ])

    @torch.compile
    def jacobian(self, targets):
        """
        Compute the Jacobian matrix for the given targets.

        Parameters:
```

```
        targets (torch.Tensor): Target tensor for which Jacobian is computed.

    Returns:
        torch.Tensor: Computed Jacobian matrix.
    """
    J = torch.empty(targets.shape[0], self.numel)

    for i in range(targets.shape[0]):
        J[i] = torch.hstack([
                d.view(1, -1) if d is not None else torch.tensor([0.]).view(1, -1)
                for d in grad(targets[i], self.param_groups[0]['params'],
                create_graph=True, retain_graph=True, allow_unused=True)
            ])

    return J

@torch.no_grad()
def loss(self, errors):
    """
    Compute the loss using the Mean Squared Error (MSE) criterion.

    Parameters:
        errors (torch.Tensor): Errors tensor.

    Returns:
        torch.Tensor: Computed loss.
    """
    return errors.T @ errors

@torch.no_grad()
def update_weights(self, update):
    """
    Update the model weights using the computed update.

    Parameters:
        update (torch.Tensor): Computed update for the model weights.
    """
    update = update.view(-1)

    offset = 0
    for group in self.param_groups:
        for param in group['params']:
            numel = param.numel()
            param.add_(update[offset: offset + numel].view_as(param))
            offset += numel

def step(self, closure=None):
    """
    Performs a single optimization step using the Levenberg-Marquardt algorithm.

    Parameters:
        closure (callable, optional): A closure that reevaluates the model and returns the loss.

    Returns:
        float: Computed loss value after the optimization step.
    """
    assert len(self.param_groups) == 1

    # Make sure the closure is always called with grad enabled
    closure = torch.enable_grad()(closure)

    # Compute errors from closure
    errors = closure()

    # Compute Jacobian matrix
    J = self.jacobian(errors)

    # Compute updates using the Levenberg-Marquardt formula
    updates = -torch.inverse(
                J.T @ J + self.mu * torch.eye(self.numel)
            ) @ J.T @ errors

    # Update weights with the computed updates
```

```
self.update_weights(updates)

# Line search for mu
for m in range(self.m_max):

    # Check if loss has decreased
    if self.loss(closure()) < self.loss(errors):
        break

    # Restore weights
    self.update_weights(update=-updates)

    self.mu *= self.mu_factor

    # Compute new updates
    updates = -torch.inverse(
                J.T @ J + self.mu * torch.eye(self.numel)
            ) @ J.T @ errors

    # Update weights
    self.update_weights(update=+updates)

if m < self.m_max:
    self.mu /= self.mu_factor

# Return the computed loss after the optimization step
return self.loss(closure()).item()
```